Ўзбекистон Республикаси
Олий ва Ўрта махсус Таълим Вазирлиги
Мирзо Улуғбек номидаги Ўзбекистон Миллий университети
Механика-математика факультети
Программалаш ва тармоқ технологиялари кафедраси

Программалаш асослари

фанидан мажмуа

(5480100 – Амалий математика ва информатика)

Тузувчилар: ф.-м.ф.н., доцент, **Ш.Ф.Мадрахимов**
ф.-м.ф.д., проф.в.б., **Н.А.Игнатъев**
асс., **М.Р.Бобожонов**
ўқт., **С.М.Джураев**

Тақризчи: ф.-м.ф.н., доцент, **А.Т.Хайдаров**

Тошкент - 2010 йил

Ўқув – услубий мажмуа таркиби

1. Фан дастури
2. Ишчи фан дастур
3. Календар тематик режа
4. Баҳолаш мезонлари ва баллар тақсимоти
5. Таълим технологияси
6. Маъруза матнлари
7. Тест топшириқлари
8. Назорат саволлари
9. Реферат мавзулари
10. Курс ишлари мавзулари
11. Малакавий битирув ишлари мавзулари
12. Мустақил таълим учун саволлар
13. Глоссарий
14. Слайдлар
15. Адабиётлар

ЎЗБЕКИСТОН РЕСПУБЛИКАСИ
ОЛИЙ ВА ЎРТА МАХСУС ТАЪЛИМ ВАЗИРЛИГИ

Руйхатга олинди

№ БД 5480100-3,1,01
2008 йил “23” август

Ўзбекистон Республикаси Олий ва ўрта махсус таълим вазирининг
2008 йил “23” августдаги
“263” - сонли буйруғи билан тасдиқланган

«ПРОГРАММАЛАШ АСОСЛАРИ»
фанининг
ЎҚУВ ДАСТУРИ

Билим соҳаси: 400000 – Фан
Таълим соҳаси: 480000 – Амалий математика ва информатика
Таълим йўналиши: 5480100 – Амалий математика ва информатика

Тошкент – 2008

Фаннинг ўқув дастури Олий ва ўрта махсус, касб-ҳунар таълими ўқув-методик бирлашмалари фаолиятини Мувофиқлаштирувчи Кенгашнинг 200_ йил “20”августдаги “4” - сон мажлис баёни билан маъқулланган.

Фаннинг ўқув дастури МИРЗО УЛУҒБЕК номидаги ЎЗБЕКИСТОН МИЛЛИЙ УНИВЕРСИТЕТИДА ишлаб чиқилди.

Тузувчилар:

Мадрахимов Ш.Ф. - физика- математика фанлари номзоди, ЎзМУ доценти;

Гайназаров С.М. - физика- математика фанлари номзоди, ЎзМУ доценти.

Тақризчилар:

Садуллаев Р. – техника фанлари доктори, профессор, ЎзРУ ФА МИТИ лаб. мудири.

Ҳайдаров А. - физика- математика фанлари номзоди, ЎзМУ доценти;

Фаннинг ўқув дастури Мирзо Улуғбек номидаги Ўзбекистон Миллий УниверчсМУ илмий-методик кенгашида тавсия қилинган (200_ йил “___” _____даги “___” -сонли баённома).

Кириш

Программалаш асослари фанининг бош мақсади талабаларга қўйилган масалани ечувчи компьютер программасини тузишга ўргатишдир. Шу мақсадда программалаш тиллари ва муҳитлари ҳақида умумий тушунчалар берилади ва бу тиллардан фойдаланишга ўргатилади.

Фан назарий ва амалий қисмлардан иборат. Назарий қисм информатика ва ҳисоблаш техникаси, алгоритмлар, C/C++ программалаш тили, Assembler тили, Visual C++ объектга йўналтирилган программалаш муҳити бобларидан ташкил топган.

Дастурда программалашга киришнинг назарий асоси бўлган алгоритмларга алоҳида эътибор қаратилган. Бу ерда алгоритмларни тавсифлаш ва кейинчалик компьютерда амалга ошириш учун зарур бўлган бир қатор математик тушунчалар - такрорлаш, ёрдамчи алгоритм, рекурсия, хотира, массив, индекс, параметр ва ҳ.к. киритилиб, турли хил синф масалаларининг алгоритмлари тузилади.

Программалаш тили - тузилган алгоритмни компьютер амалга ошириш учун воситадир. Бу ўринда турли мураккабликдаги синтаксис ва семантикага эга бўлган тиллардан фойдаланиш мумкин. Фанда C++ тилининг алфавити, тил қурилмаларининг умумий синтаксиси, берилганлар турлари, операциялар ва функциялар қаралади, оқим билан ишлаш, кўрсатгичлар, амалларни қайта юклашлар, функцияларни қайта юклаш, қолиплар ва модулли программмалаш ва графика модули билан ишлаш ўргатилади.

Ҳозирда мавжуд программалаш тиллари икки тоифага бўлинади, биринчисига Assembler тили кирса, иккинчисига қолган барча тиллар киради. Юқори босқич тиллари у ёки бу тадбиқий масалаларни (иқтисодий, математик ва ҳакоза) ечишга мослашган бўлиб, улар бундай масалаларни ечиш алгоритмини ифодалаш воситаларини ўз ичига олади. Assembler эса процессор архитектурасининг акси, яъни унинг мантиқий қурилмалари, иш ҳолатлари ва амал қилиш схемаларини ўзида акслантиради. Assemblerда ёзилган программа юқори самарадорлиги, минимал ҳажми ва бажарилишининг максимал тезкорлиги билан ажралиб туради. Шунинг учун Assembler тезлик ва хотирага чекловлар қўйилган ҳолларда, қурилмаларга “қаттиқ” боғланган драйверлар яратишда ва компьютерга ностандарт аппарат қурилмаларини улаш масалаларида қўл келади. Assembler тилини ўрганиш методик нуқтаи-назардан ҳам аҳмиятлидир. Assembler тилини ўрганиш – процессор, ва умуман компьютер ишлашини тўлиқ ўрганиш, C/C++ ва бошқа тилларда программа тузувчи учун юқори босқич тиллар формализмлари “ортида” қандай жараёнлар рўй бераётганлигини англал имкониятини беради.

Процессорнинг ҳимояланган режимида (Windows ОСда) амал қилувчи 32-разрядли процессорлар принципиал жиҳатдан янги программалаш технологияларини юзага келишига сабаб бўлди. Объектга йўналтирилган программалашга асосланган C++ ушбу талабаларга жавоб берувчи программалаш тилидир. C++ тили асосида Visual C++ визуал программалаш муҳити яратилди ва дунёда ўзининг салмоқли ўрнига эга бўлди. Visual C++ муҳитида программалаш объектга йўналтирилган программалашнинг назарий асосларини тушунтириш, синфлар ва улар шажарасини ҳосил қилиш, визуал программалаш муҳити, шакл, компоненталар, иловалар яратиш ва шу каби таянч тушунчалар ҳақида маълумот бериш ва улар билан ишлаш сабоқларини бериш орқали амалга оширилади.

Программалашга ўргатиш мос масалаларга таянган ҳолда олиб борилади. Фаннинг амалий қисмида алгоритмлар, C++ ва Assembler программалаш тили, Visual C++ муҳитида ишлаш бўйича масалаларни ўз ичига олади.

Программалаш асослари фани бевосита ва билвосита Тизимли программалаш, Берилганлар базасини бошқариш тизимлари, Компьютер графикаси ва Компьютер тармоқлари фанлари билан узвий боғлиқдир ва бу фан компьютер технологиялари бўйича мутахассис тайёрлашда умумий асос ролини ўйнайди.

Ўқув фанининг мақсади ва вазифалари

Фанни ўқитишдан мақсад - Информатика ва ахборот технологиялари йўналишининг бакалавр босқичи талабаларига программалаш асосларини етарли даражада ўқитиш, шу билимларга таянган ҳолда компьютер моделлаштиришга келадиган тадбиқий масалаларнинг программа таъминотини амалга оширишга ўргатиш ва ихтисослик фанларини ўзлаштиришда таянч билимларга эга бўлиш.

Фаннинг вазифалари: Масала ечишнинг алгоритмик асосларини ўрганиш, компьютер ишлашининг тамоили, программалаш тиллари синфлаш, компьютерда берилганлар ва буйруқларни тасвирланиши, C++ тилида программалаш, Ассемблер тили, объектга йўналтирилган программалаш технологиялари, визуал программалаш муҳитида ишлаш бу фаннинг асосий вазифалари ҳисобланади.

Фан бўйича талабаларнинг билим, малака ва кўникмаларига қўйиладиган талаблар

«Программалаш асослари» фанини ўзлаштириш жараёнида амалга ошириладиган масалалар доирасида бакалавр:

- Алгоритмлар ва берилганларни тасвирлаш. Алгоритмнинг асосий шакллари ва уларни тасвирлаш. Турли тоифадаги масалаларни ечиш алгоритмлари. Саноқ тизимлари. Берилганларнинг ички кўриниши ва берилганлар турларини ўзгартириш (алмаштириш). Берилганлар турлари устида амаллар. Берилганлар тузилмалари ва уларни қайта ишлаш технологиялари фан доирасида билим ва малакага эга бўлишлари, уларнинг моҳиятларини тушунишлари керак;
- C++ программалаш тили. Тил алфавити, синтаксиси ва семантикаси, программа тузилиши, берилганларнинг асосий турлари, қиймат бериш, тармоқланувчи ва такрорлаш операторлари, функциялар, кўрсатгичлар, ўқиш-ёзиш оқимлари, фавқулотда ҳолатларни қайта ишлаш, синфлар, объектга йўналтирилган программалаш технологиялари, контейнерларни тушунииши ва уларни қўллаган ҳолда тадбиқий масалаларни еча олиши керак;
- Ассемблер тили. Процессор, регистр ва шина тушунчалари, реал ва ҳимояланган режим архитектуралари, программаларнинг сегмент тузилиши, стек, узилишлар тизими, ўқиш-ёзиш тизимлари, Ассемблер макровоситалари, LDT ва GDT жадваллари, ҳимояланган режимда адреслаш усуллари, ҳимояланган режимда узилишларни қайта ишлаш, Windows иловасини яратиш ҳақида умумий тасаввурларга эга бўлиши ва махсус ҳолатлар учун ассемблер программасини ярата олиши керак;
- Объектга йўналтирилган ва визуал программалаш. Синфлар, синф майдонлари ва методлари, конструктор ва деструкторлар, ворислик, виртуал методлар, илова яратиш устаси. консол ва диалог иловалари. Кўпдарчали ва бир дарчали иловалар, хабарлар ва буйруқлар, WIN32 тизими хабарлар харитасини ишлатиш, график интерфейс билан ишлаш, NET технологияси, C# тилини хусусиятлари билиши ва программалаш муҳитида ишлай олиши ва Windows иловаларини ярата олиши керак.

Фаннинг ўқув режадаги бошқа фанлар билан ўзаро боғлиқлиги ва услубий жиҳатдан узвий кетма-кетлиги

Мазкур фанни ўзлаштириш Дискрет математика, Математик мантиқ ва Алгоритмлар назарияси фанлари билан узвий боғланган. Фан мазмуни Информатика соҳасидаги Тизимли программалаш, Операцион тизимлари, Web программалаш, Берилганлар базасини бошқариш тизимлари, Тадбиқий масалаларни математик моделлаштириш тизимлари (Mathcad, Maple, Statistica ва бошқалар) ўрганиш учун таянч ҳисобланади.

Фаннинг ишлаб чиқаришдаги ўрни

Мазкур дастурга кўра ушбу фан доирасида кўплаб модел масалалар ўрганиладики, бу мазкур фанни чуқур ўрганган ҳар бир бакалавр олган билим ва кўникмаларини ишлаб-чиқаришда, илмий-тадқиқот ишларида, шунингдек, таьлим тизимида самарали фойдаланиши имконини беради.

Фанни ўқитишда замонавий ахборот ва педагогик технологиялар

Программалаш асослари фанини ўқитиш маъруза, амалий машғулотлар ва мустақил таълим кўринишида олиб борилади. Фаннинг мазмуни уни ўқитишда замонавий ахборот технологияларидан, хусусан, компьютер техникасидан фойдаланишни тақоза этади. Шу билан биргаликда фанни ўқитишда илғор ва замонавий усулларидан фойдаланиш, янги информацион-педагогик технологияларни тадбиқ қилиш катта самара бериши шубҳасиз. Хусусан, фанни ўқитишда электрон дарсликликлардан ва мустақил таълим учун масофавий таълим сайтларидан фойдаланиш мақсадга мувофиқ ҳисобланади.

Асосий қисм

Фаннинг назарий машғулотлари мазмуни

Ахборот ва ахборот технологиялари. Информация турлари: узлуксиз ва дискрет информация. Информатика. ЭҲМни ривожланиш тарихи ва қўллаш соҳалари. ЭҲМ авлодлари, ЭҲМ нинг асосий қурилмалари. ЭҲМда масала ечиш босқичлари. Саноқ системалари. Маълумотларнинг ЭҲМ хотирасидаги кўриниши. Математик мантиқ элементлари.

Алгоритм тушунчаси, унинг хоссалари ва тасвирлаш усуллари. Блок схема. Чизиқли алгоритмлар, тармоқланувчи алгоритмлар ва такрорланувчи алгоритмлар. Бирор синф масаласи ёки синфлар композицияси учун алгоритмлар яратиш.

С++ тили синтаксиси. Берилганлар турлари. Ўзгарувчилар ва ифодалар. Операторлар - “ифода”, тармоқланувчи, такрорлаш ва бошқарувни узатиш операторлари. Кўрсатгичлар ва массивлар. Фойдаланувчи томонидан аниқланадиган турлар. Модулли программалаш. Функциялар эълон қилиш ва аниқлаш. Локал ва глоба параметрлар. Рекурсив функциялар. Фунцияларни қайта юклаш. Функция қолиплари. Main() функцияси. Стандарт кутубхона функциялари. Препроцессор директивалари. Идентификаторларнинг амал доираси. Стандарт оқимлар. Фавқулотда ҳолатлар.

Процессор, регистр ва шина тушунчалари ва уларни вазифалари. Реал режим архитектураси. Хотира фазосининг тақсималниши. Процессор регистрлари. Программаларнинг сегмент тузилиши. Стек. Узилишлар тизими. Ўкиш-ёзиш тизими. Берилганларни тавсифлаш. Арифметик буйруқлари. Адреслаш усуллари. Шарт ва шартли ўтишлар. Такрорлашлар. Шартсиз ўтишлар. Қисмпрограммаларни чақириш. Assembler макровоситалари. MS DOS иловаларининг турлари ва уларни яратиш. Узилишларини қайта ишлаш. Резидент программалар. Процессорнинг ҳимояланган иш режими. Ҳимояланган режим регистрлар тизими. LDT ва GDT жадваллари. Ҳимояланган режимда адреслаш усули. Ҳимояланган режимда узилишларни қайта ишлаш. Assemblerда Windows иловасини яратиш.

Синфлар. Синфни ва объектларни тавсифлаш. Синф конструктори. Синф майдонлари ва методлари. Конструктор ва Деструкторлар. Амалларни қайта юклаш. Ворислик. Мурожаат калити. Оддий ворислик. Виртуал методлар. Тўпламли ворислик. Синфлар қолиплари, уларни яратиш ва ишлатиш. Фавқулотда ҳолатларни қайта ишлаш. Фавқулотда ҳолат синтаксиси. Фавқулотда ҳолатни илиб олиш. Бир турни иккинчисига келтириш. Стандарт кутубхона. Оқим синфлари. Стандарт оқимлар. Берилганларни форматлаш. Оқимлар билан алмашиш методлари. Файл ва сатр оқимлари. Сатрлар. Сатр устида амаллар. Сатр функциялари. Контейнерлар. Кетма-кет ва ассоциатив контейнерлар. Итераторлар ва функционал объектлар. Стандарт алгоритмлар.

Илова яратиш устаси. Консол ва диалог иловалари. Кўпдарчали ва бир дарчали иловалар. Хужжатлар қолипи. Диалог дарчалар ва оддий бошқариш элементлари, диалог синфини яратиш, киритмалар яратиш, “усталар” яратиш. Рўйхатлар синфи, чизиқли индикатор ва чизиқди регулятор синфлари. Хабарлар ва буйруқлар. Хабарларни қайта

ишлаш, хабарлар харитаси, WIN32 тизими хабарлар харитасини ишлатиш. График интерфейс билан ишлаш. Матнларни акс эттириш. График қаламни ишлатиш, графика чизиш. Матнлар билан ишлаш синфлари. Ҳолат сатрини ишлатиш. Хужжатларни чоп этиш. Visual C++ тилнинг хусусиятлари. NET технологияси, C# тилини хусусиятлари.

Амалий машғулотларни ташкил этиш бўйича кўрсатма ва тавсиялар

Амалий машғулотлар ўтказилишидан мақсад программалаш бўйича олинган назарий билимларни амалда мустаҳкамлаш ва турли тоифадаги масалаларни ечишга қўллашдан иборат. Амалий машғулотларни бир қисми аудиторияда доскада ечилиши билан ўтказилса, унинг катта қисми бевосита компьютерда амалга оширилиш керак.

Фан амалий машғулотларининг мазмуни

Саноқ системалари. Бир саноқ системасидан иккинчисига ўтиш. Берилганларни компьютер хотирасида тасвирланиши. Кодлаш.

Алгоритмни тасвирлаш усуллари. Блок схема. Чизиқли алгоритмлар, тармоқланувчи алгоритмлар ва такрорланувчи алгоритмлар. Бутун сонли арифметика масалалари. Ичма-ич жойлашган такрорланувчи жараенлар, итерацион жараёнлар. Берилганлар кетма-кетлигини сақлаш зарур масалалар. Кетма-кетликларни тартиблаш, оддий саралаш масалалари.

Берилганлар турлари. Программа тузилиши. Ўзгарувчилар, амаллар, ифодалар. “Ифода” оператори. Тармоқланувчи ва такрорлаш оператори. Кўрсатгичлар. Мурожаатлар. Бир ва икки ўлчамлаи массивлар. Тузилмалар. Бирлашмалар. Функцияларни эълони ва аниқлаш. Оддий функциялар. Функция параметрлари ва қайтарувчи қиймати. Рекурсив функциялар. Стандарт функциялар. Препроцессор директивалари. Берилганларнинг динамик тузилмалари: чизиқли рўйхатлар, стеклар, навбатлар ва бинар дарахтлар. Файл структуралар билан ишлаш, саралаш ва қидириш алгоритмлари, ифодалар ҳисоблаш, интерпретатор яратиш, криптография.

Ассемблер берилганлар тури. Буйруқлар формати. Assemblerларда программа тузилиши. Арифметик буйруқлар. Мантиқий буйруқлар. Ўтиш буйруғи. Занжирли буйруқлар. Assemblerдаги мураккаб тузилишли берилганлар. Assemblerда макроаниқлаш ва макробуйруқлар. Assembler тилидаги процедуралар. Узилишлар. Узилишлар назоратчиси. Драйверлар. Микропроцессорнинг ҳимояланган иш режими. Программа компиляцияси. Ҳимояланган режимда узилишларни қайта-ишлаш.

Синфлар ва уларни эълон қилиш. This кўрсатгичи. Конструкторлар. Синф майдонлари ва методлари. Унар ва бинар амалларни қайта юклаш. Қиймат бериш, new, delete, функцияларни чақириш ва индекслаш амалларини қайта юклаш. Ворислик, мурожаат калити. Оддий ворислик ва тўпламли ворислик. Синф қолипини яратиш. Фавқулотда ҳолатларни қайта ишлаш. Интернет учун программалаш, FTP ва HTTP форматлари билан ишлаш. Стандарт кутубхоналар билан ишлаш.

Илова яратиш устаси. Консол илова ва диалог иловалари. Кўпдарчали илова. Диалог дарчалар ва оддий бошқариш элементлари. Рўйхатлар синфи, чизиқли индикатор ва чизиқди регулятор синфлари. Хабарларни қайта ишлаш. WIN32 тизими хабарлар харитасини ишлатиш. График қаламни ишлатиш, графика чизиш. Матнлар билан ишлаш синфлари, содда матн таҳририни яратиш. Воситалар панелини яратиш. Ҳолат сатрини ишлатиш. Хужжатларни чоп этиш. Visual C++ тилнинг хусусиятлари. NET технологияси, C# тилини хусусиятлари.

Илова: Келтирилган мавзуларни камида 70-80%ини талабалар ўзлаштиришлари шарт.

Мустақил ишни ташкил этишнинг шакли ва мазмуни

Мустақил ишларни бажариш жараёнида талабалар қуйидаги ишларни бажарадидар:

- Дарслик ва ўқув қўлланмалар асосида фан мавзулари бўйича назарий тайёргарлик кўриш, амалий ва лаборатория машғулотларига тайёрланиш;
- Тарқатма материаллар бўйича маърузаларни чуқур ўзлаштириш;
- Фан мазмунида кўрсатилмаган программалаш тиллари ва муҳитлари билан танишиш ва қиёсий таҳлил қилиш;
- Масофавий таълим орқали программалаш билан турдош фанлар бўйича ўқув курсларида қатнашиш ва мос сертификатларга эга бўлиш тавсия қилинади.

Мустақил иш мавзулари

Информация турлари: узлуксиз ва дискрет информация. ЭҲМ авлодлари, ЭҲМ нинг асосий қурилмалари. ЭҲМда масала ечиш босқичлари. Саноқ системалари. Маълумотларнинг ЭҲМ хотирасидаги кўриниши. Математик мантиқ элементлари.

Алгоритм тушунчаси, унинг хоссалари ва тасвирлаш усуллари. Блок схема. Чизиқли алгоритмлар, тармоқланувчи алгоритмлар ва такрорланувчи алгоритмлар. Бирор синф масаласи ёки синфлар композицияси учун алгоритмлар яратиш.

С ва С++ тили синтаксислари. С++ ва бошқа тилларда (Pascal) модулли программалаш. Стандарт кутубхона. Оқим синфлари. Контейнер синфлари. Итераторлар и функционал объектлар. Алгоритмлар кутубхонаси. Сонли ҳисоблаш функцияларининг кутубхонаси.

8, 16, 32 разрядли процессорлар. Узилишлар назоратчиси. Умумий соҳа орқали параметрларни узатиш. BIOS ва операцион тизим узилишлари. 32 разрядли процессорларда қўшимча сегмент регистрларидан фойдаланиш. Турли муҳитларда яратилган программа объектларини боғлаш. LDT ва GDT жадваллари. Ҳимояланган режимда адреслаш. Assemblerда 32-разрядли иловасини яратиш.

С++ синфни ва объектларни тавсифлаш. С++ ва Delphi синф эълонининг фарқли томонлари. Тўпламли ворислик. Синфлар қолиплари, уларни яратиш ва ишлатиш. Фавқулотда ҳолат синтаксиси, фавқулотда ҳолатни илиб олиш. Бир турни иккинчисига келтириш. Стандарт оқимлар. Берилганларни форматлаш. Сатр функциялари. Контейнерлар. Кетма-кет ва ассоциатив контейнерлар. Стандарт алгоритмлар.

Консол ва диалог иловаларини яратиш. Иловалар, хужжатлар ва тасвир синфлари. Хужжатлар қолипи. Диалог дарчалар ва оддий бошқариш элементлари. Рўйхатлар синфи. Хабарлар ва буйруқлар. Хабарларни қайта ишлаш, хабарлар харитаси, WIN32 тизими хабарлар харитасини ишлатиш. График интерфейс билан ишлаш. Графика чизиш. Содда матн таҳririни яратиш. NET технологияси, С# тилини хусусиятлари.

Илова: Келтирилган мавзуларни камида 70-80%ини талабалар ўзлаштиришлари шарт.

Дастурнинг информацион - услубий таъминоти

Фанни ўқитиш жараёнида мавжуд нашр қилинган ўқув қўлланмалари ва электрон манбалар, Интернет тизимидаги мос таълим сайтлари маълумотларидан, хусусан http://www.intuit.ru, http://www.book.ru, http://www.zionet.uz, ва шунга ўхшаш сайтларидан фойдаланилади. Компьютер техникасини қўллаш билан боғлиқ замонавий педагогик ва информацион технологиялар асосланган ўқитиш методлари қўлланилади.

Фойдаланиладиган асосий дарсликлар ва ўқув қўлланмалар рўйхати

Асосий дарсликлар ва ўқув қўлланмалар

1. Б. Страуструп. Язык программирования С++. Специальное издание.-М.:ООО «Бином-Пресс», 2006.-1104 с.
2. Павловская Т.А. С++. Программирование на языке высокого уровня – СПб.: Питер. 2005.- 461 с.
3. Подбельский В.В. Язык СИ++.- М.; Финансы и статистика- 2003 562с.
4. Павловская Т.С. Щупак Ю.С. С/С++. Структурное программирование. Практикум.-СПб.: Питер,2002-240с
5. Павловская Т.С. Щупак Ю.С. С++. Объектно- ориентированное программирование. Практикум.-СПб.: Питер,2005-265с
6. Глушаков С.В., Коваль А.В., Смирнов С.В. Язык программирования С++: Учебный курс.- Харьков: Фолио; М.: ООО «Издательство АСТ», 2001.-500с.
7. Юров В., Хорошенко С. Assembler: Учебный курс- СПб, “Питер”,2000.-672с.

Қўшимча адабиётлар

1. Финогенов К.Г. Основы языка Assemblera.-М.: Радио и связь, 2001. - 288 с.
2. Пильшиков В.Н. Упражнения по языку Паскаль-М.: МГУ, 1986.
3. Абель П. Assembler для IBM PC и программирования. 1991. М.: “Высшая школа”, 1992.- 447 с.
4. Скенлон Л. Персональный ЭВМ IBM PC и XT. Программирование на языке Assemblera. -М.: Радио и связь. 1991.- 336 с.
5. Гофман В. Э., Хомоненко А.Д. Delphi 5. - СПб.: БХВ-Санкт-Петербург, 2000. -800с.
6. Немнюгин С.А. Turbo pascal, учебник. Изд. Питер., 2001, -496 с.
7. Поляков Д.Б., Круглов И.Ю. Программирование в среде Турбо-паскаль. (версия 5.5).М.:МАИ,1992.-576с.
8. Абрамов С.А.,Гнезделова Капустина Е.Н.и др. Задачи по программированию. - М.: Наука, 1988.
9. Вирт Н. Алгоритмы + структуры данных = программа.-М.:Мир,1985.-405с.
10. Информатика. Базовой курс. Учебник для Вузов., Санк-Петербург, 2001. под редакцией С.В.Симоновича.
11. Нортон П. Программно-аппаратная организация IBM PC.-М.:Мир,1991.-327с.
12. Фигурнов В.Э. IBM PC для пользователя. М.: Финансы и статистика. Юнити. 1997.
13. Юров В. Assembler: практикум. -СПб.: Питер, 2002.- 400с.
14. Informatika va programmalsh.O’quv qo’llanma. Mualliflar: A.A.Xaldjigitov, Sh.F.Madraximov, U.E.Adamboev, O’zMU, 2005 yil, 145 bet.

“ТАСДИҚЛАЙМАН”
Механика-математика
факультети декани
_____ Б.А. Шоимқулов

2010 йил “28 ” август

ПРОГРАММАЛАШ АСОСЛАРИ
фани бўйича

5480100 – Тадбиқий математика ва информатика
йўналиши 1-курси учун

ИШЧИ ЎҚУВ ДАСТУР

Умумий ўқув соати – 228с.

Шу жумладан:
Маъруза – 62с.
Амалий машғулот – 166с.

Мустақил таълим соати – 212с.

Фаннинг ишчи ўқув дастури М.Улуғбек номидаги Ўзбекистон Миллий Университети Механика-математика факультети Программалаш ва тармоқ технологиялари кафедрасининг 2010 йил “ 27 ” августдаги 1 – сонли мажлисида муҳокама этилди ва маъқулланди.

Ишчи дастур бакалаврият йўналишининг намунавий ўқув дастури ва ўқув режасига мувофиқ ишлаб чиқилди.

Тузувчилар: ф.-м.ф.н., доц.в.б. А**.М. Икрамов** _____ (имзо)

ф.-м.ф.н., доц. С**.М. Гайназаров** _____ (имзо)

Тақризчи: ф-м.ф.н., доцент **А.Т. Хайдаров** _____ (имзо)

Кафедра мудири: ф.-м.ф.н., доцент Ш**.Ф. Мадрахимов** _____ (имзо)

Фаннинг ишчи ўқув дастури Механика-математика факультети Илмий кенгашининг 2010 йил “28 ” августдаги 1 – сонли қарори билан тасдиқланди.

Илмий кенгаш раиси:
2010 йил “ 28 ” август _____ Б.А. Шоимқулов
(имзо)

ПРОГРАММАЛАШ АСОСЛАРИ

Кириш

Программалаш асослари фанининг бош мақсади талабаларга қўйилган масалани ечувчи компьютер программасини тузишга ўргатишдир. Шу мақсадда программалаш тиллари ва муҳитлари ҳақида умумий тушунчалар берилади ва бу тиллардан фойдаланишга ўргатилади.

Фан назарий ва амалий қисмлардан иборат. Назарий қисм информатика ва ҳисоблаш техникаси, алгоритмлар, C/C++ программалаш тили, Assembler тили, Visual C++ объектга йўналтирилган программалаш муҳити бобларидан ташкил топган.

Дастурда программалашга киришнинг назарий асоси бўлган алгоритмларга алоҳида эътибор қаратилган. Бу ерда алгоритмларни тавсифлаш ва кейинчалик компьютерда амалга ошириш учун зарур бўлган бир қатор математик тушунчалар - такрорлаш, ёрдамчи алгоритм, рекурсия, хотира, массив, индекс, параметр ва ҳ.к. киритилиб, турли хил синф масалаларининг алгоритмлари тузилади.

Фанда C++ тилининг алфавити, тил қурилмаларининг умумий синтаксиси, берилганлар турлари, операциялар ва функциялар қаралади, оқим билан ишлаш, кўрсатгичлар, амалларни қайта юклашлар, функцияларни қайта юклаш, қолиплар ва модулли программмалаш ва графика модули билан ишлаш ўргатилади.

Фаннинг мазмуни

Маъруза машғулотлари (62 с.)

Ахборот ва ахборот технологиялари. Информация турлари: узлуксиз ва дискрет информация. Информатика. ЭҲМни ривожланиш тарихи ва қўллаш соҳалари. ЭҲМ авлодлари, ЭҲМнинг асосий қурилмалари. ЭҲМда масала ечиш босқичлари. Саноқ системалари. Маълумотларнинг ЭҲМ хотирасидаги кўриниши. Математик мантиқ элементлари.

Алгоритм тушунчаси, унинг хоссалари ва тасвирлаш усуллари. Блок схема. Чизиқли алгоритмлар, тармоқланувчи алгоритмлар ва такрорланувчи алгоритмлар. Бирор синф масаласи ёки синфлар композицияси учун алгоритмлар яратиш.

C++ тили синтаксиси. Берилганлар турлари. Ўзгарувчилар ва ифодалар. Операторлар - “ифода”, тармоқланувчи, такрорлаш ва бошқарувни узатиш операторлари. Кўрсатгичлар ва массивлар. Фойдаланувчи томонидан аниқланадиган турлар. Модулли программалаш. Функциялар эълон қилиш ва аниқлаш. Локал ва глобал параметрлар. Рекурсив функциялар. Фунцияларни қайта юклаш. Функция қолиплари. Main() функцияси. Стандарт кутубхона функциялари. Препроцессор директивалари. Идентификаторларнинг амал доираси. Стандарт оқимлар. Фавқулотда ҳолатлар.

Амалий машғулотлар (166с.)

Саноқ системалари. Бир саноқ системасидан иккинчисига ўтиш. Берилганларни компьютер хотирасида тасвирланиши. Кодлаш.

Алгоритмни тасвирлаш усуллари. Блок схема. Чизиқли алгоритмлар, тармоқланувчи алгоритмлар ва такрорланувчи алгоритмлар. Бутун сонли арифметика масалалари. Ичма-ич жойлашган такрорланувчи жараенлар, итерацион жараёнлар. Берилганлар кетма-кетлигини сақлаш зарур масалалар. Кетма-кетликларни тартиблаш, оддий саралаш масалалари.

Тил таркиби: алфавит, калит сўзлар, ўзгармаслар, изоҳлар. Берилганлар турлари. Программа тузилиши. Ўзгарувчилар, амаллар, ифодалар. “Ифода” оператори. Тармоқланувчи оператор. Такрорлаш оператори. Такрорланувчи ҳисоблаш жараёнлари. Бошқарувни узатиш оператори. Кўрсатгичлар. Мурожаатлар. Бир ва икки ўлчамлаи массивлар. Турларни қайта номлаш, Санаб ўтилувчи тур. Тузилмалар. Бирлашмалар. Функцияларни эълони ва аниқлаш. Оддий функциялар. Функция параметрлари ва

қайтарувчи қиймати. Рекурсив функциялар. Стандарт функциялар. Препроцессор директивалари. Модулли программалаш. Берилганларнинг динамик тузилмалари: чизиқли рўйхатлар, стеклар, навбатлар ва бинар дарахтлар. Қисман тўлдирилган массивлар билан ишлаш. Хотирани бошқариш, файл структуралар билан ишлаш, саралаш ва қидириш алгоритмлари, ифодалар ҳисоблаш, интерпретатор яратиш, криптография.

Мустақил иш мавзулари (212с.)

Информация турлари: узлуксиз ва дискрет информация. ЭҲМ авлодлари, ЭҲМ нинг асосий қурилмалари. ЭҲМда масала ечиш босқичлари. Саноқ системалари. Маълумотларнинг ЭҲМ хотирасидаги кўриниши. Математик мантиқ элементлари.

Алгоритм тушунчаси, унинг хоссалари ва тасвирлаш усуллари. Блок схема. Чизиқли алгоритмлар, тармоқланувчи алгоритмлар ва такрорланувчи алгоритмлар. Бирор синф масаласи ёки синфлар композицияси учун алгоритмлар яратиш.

С ва С++ тили синтаксислари. С++ ва бошқа тилларда (Pascal) модулли программалаш. Стандарт кутубхона. Оқим синфлари. Контейнер синфлари. Итераторлар и функционал объектлар. Алгоритмлар кутубхонаси. Сонли ҳисоблаш функцияларининг кутубхонаси.

Фан мавзуларининг соатлар бўйича тақсимланиш жадвали
Маъруза машғулотлари (62 с.)

№	Фан мавзулари	Ажратилган соатлар
1	Информатика, Ахборот ва Ахборот технологиялари тушунчалари.	2
2	ЭҲМнинг ривожланиш тарихи, қўлланиш соҳалари, авлодлари, асосий қурилмалари	2
3	Саноқ системалари. Иккилик саноқ системаси компьютер амал қилиш тамоилининг асоси	4
4	Маълумотларни компьютер хотирасида тасвирланиши. Кодлаш.	2
5	Масалаларни ЭҲМдан фойдаланиб ечиш босқичлари. Ҳисоблаш эксперименти. Модел, алгоритм тушунчаси, унинг хоссалари ва тасвирлаш усуллари.	4
6	Алгоритмни тасвирлаш усуллари. Блок схема	2
7	Чизиқли алгоритмлар,тармоқланувчи алгоритмлар ва такрорланувчи алгоритмлар	4
8	Ичма-ич жойлашган такрорланувчи жараенлар, итерацион жараенлар.	2
9	Программалаш тиллари. Тил синтаксиси. Бекус - Науре шакли.	2
	Оралиқ назорат	
10	С++ тили синтаксиси.	2
11	Берилганлар турлари. Ўзгарувчилар ва ифодалар.	2
12	Операторлар - “ифода”, тармоқланувчи, такрорлаш ва бошқарувни узатиш операторлари.	2
13	Кўрсатгичлар	2
	1- Семестр якуний назорати	

14	Массивлар	4
15	Фойдаланувчи томонидан аниқланадиган турлар.	2
16	Модулли программалаш.	2
17	Функциялар эълон қилиш ва аниқлаш. Локал ва глобал параметрлар. Рекурсив функциялар.	4
18	Фунцияларни қайта юклаш. Функция қолиплари. Main() функцияси.	4
19	Стандарт кутубхона функциялари.	2
	Оралиқ назорат	
20	Препроцессор директивалари.	2
21	Идентификаторларнинг амал доираси.	2
22	Стандарт оқимлар.	4
23	Фавқулотда ҳолатлар.	4
	2-семестр якуний назорати	
Жами		62

Амалий машғулотлар

№	Фан мавзулари	Ажратилган соатлар
1	Саноқ системалари. Иккилик саноқ системаси компьютер амал қилиш тамоилининг асоси	8
2	Маълумотларни компьютер хотирасида тасвирланиши. Кодлаш.	6
3	Алгоритмни тасвирлаш усуллари. Блок схема	6
4	Чизиқли алгоритмлар,тармоқланувчи алгоритмлар ва такрорланувчи алгоритмлар	8
5	Ичма-ич жойлашган такрорланувчи жараенлар, итерацион жараенлар.	8
	1-жорий назорат	
6	Программалаш тиллари. Тил синтаксиси. Бекус - Науре шакли.	6
7	Тил таркиби: алфавит, калит сўзлар, ўзгармаслар, изоҳлар. Берилганлар турлари.	8
8	Программа тузилиши. Ўзгарувчилар, амаллар, ифодалар. “Ифода” оператори.	8
9	Тармоқланувчи оператор.	6
	2 - жорий назорат	
10	Такрорлаш оператори. Такрорланувчи ҳисоблаш жараёнлари.	8
11	Бошқарувни узатиш оператори.	6
12	Кўрсатгичлар. Мурожаатлар.	8
13	Бир ва икки ўлчамлаи массивлар.	8
	3-жорий назорат (1-семестр якуни)	

14	Турларни қайта номлаш, Санаб ўтилувчи тур.	6
15	Тузилмалар. Бирлашмалар.	6
16	Функциялар эълони ва аниқланиши. Оддий функциялар.	6
17	Функция параметрлари ва қайтарувчи қиймати. Рекурсив функциялар. Стандарт функциялар.	8
	1 - жорий назорат (2-семестр)	
18	Препроцессор директивалари	8
19	Модулли программалаш.	8
20	Берилганларнинг динамик тузилмалари: чизиқли рўйхатлар, стеклар, навбатлар ва бинар дарахтлар.	8
21	Қисман тўлдирилган массивлар билан ишлаш.	8
	2 - жорий назорат	
22	Хотирани бошқариш	8
23	Файл структуралари билан ишлаш	8
24	Берилганларни саралаш ва қидириш алгоритмлари	8
25	Графика масалалари	8
26	Модулли программалаш масалалари	8
	3 - жорий назорат (2-семестр якуни)	
Жами		**166**

Мустақил ўрганиш мавзулари ва уларнинг соатлар бўйича тақсимоти

№	Мавзулар	Соат ажратмаси
1	ЭҲМ авлодлари ва уларнинг синфларга ажратилиши	4
2	Программалаш тиллари ва уларнинг синфлари	6
3	Информация турлари: узлуксиз ва дискрет информация.	6
4	ЭҲМда масала ечиш босқичлари. Масаланинг математик моделлари	10
5	Саноқ системалари. Рақамлар ва саноқ системаларининг юзага келиш тарихи	14
6	Маълумотларнинг ЭҲМ хотирасидаги кўриниши. Математик мантиқ элементлари.	18
7	Алгоритм тушунчаси ва унинг асосий хусусиятлари. Бирор синф масаласи ёки синфлар композицияси учун алгоритмлар яратиш.	16
	Программалаш технологиялари	16
8	С ва С++ тиллари синтаксисларининг қиёсий таҳлили.	12
	С++ ва бошқа тилларда (Pascal) модулли программалаш.	14
9	Стандарт кутубхона.	12
10	Оқим синфлари.	16
11	Контейнер синфлари	16
12	Итераторлар и функционал объектлар.	16
13	Алгоритмлар кутубхонаси.	18
13	Сонли ҳисоблаш функцияларининг кутубхонаси	18
Жами		**212**

Асосий адабиётлар

8. Каримов И.А. Юксак малакали мутахассислар – тараққиёт омили. Т., Ўзбекистон, 1995 й.
9. Б. Страуструп. Язык программирования С++. Специальное издание.-М.: ООО «Бином-Пресс», 2006.-1104 с.
10. Павловская Т.А. С++. Программирование на языке высокого уровня – СПб.: Питер. 2005.- 461 с.
11. Подбельский В.В. Язык СИ++.- М.; Финансы и статистика- 2003 562с.
12. Глушаков С.В., Коваль А.В., Смирнов С.В. Язык программирование С++: Учебный курс//Харков: Фолио;М.: ООО «Издательство АСТ», 2001.- 500с.
13. Informatika va programmalsh.O'quv qo'llanma. Mualliflar: A.A.Xaldjigitov, Sh.F.Madraximov, U.E.Adamboev, O'zMU, 2005 yil, 145 bet.
14. Pascal tilida programmalash bo'yicha masalalar to'plami. O'quv qo'llanma. Mualliflar: A.A.Xaldjigitov, Sh.F.Madraximov, A.M.Ikromov, S.I.Rasulov, O'zMU, 2005 yil, 94 bet.

Қўшимча адабиётлар

15. Павловская Т.С. Щупак Ю.С. С/С++. Структурное программирование. Практикум.-СПб.: Питер,2002-240с
16. Павловская Т.С. Щупак Ю.С. С++. Объектно- ориентированное программирование. Практикум.-СПб.: Питер,2005-265с
17. Романов Б.А. Практикум по программированию на С++: Учебное пособие. СПб.: BXB-Петербург, Новосибирск: Из-во НГТУ, 2004.- 432с.
18. Смайли Джон. Учимся программировать на С++ вместе с Джоном Смайли. –СПб: ООО «ДиаСофтЮП», 2003.-560с.
19. Пильшиков В.Н. Упражнения по языку Паскаль-М.: МГУ, 1986.
20. А. Фридман и др. Архив программ на С/С++ - М. Бином 2001- 638 с.
21. Абрамов С.А., Гнезделова Капустина Е.Н. и др. Задачи по программированию. - М.: Наука, 1988.
22. Брябрин В.М. Программное обеспечение персональных ЭВМ. –М.: Наука.,1989.-272с.
23. Вирт Н. Алгоритмы + структуры данных = программа.-М.:Мир,1985.-405с.
24. Джордейн Р. Справочник программиста персональных компьютеров типа IBM PC, XT и AT. -М.: Финансы и статистика, 1992.-544с.
25. Информатика. Базовой курс. Учебник для Вузов., Санк-Петербург, 2001. под редакцией С.В.Симоновича.
26. Нортон П. Программно-аппаратная организация IBM PC.-М.:Мир,1991.-327с.

Маъруза матнлари аннотациялари

1. Информатика, ахборот ва ахборот технологиялари тушунчалари.
Информатика фани ҳақида тушунча. Ахборот турлари , ахборотни сақлаш, қайта ишлаш ва узатиш. Ахборот ўлчамлари. Ахборот технологиялари ҳақида тушунча.

2. ЭҲМнинг ривожланиш тарихи, қўлланиш соҳалари, авлодлари, асосий қурилмалари.
Илк ЭХМ машиналари. ЭХМ машиналари асосчилари. ЭХМ авлодлари. ва улар орасидаги фарқлар. ЭХМ асосий ва қушимча қурилмалари ва уларнинг вазифалари.

3. Саноқ системалари. Иккилик саноқ системаси. Компьютер амал қилиш тамоилнинг асоси.
Иккилик, саккизлик, ўнлиқ ва ўн олтилик саноқ системалари. саноқ системалари устида арифметик амаллар. Иккилик саноқ системасида арифметик амалларнинг компьютердаги ўзига хос вазифа ва хусусиятлари.

4. Маълумотларни компьютер хотирасида тасвирланиши. Кодлаш.
Компьютер хотирасини ташкил этувчи қурилмалар. Компьютер хотирасини тасвирлаш. Компьютер хотирасида кодлаш.

5. Масалаларни ЭХМдан фойдаланиб ечиш босқичлари. Ҳисоблаш эксперименти. Модель, алгоритм тушунчаси, унинг хоссалари ва тасвирлаш усуллари.
Масалаларни ЭХМда ечиш учун асосий вазифалар. Масалани ЭХМда ечиш учун босқичларга ажратиш.Ечиш усули, математик модель, алгоритм тушунчалари ва улардан масалани ечишда фойдаланиш. Алгоритм хоссалари ва тасвирлаш усуллари.

6. Алгоритмнинг тасвирлаш усуллари. Блок-схема.
Алгоритмнинг тасвирлаш усуллари. Блок-схема тушунчаси. Блок-схемани ташкил этувчи элементалри.

7. Чизиқли алгоритмлар, тармоқланувчи алгоритмлар ва такрорланувчи алгоритмлар.
Алгоритм турлари. Чизиқли, тармоқланувчи ва такрорланувчи алгоритмлар тушунчалари, вазифалари ва асосий фарқлари. Хар бир алгоритмга мисоллар.

8. Ичма-ич жойлашган такрорланувчи жараёнлар, итерацион жараёнлар.
Масалаларни ичма-ич жойлашган такрорланувчи жараёнлар ёрдамида ечиш. Масалаларни итерацион жараёнлар ёрдамида ечиш. Такрорланувчи ва итерацион жараёнларнинг фарқлари ва фойдаланиш соҳалари.

9. Программалаш тиллари. Тил синтаксиси. Бекус-Науре шакли.
Программалаш тиллари тарихи. Программалаш тиллари синтакисиси асоси. Бекус-Науре шакли ва ундан фойдаланиш.

10. С++тили синтаксиси.
С++тили синтаксиси. Асосий белгилари. Ёзиш кетма-кетликлари.

11. Берилганлар турлари. Ўзгарувчилар ва ифодалар.
Программалаш тилларида берилганлар тушунчаси. Компьютер хотираси ва берилганлар орасидаги алоқа. Ўзгарувчилар ва уларнинг турлари. Ифодалар ва уларни ёзиш кетма-кетлиги, амал қилиш соҳаси.

12. Операторлар – “ифода”, тармоқланувчи, такрорлаш ва бошқарувни узатиш операторлари.
Оператор тушунчаси. Оператор вазифасини бажарувчи ифодалар. Тармоқланувчи, такрорлаш ва бошқарувни узатиш операторлари ва улардан фойдаланиш.

13. Кўрсатгичлар.
Кўрсатгич тушунчаси. Кўрсатгичларнинг хотирадаги тақсимоти. Кўрсатгичлар устида амаллар.

14. Массивлар.

Массив тушунчаси. Вектор тушунчаси. Икки ўлчамли массивлар. Икки ўлчамли массивлар устида амаллар. Кўп ўлчамли массивлар. Кўп ўлчамли массивлар устида амаллар.

15. Фойдаланувчи томонидан аниқланадиган турлар.

Фойдаланувчи томонидан янги тур аниқлаш. Унинг ўзига хос хусусиятлари. Фойдаланувчи томонидан янги турларга мисоллар. Уларни ишлатиш соҳалари ва оддий турлардан кескин фарқли жихатлари.

16. Модулли программалаш.

Модуль тушунчаси. Модуль ясаш. Модуль ва асосий программа орасидаги муносабатлар. Модулли программалашнинг ўзига хос хусусиятлари.

17.Функциялар эълон қилиш ва аниқлаш.Локал ва глобал параметрлар. Рекурсив функциялар.

Функция тушунчаси. Программани функцияларга ажратиш. Функция эълон қилиш ва ишлатиш қоидалари. Локал ва глобал параметрлардан фойдаланиш ва уларнинг фарқи. Функциядан рекурсив функция ясаш. Рекурсив функциядан фойдаланиш соҳалари ва усуллари.

18. Функцияларни қайта юклаш. Функция қолиблари. Main () функцияси.

Функцияларни қайта юклаш тушунчаси ва улардан фойдаланиш. Қайта юклаш афзалликлари. Функция қолипларини ясаш ва фойдаланиш. Main () функцияси ва алоҳида белгилари.

19. Стандарт кутубхона функциялари.

Кутубхоналардан фойдаланиш тушунчаси. Стандарт кутубхона ўзгарувчилари, функциялари ва бошқа ташкил этувчилари ва улардан фойдаланиш.

20.Препроцессор директивалари.

Препроцессор тушунчаси. Препроцессордан фойдаланиш ва қўллаш соҳалари.

21. Идентификаторларнинг амал доираси.

Идентификатор тушунчаси ва унинг амал қилиш доираси.

22. Стандарт оқимлар

Оқим тушунчаси. Стандарт оқимлар ва улардан фойдаланиш. Оқимлар турлари.

23. Фавқулотда ҳолатлар.

Фавқулотда ҳолатлар келиб чиқадиган вазиятлар. Фавқулотда ҳолатларни бартараф этиш усуллари. Фавқулотда ҳолатларни бартараф этишнинг афзалликлари.

Маъруза матнлари

Маъруза 1
Информатика, Ахборот ва Ахборот технологиялари тушунчалари

Информация, ахборот ёки маълумот каби тушунчалар бизга кундалик хаётимизда таниш бўлишига карамасдан, информация тушунчасининг катъий таърифи мавжуд эмас. Бирор ўрганилаётган жонли ёки жонсиз обйект тўгрисидаги йигилган огзаки, ёзма (матн, жадвал, расм-чизма, схема) ёки бошка турли кўринишдаги маълумотлар (ахборотлар) ёки берилган тўпламни, биз одатда информация деб кабул киламиз. XX аср ўрталарига келиб, бу тушунча кенг маънода тушиниладиган «Информация» сўзига айланди. Информация сўзи билан планеталар бўйлаб жўнатиладиган сигналлардан тортиб, то ўсимлик ва хайвонот олами ва хаттоки, инсон организмининг энг кичик тузилмалари - генларида сакланадиган маълумотлар хам ифодаланади. Лекин, хар кандай маълумотлар тўплами обйект тўгрисида аник маълумот бермайди ва шу билан биргаликда йигилган маълумотни тахлил килиш учун махсус усуллар ва техник курилмалар зарур бўлиши хам мумкин.

Информация бу бизни ўраб турган моддий оламнинг обйектлари, вокеа-ходисалари, жараёнлар ва уларнинг ўзаро таъсири, ривожланиши ва хоказолар хакидаги маълумотлар тўпламининг инсон томонидан, унинг сезиш органлари ёки ёрдамчи техник воситалар ёрдамида англаниши, ўрганилиши натижасида хосил бўлган хулоса ва маълумотлардир. Информация одатда узлуксиз (аналог) ва ракамли (дискрет) кўринишларда бўлади. Узлуксиз информацияга мисол сифатида одам товушини, мусика асарини, ракамли сигналга эса, 0 ва 1 сонлар комбинациялари оркали ифодаланган узлукли маълумотларни келтириш мумкин.

Информатика - бу французча Informatique сўзи бўлиб, *Information* (Информация) ва *Avtomatique* (Автоматика) сўзларидан ташкил топган ва информация йигишни, кайта ишлашни, узатишни, компютер ёки бошка техник воситалар ёрдамида автоматик тарзда амалга оширишни ўрганишга багишланган фандир. Бу фан Гарбий Йевропа давлатлари ва Америкада «Сомрутер Ссиенсе», МДХ ва Шаркий Йевропа давлатларида эса «Информатика» номи билан юритилади. Бу ўринда электрон хисоблаш машиналари жуда катта хажмдаги информацияни кайта ишлашга имкон берадиган самарали куролдир.

Иккинчи мухим тушинчаси «Информацион технологиялар» тушунчасидир. Одатда, «технология» сўзи моддий ишлаб чикариш сохасига нисбатан ишлатилиб, бирор материални кайта ишлаш ёки предметни тайёрлаш жараёнининг махсус техник усулларини ифодалаш максадида ишлатилади. Информацион технологияларда кайта ишлаш учун «хомашё» сифатида «информация» каралади ва у компютер-программа ва кўшимча техник воситалар ёрдамида автоматик тарзда кайта ишланади.

Хозирги кунга келиб, «Информацион технологиялар» информатика фанининг ажралмас бир кисми бўлиб, у инсон фаолиятининг турли сохаларида учрайдиган информацияларни, аппарат-программа воситалари ва усуллари ёрдамида кайта ишлаш каби вазифаларни бажаришга мўлжалланган. Бу таърифдан кўриниб турибдики, информатика ва информацион технологиялар тушунчалари бир-бирига жуда якин.

Информацион технологияларнинг асосий аппарат воситаси электрон хисоблаш машинасидир. Дунё бозорида мавжуд турли-туман ЭХМ парклари орасида ИВМ (Интернатионал Бусинесс Мачине Сорпоратион) компютерлари етакчи ўрин тутади.

Хисоблаш техникасининг ривожланиш тарихига назар ташлайдиган бўлсак, унинг куйидаги бир неча мухим даврларни ўз ичига олишини кўриш мумкин:

1. ибтидоий хисоблаш воситалари ва хисоб чўтлар даври;

2. механик машиналар даври;
3. электро-механик машиналар даври;
4. электрон хисоблаш машиналари даври.

Механик машиналаргача бўлган давр. Хисоблаш ишларининг тарихи одамзод пайдо бўлишидан бошланади. Йер юзидаги энг биринчи хисоблаш воситаси сифатида ибтидоий одамлар томонидан кўл бармоклари фойдаланилган .

Кўл ва оёк бармоклари ибтидоий "хисоблаш воситаси" вазифасининг ўтаган. Бинобарин, ўша кадим замонлардаёк хисоблашнинг энг биринчи ва энг оддий усули-бармок хисоби пайдо бўлган. У кадимий кабилаларда хисобни 20 гача олиб боришни таъминлаган. Хисоблашнинг бу усулида бир кўл бармоклари "беш" ни, икки кўл бармоклари "ўн" ни, кўл ва оёк бармоклари биргаликда "йигирма" ни билдирган.

Дастлабки ва энг содда сунъий хисоб асбобларидан бири биркадир. Бирка 10 ёки 12 та таёкчадан иборат бўлиб, таёкчалар турли-туман шакллар билан ўйилган. Кишилар бирка ёрдамида подадаги моллар сонини, йигиб олинган хосил микдорини, карз ва хоказоларни хисоблашган.

Хисоблаш ишларининг мураккаблашуви эса янги хисоблаш асбоблари ва усулларини излашни такозо этарди. Ана шундай эхтиёж туфайли вужудга келган ва кўринишидан хозирги чўтни эслатувчи абак асбоби хисоблаш ишларини бирмунча осонлаштирди. Дастлабки хисоб асбобларидан яна бири ракамлар ёзилган бир канча таёкчалардан иборат бўлиб, шотландиялик математик Жон Непер номи билан аталган. Непер таёкчалари ёрдамида кўшиш, айириш ва кўпайтириш амаллари бажарилган. Кейинрок бу асбоб анча такомиллаштирилади ва нихоят логарифмик чизгич яратилишига асос бўлди.

Механик давр. Хисоблаш техникасида механик мосламалар даврини бошлаб берган машиналардан бири немис олими Вилгелм Шиккард томонидан 1623 йили ихтиро килинди. Бирок, бу хисоблаш машинаси жуда тор доирадаги кишиларгагина маълум бўлганлиги сабабли узок вактларгача бу борадаги биринчи ихтирочи 1645 йили арифмометр ясаган француз математиги Блез Паскал деб хисобланиб келинган. Лекин, 1958 йили Штутгарт шахри кутубхонасида И. Кеплернинг кўлёзма ва хужжатлари орасидан топилган хисоблаш машинаси чизмаси бу борадаги биринчи ихтирочи Шиккард эканлигини узил-кесил тасдиклади.

Маъруза 2
ЭҲМнинг ривожланиш тарихи, қўлланиш соҳалари, авлодлари, асосий қурилмалари

Тарихан киска давр ичида ЭХМнинг тўртта авлоди яратилди. ЭХМларни авлодларга ажратишларни яратишда нималарга асосланганлиги, кандай тузилганлиги, техник характеристикалари, фойдаланувчилар учун кулайлиги ва бошка томонлари асос килиб олинади.

ЭХМларнинг биринчи авлоди (1940-1950 йиллар) каторига Мустакил Давлатлар Хамдўстлиги давлатларининг олимлари яратган МЭСМ, БЭСМ-1, БЭСМ-2, МИНСК-1, УРАЛ-1, УРАЛ-2 ва бошкалар киради.

Бу машиналарнинг хаммаси электрон лампалар асосида курилган. Улар ўлчамларининг катталиги, электр кувватини кўп истеъмол килиши, амалларнинг бажарилиш тезлиги пастлиги, катта микдорда ахборотларни саклай олмаслиги ва ишончсизлиги билан ажралиб турарди. Бу тоифа машиналар секундига ўртача 10000 амал бажаради. Хотирасига факат 2047 тагача сўз сигади.

ЭХМларнинг иккинчи авлоди (1950-1965 йиллар) транзисторлар (ярим ўтказгич ва магнитли элементлар) тузилган бўлиб, бу авлодга мансуб машиналарнинг ўзига хос хусусиятларидан бири, улар кўлланиш сохаси бўйича ихтисослаштирилгандир. Иккинчи авлод ЭХМларида олдингиларига караганда маълумотларни кўпрок, тезрок ва ишончли кайта ишлаш имконияти яратилди.

ЭХМнинг иккинчи авлодига куйидаги машиналар киради: Минск-2, Раздан-3, М-220, БЭСМ-6, Мир, Найири, Минск-22, Минск-32, Урал-14 ва бошкалар. Бу машиналарда кўйилган масалаларни тез йечиш имкониятини яратидиган программадан, яъни масалани йечишда ЭХМ бажариши лозим бўлган амаллар кетма-кетлигидан фойдаланиш мумкин. Бундай ЭХМларнинг ўртача тезлиги 100000 амал-секунд, хотирасига 10000 тагача сўз сигади.

Электрон хисоблаш машиналарининг кейинги мукаммаллашуви турли вазифаларни бажарувчи мосламаларнинг яратилишига олиб келди, бу эса ўз навбатида, элемент ва схемаларнинг ўлчамларини кичрайтиришни ва уларнинг ишлашдаги ишончлигини оширишни, хотира сигимини катталаштиришни, ишлаш тезлигини яна хам тезлатишни талаб этди. Шунга асосан, микроэлектроникада тез орада 1 куб.см хажмли кристаллида энг камида 5 дона электроника элементини бирлаштирган электрон курилма, яъни митти интеграл схемалар пайдо бўла бошлади. Бундай схемалар иккинчи авлод машиналарида мавжуд бўлган барча камчиликларнинг анча кисмини бартараф этишга ва янги хисоблаш машиналарининг пайдо бўлишига замин яратди. Интеграл схема, аввало ясалаётган мосламаларни жуда хам кичиклашишига олиб келди.

ЭХМларнинг учинчи авлоди (1965-1975 йиллар) транзисторлар ва турли хил эхтиёт кисмлар ўрнига интеграл схемалардан кенг кўламда фойдаланиши билан характерланади.

Интеграл схемалардан фойдаланиш туфайли машиналарнинг техник ва ишлатиш характеристикаларини анча яхшилашга, жумладан, ихчамлашувига ва ишлаш тезлигининг ошишига эришилди. Бундай машиналарнинг ишлаши анча самарали ва ишончли бўлди. Уларнинг хотира сигими 2048 Кбайтгача кенгайди. Бу авлод машиналарини бир нечта мамлакатлар биргаликда ишлаб чикарганлиги учун уларни Ягона Система (ЭС-единая система) туридаги машиналар деб номланди, уларнинг номлари «ЭС» кискартмасидан бошланади: ЭС-1050, ЭС-1022 ва бошкалар.

Бу машиналар турига караб, секундига 2 миллионгача турли арифметик амалларни бажариши мумкин бўлди.

Фан ва техниканинг ривожланиши одам билан ЭХМ ўртасида мулокот килиш мумкин бўлган хисоблаш машиналари яратиш заруратини тугдирди. Бу имконият янги пайдо бўлаётган тўртинчи авлод машиналарида амалга оширилди.

ЭХМларнинг тўртинчи авлоди (1975 йилдан бошлаб). Уларда элемент базаси сифатида катта интеграл схемалар, яъни 1см. куб хажмда 100 мингтагача элементни бирлаштирган микросхема кўлланилади.

Хозирги кунда бешинчи авлод машиналарини ишлаб чикиш устида катта ишлар килиняпти. Айникса, бу сохада XX асрнинг 80-йилларида Япония олимлари таклиф этган бешинчи авлоднинг лойихаси диккатга сазовордир. Бу лойиха кейинги давр машиналарини яратишни кўзда тутган. Япония олимларининг таъбирича, ушбу авлод машиналари мантикий масалаларни хал кила оладиган, огзаки гапларни "ешитадиган" ва "тушинадиган", матнларни ўкиётган тезликда таржима кила оладиган хамда "кўра" оладиган, "тушунадиган" бўлиши керак.

Бундай компютерлар катта ва ўта катта интеграл схемалар асосида курилиши назарда тутилган. Охирги пайтларда ривожланган мамлакатлардаги илмий лабораторияларда оксил молекулалари билан тажриба ўтказилмокда. Улар компютерларнинг арифметик асосини ташкил килувчи асосий элемент иккилик санок системасида хотирловчи катакчалар вазифаларини ўтаяпти. Албатта, ушбу ёъналиш бўйича ЭХМ куриш хакида гап юритишга эрта албатта, лекин тажрибалар яхши натижаларга олиб келса, компютерларнинг янги даврини бошлайдиган биокомпютерларга эга бўлишимиз хам мумкин. Хисоблаш техникаси воситаларининг тузилиши ва уларни ишлаб чикаришнинг такомиллашуви билан электрон хисоблаш машиналарининг янгилари пайдо бўлаверади. Хозирда ЭХМларни куйидаги турларга ажратиш мумкин: микро, шахсий, мини, ўртача тезликда ишлайдиган, катта тезликда ишлайдиган супер ЭХМлар. Хозирги кунда ЭХМлардан физика, математика, астрономия, геофизика, техника ва бошка бир

талай фан сохаларида турли хил мураккаб амалий масалаларни йечишда муваффакиятли фойдаланилмокда, жумладан, атом энергетикаси, гидротехника иншоотларини куриш, кемасозлик, космик фазони забт этиш ва бошка шу каби кўплаб сохаларнинг бекиёс даражада тез ривожланиб кетиши, шак-шубхасиз хисоблаш техникасининг кенг кўламда самарали кўлланилаётганлиги натижасидир. Хозирги кунда ЭХМлар кўлланилмаётган бирор сохани топиш кийин, якин келажакда унинг яна хам кенгрок ривожланиши ва инсон фаолиятига чукуррок кириб бориши кутилмокда.

Маъруза 3

Санoқ системалари ва уларда арифметик амаллар. Иккилик санoқ системаси компьютер амал қилиш тамоилининг асоси

ЭХМ - бу электрон ракамли курилмадир. Электрон курилма дейилишига сабаб хар кандай маълумотлар ЭХМ да электр сигналлари оркали кайта ишланади. Ракамли дейилишига сабаб ЭХМ да хар кандай маълумот сонлар ёрдамида тасвирланади.

Сонларни ёзиш усулига санок системаси деб аталади. Сонларни ёзиш учун хар бир санок системасида ўзига хос турли белгилар тўпламидан фойдаланилади. Фойдаланилган тўпламдаги белгилар уларнинг сони, санок системасини характерловчи асосий катталиклардир. Санок системасида фойдаланиладиган белгилар сони санок системасининг асосини ташкил этади. Берилган санок системасида сонларни ёзишдаги фойдаланилган белгилар сонига караб, ўнлик, иккилик, саккизлик, ўн олтилик ва бошка санок системаларни киритиш мумкин. Шу билан бирга санок системаларини *pozision* ва *nopozision* турларга ажратиш мумкин. Позицион санок системасида берилган соннинг киймати сонни тасвирловчи ракамларнинг эгаллаган ўрнига боглик бўлади. Мисол сифатида, 0,1,2,3,. . . ,9 араб ракамларидан ташкил топган ўнлик санок системани караш мумкин. Нопозицион санок системаларида, белгининг киймати унинг эгаллаган ўрнига боглик эмас. Мисол сифатида рим ракамлари санок системасини келтириш мумкин. Масалан, XX сонида X раками, кайерда жойлашганига карамасдан ўнлик санок системасидаги 10 кийматини англатади.

Куйидаги жадвалда ўнлик санок системасида берилган 1 дан 16 гача сонларнинг иккилик, саккизлик ва ўн олтилик санок системаларидаги кўриниши келтирилган.

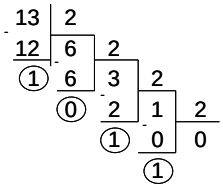
Бу жадвал бўйича бир санок систамасидан иккинчисига ўтиш масаласини кўриб ўтайлик. Масалан: 10 лик санок системасидаги 13 сонига 8 лик санок системасида 15 сони мос келади ва у 13 ни 8 га бўлинганда хосил бўлган бутун сон 1 ва колдик 5 лардан ташкил топган. Худди шунингдек 13 ни 6 га бўлганда хосил бўлувчи бутун сон 2 ва колдик 1 лар 21 сонини хосил килади. Бу сон 13 сонининг 6 лик санок системасидаги кийматидир.

Одатда бирор X сонининг кайси санок системасига тегишлилигини кўрсатиш учун унинг пастида индекс сифатида зарур санок системасининг асоси кўрсатилади.

SANOQ SISTEMALARI							
2	**3**	**4**	**5**	**6**	**8**	**10**	**16**
0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1
10	2	2	2	2	2	2	2
11	10	3	3	3	3	3	3
100	11	10	4	4	4	4	4
101	12	11	10	5	5	5	5
110	20	12	11	10	6	6	6
111	21	13	12	11	7	7	7
1000	22	20	13	12	10	8	8
1001	100	21	14	13	11	9	9
1010	101	22	20	14	12	10	A
1011	102	23	21	15	13	11	B
1100	110	30	22	20	14	12	C
1101	111	31	23	21	15	13	Д
1110	112	32	24	22	16	14	E
1111	120	33	30	23	17	15	F
10000	121	100	31	24	20	16	10

Масалан, X_6 – X сўннинг 6 лик саноқ системасига тегишли эканлигини кўрсатади.

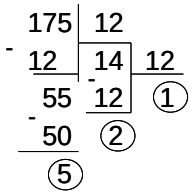
$X_{10}=13$ соннинг X_2-иккилик саноқ системасидаги кўринишини топайлик. Юқоридагидек, 13 ни кетма-кет 2 га бўламиз ва бўлишни то бутун қисмида нол ҳосил бўлгунча давом эттирамиз.



Ўнгдан чапга тартибида ёзилган колдиклар, яъни 1101 сони $X_{10}=13_{10}$ сонининг иккилик санок системасидаги кўриниши бўлади.

Энди 8 лик санок системасидан 10 лик санок системасига бўлиш ёъли билан ўтишга доир мисоллар кўрайлик. Масалан, жадвал бўйича 15_8 га 13_{10} мос келади. Энди уни топиб курайлик, бунинг учун 15_8 ни 10 лик санок системасининг асоси-10 нинг 8 лик санок системасидаги кўриниш - 12 га бўлиш керак бўлади. 15_8 ни 12_8 га бўлса бутун кисмида 1 ва колдикда 3, яъни 13_{10} - хосил бўлади. Бунга жадвал оркали ишонч хосил килиш хам мумкин.

Иккинчи мисол: 175_8 сонини 10 лик санок системасидаги кўринишини топиш талаб килинган бўлсин. Худди юкоридагидек 175_8 ни 128 га кетма-кет бўламиз. Эслатиб ўтамиз, бўлиш амали 8 сонлик санок системасида олиб борилади. (Жадвалга каралсин)



R санок системасида берилган сонни Q санок системасига ўтказиш учун, R санок системасидаги X сони Q санок системасининг асосига, яъни Q га кетма-кет, то бутун кисмида 0 хосил бўлгунча давом эттириш керак. Колдиклар ўнгдан чапга караб кетма-кет ёзилса, R санок системасида берилган X_r сонининг Q санок системасидаги X_q кўриниши хосил бўлади. Бўлиш амали берилган R санок системасида амалга оширилади.

Баъзи бир санок системаларидан иккинчисига кулайрок, осонрок холда ўтиш имкониятлари мавжуд. Хусусий холда, 2 га каррали сонларнинг биридан 2 иккинчисига ўтиш коидасини кўриб ўтамиз.

Масалан, 8 лик санок системасида берилган $X_8 = 5361$ сонидан X_2 га бўлиш учун, X_8 нинг хар бир ракамини 2 ликдаги кўриниши-триадалар ($2^3 = 8$) билан алмаштириб чикамиз:

$$X_2 = \underline{101}\ \underline{011}\ \underline{110}\ \underline{001}$$
$$5\quad 3\quad 6\quad 1$$

$D8A2_{16}$ ни 2 лик саноқ системасига ўтказиш учун унинг ҳар бир рақамини 2 лик саноқ системасидаги тўртликлар- тетрадалар билан алмаштирамиз:

$$X_2 = \underline{1101}\ \underline{1000}\ \underline{1010}\ \underline{0010}$$
$$D\quad 8\quad A\quad 2$$

Маъруза 4
Маълумотларни компьютер хотирасида тасвирланиши. Кодлаш

Масалаларни ЭҲМдан фойдаланиб ечиш босқичлари. Ҳисоблаш эксперименти. Модел, алгоритм тушунчаси, унинг хоссалари ва тасвирлаш усуллари

ЭХМ хотирасидаги барча бошлангич маълумотлар кодлаштирилган холда 0 ва 1 кўринишида бўлади. Хотирадаги хамма ишчи программалар ва буйруклар хотирада бир нечта байтларда жойлаштирилган бўлади. Хар бир буйрук - кўрсатма, махсус кодга эга бўлиб, компютерга у ёки бу амални бажариш кераклигини билдиради. Буйруклар сифатида - икки сон устида бирор арифметик амал, дискдан маълумотни ўкиш, экранга белгини узатиш, принтерда белгини чоп килиш каби кўрсатмалар бўлиши мумкин. Масалан, йигиндини хисоблаш учун куйидаги форматдаги буйрук коди бўлиши мумкин.

Буйруқ коди (+ amali)	1-operand адреси (s)	2-operand адреси (d)
010110	111010101001	110001100101

Бу ерда 010110 кўшиш «+» амалининг коди, 111010101001 *s* ўзгарувчи учун хотирада ажратилган жойнинг адреси, 110001100101 эса д ўзгарувчи учун кийматлар устида кўшиш амалини бажариб, натижани хотира регистрларининг бирига жойлаштиради, кўп холларда бу сумматор деб номланувчи А регистри бўлади.

Программага кирувчи бундай буйруклар кетма-кетлиги хотиранинг маълум бир кисмида жойлашади. Программа бўйича ЭХМ нинг иши Бошкарув курилмасини (BQ) Буйрук адреси санагичи (BAS) деб номланувчи регистрга «карашдан» бошланади. да программадаги биринчи буйруknинг хотирадаги адреси бўлади. Шу буйрукни бажарилиши билан ЭХМ иш бошлайди. Бу буйрук бажарилиш пайтида BAS да навбатдаги буйрук адреси пайдо бўлади, кейин процессор шу буйрукни бажаради, BAS да навбатдаги буйрук адреси пайдо бўлади ва хоказо. Бу жараён BAS да «программа бўйича ишлаш тугаши» буйругининг адреси пайдо бўлиб, шундан кейин ЭХМ ўз ишини тугатади.

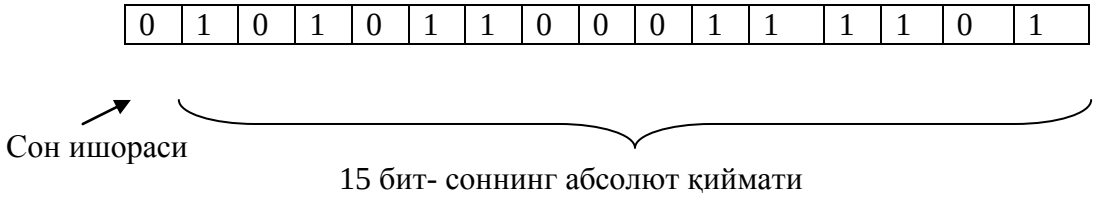
BQ кўрсатилган адрес бўйича хотирада маълумотни ўкийди ва арифметик-мантикий курилма (AMQ)ни бу амални бажаришга созлайди. BQ кийматлар операндларнинг буйрукдаги адреслари оркали хотирадан топади. Бу ишлардан кейин BQ «дам олади», яъни AMQ га кўрсатилган амални бажаришга имкон беради. AMQ бажарилган амал натижасини ўзининг чикишларида номоён килади. Ўз ишини тугатган AMQ, бу хакида BQ га сигнал оркали маълум килади. BQ натижани кўрсатилган хотира

адресига ёки транслатор томонидан кўзда тутилган жойга (регистрга) жойлаштиради. Шундан кейин BQ BAS дан навбатдаги буйрукни ўкийди ва юкоридаги жараён такрорланади. Шундай бўлиши мумкинки, кейинги буйрук маълумотни бирор-бир ташки курилмага (екранга, когозга, дискка) чикариш буйруги бўлиши мумкин. Бу холда BQ мос курилмага мурожаат килади, уни ишга тайёрлайди ва уни ишга туширади. Маълумотни чикариш тугагандан кейин чикариш курилмаси BQ га бу хакида сигнал беради. Худди шундай, кейинги буйрук маълумотни киритиш бўлса, бошкарув клавиатура, дискдан ўкиш ва бошка курилмаларга берилади ва улар ўз ишини тугатади. Кейин BQ BAS дан навбатдаги буйрукни олади ва уни бажаришга ўтади.

Шундай килиб, ЭХМ программага риоя килинган холда машинанинг барча курилмаларини мувофик ишлашини таъминлайди.

Сонларнинг ЭХМ хотирасида саклаш учун иккилик санок системасидан фойдаланиш жуда хам кулай. Бу санок системасини кўллаш, икки тургун холатга ўтиш имконини беради. Бу холатлардан бири соннинг разряди 1 учун бошкаси эса 0 учун хизмат килади. Бундай элементлар тузилиш жихатдан содда ва ишончлидир. Худди шундай соннинг ишоралари учун хам фойдаланиш мумкин. Одатда элементнинг 0 га мос келувчи холати, ишора регистрида Q ишораси учун, бирга мос келувчи холати эса, - ишораси учун фойдаланилади.

Икки байтлик ишорали бутун соннинг ЭХМ хотирасидаги тасвирланиши:



Тўрт байтлик хақиқий соннинг тасвирланиш:

Ишора	**Соннинг тартиби**	**Соннинг мантиссаси**
1 бит	6-бит	25-бит

Санок системаларига боглик бўлмаган холда, сонлари фиксирланган ва сузувчи вергулли деб номланувчи икки кўринишларда ифодалаш мумкин. Фиксирланган вергулли сонлардан фойдаланувчи машиналарда, ишора разрядларидан бошка хамма регистр разрядлари (ёки ячейкалар) сонларнинг разрядларини ифодалаш учун хизмат килади ва регистрнинг хар бир разрядига, соннинг аник ва хар доим битта разряди мос келади. Бу холат арифметик амаллар бажаришни соддалаштиради, лекин машинада ишлатиладиган сонлар диапазонини жуда чеклаб кўяди. Одатда бу диапазон $-1<x<1$ бўлади. Бу эса машинага ёзилган сонларни ўкишда вергулни соннинг энг катта разряди олдига кўйиш керак деган келишувга мос келади. Бу диапазонга тушмайдиган сонлардан фойдаланиш учун, программа тузувчи масштабли кўпайтувчиларни кўллаши керак бўлади. Сонларнинг фиксирланган вергулли кўринишда ёзишнинг бошка бир камчилиги, абсолут киймати жихатидан кичик бўлган сонларни ифодалашда кийматли ракамлар соннинг анча камлигида, яни бирга якин сонларга караганда нисбий аниклигининг камлигида кўринади.

Бу камчиликлардан кутулиш учун замонавий ЭХМ ларда сонларни сузувчи вергулли кўринишда ёзиш мумкин. Бу холда разрядларнинг бир кисми соннинг тартиби учун, колган кисми эса соннинг мантиссаси учун ажратилади. Хар икки кисмлари биттадан разряд сон тартибининг ва мантиссасининг ишораси учун ажратилади.

Одатда сузувчи вергулли машиналарда сонларнинг нормаллашмаган ёзувчидан хам фойдаланиш мумкин, лекин ундан фойдаланмаган макул, чунки нормаллашмаган сонлар устида бази бир амаллар нотўгри бажарилади ёки умумий бажариб бўлмайди. Замонавий шахсий компютерларда сонларни ёзиш учун хар хил узунликдаги ячейкалар (сўзлар) хам да кодлашнинг (ишорали ва ишорасиз) турли усуллари фойдаланилади.

Белгили ва сатр ўзгарувчиларини кодлаш учун АССИИ жадвали фойдаланилади. Бу жадвалда 256 та белгини хар бирига [0,255] ораликдаги бир бутун сон мос келади. Мантикий кийматлар одатда битта разрядга бир ёки нолни ёзиш оркали аникланади.

Маъруза 5
Алгоритмни тасвирлаш усуллари. Блок схема. Чизиқли, тармоқланувчи ва такрорланувчи алгоритмлар

Юкорида кайд килганимиздек, кўйилган бирор масалани ЭХМда йечиш учун, аввал унинг математик моделини, кейин алгоритмини ва программасини тузиш керак бўлади. Бу учликда алгоритм блоки мухим ахамиятга эга. Энди алгоритм тушунчасининг таърифи ва хоссаларини баён киламиз.

Алгоритм бу олдимизга кўйилган масалани йечиш зарур бўлган амаллар кетма-кетлигидир.

Масалан квадрат тенгламани йечиш учун куйидаги амаллар кетма-кетлиги зарур бўлади:

1. *a,v,s*- коэффициентлар берилган бўлсин,
2. берилган *a,v,s*- коэффициентлар ёрдамида дискриминант $D=b^2-4ac$ хисобланади,
3. $D>0$ бўлса $X_{1/2}=\left(-b\neq\sqrt{D}\right)/(2*a)$
4. $D<0$ бўлса хакикий ечими йўқ

Мисол сифатида яна берилган *a, v, s* томонлари бўйича учбурчакнинг юзасини Герон формуласи бўйича хисоблаш масаласини кўриб ўтайлик.

1. *a, v, s* -учбурчакнинг томонлари узунликлари,
2. $r=(a+v+s)/2$ – периметрнинг ярми хисоблансин,
3. $T=p(r-a)(r-v)(r-s)$ хисоблансин,
4. $S=\sqrt{T}$ хисоблансин.

Юкоридаги мисоллардан кўриниб турибдики, алгоритмнинг хар бир кадамда бажариладиган амаллар тушинарли ва аник тарзда ифодаланган, хамда чекли сондаги амаллардан кейин аник натижани олиш мумкин.

Зикр этилган, тушинарлилик, аниклик, чеклилик ва натижавийлик тушунчалари алгоритмнинг асосий хоссаларини ташкил этади. Бу тушунчалар кейинги параграфларда алохида кўриб ўтилади.

Алгоритм сўзи ва тушунчаси IX асрда яшаб ижод этган буюк аллома Мухаммад ал-Хоразмий номи билан узвий боглик. Алгоритм сўзи Ал-Хоразмий номини Европа олимлари томонидан бузиб талаффуз килинишидан юзага келган. Ал-Хоразмий биринчи бўлиб ўнлик санок системасининг тамойилларини ва ундаги тўртта амалларни бажариш коидаларини асослаб берган.

Мустакил бажариш учун топшириклар:

1. $Y=a(b+cx)-dx$ формула бўйича киймат хисоблаш алгоритми тузилсин.

2. Бир тўгри чизикда ётмайдиган учта нукта (A, B, C) оркали ўтувчи айланани ясаш алгоритми тузилсин.

3. "Светофордан (уч чирокли) фойдаланиш" алгоритми тузилсин

4. $Y=ax+b$ кўринишдаги функсиянинг ($a\neq0$) графигини чизиш алгоритми тузилсин.

Алгоритмнинг асосий хоссалари

Алгоритмнинг 5-та асосий хоссаси бор.

1.Дискретлилик (Чеклилик). Бу хоссанинг мазмуни алгоритмларни доимо чекли кадамлардан иборат килиб бўлаклаш имконияти мавжудлигида. Яъни уни чекли сондаги оддий кўрсатмалар кетма-кетлиги шаклида ифодалаш мумкин. Агар кузатилаётган жараённи чекли кадамлардан иборат килиб кўллай олмасак, уни алгоритм деб бўлмайди.

2.Тушунарлилик. Биз кундалик хаётимизда берилган алгоритмлар билан ишлаётган электрон соатлар, машиналар, дастгохлар, компютерлар, турли автоматик ва механик курилмаларни кузатамиз.

Ижрочига тавсия этилаётган кўрсатмалар, унинг учун тушинарли мазмунда бўлиши шарт, акс холда ижрочи оддийгина амални хам бажара олмайди. Ундан ташкари, ижрочи хар кандай амални бажара олмаслиги хам мумкин.

Хар бир ижрочининг бажариши мумкин бўлган кўрсатмалар ёки буйруклар мажмуаси мавжуд, у ижрочининг кўрсатмалар тизими (системаси) дейилади. Демак, ижрочи учун берилаётган хар бир кўрсатма ижрочининг кўрсатмалар тизимига мансуб бўлиши лозим.

Кўрсатмаларни ижрочининг кўрсатмалар тизимига тегишли бўладиган килиб ифодалай билишимиз мухим ахамиятга эга. Масалан, куйи синфнинг аълочи ўкувчиси "сон квадратга оширилсин" деган кўрсатмани тушинмаслиги натижасида бажара олмайди, лекин "сон ўзини ўзига кўпайтирилсин" шаклидаги кўрсатмани бемалол бажаради, чунки у кўрсатма мазмунидан кўпайтириш амалини бажариш кераклигини англайди.

3. Аниклик. Ижрочига берилаётган кўрсатмалар аник мазмунда бўлиши зарур. Чунки кўрсатмадаги ноаникликлар мўлжалдаги максадга эришишга олиб келмайди. Одам учун тушинарли бўлган "3-4 марта силкитилсин", "5-10 дакика киздирилсин", "1-2 кошик солинсин", "тенгламалардан бири йечилсин" каби ноаник кўрсатмалар робот ёки компютерни кийин ахволга солиб кўяди.

Бундан ташкари, кўрсатмаларнинг кайси кетма-кетликда бажарилиши хам мухим ахамиятга эга. Демак, кўрсатмалар аник берилиши ва факат алгоритмда кўрсатилган тартибда бажарилиши шарт экан.

4.Оммавийлик. Хар бир алгоритм мазмунига кўра бир турдаги масалаларнинг барчаси учун хам ўринли бўлиши керак. Яъни масаладаги бошлангич маълумотлар кандай бўлишидан катъий назар алгорим шу хилдаги хар кандай масалани йечишга ярокли бўлиши керак. Масалан, икки оддий касрнинг умумий махражини топиш алгоритми, касрларни турлича ўзгартириб берсангиз хам уларнинг умумий махражларини аниклаб бераверади. Ёки учбурчакнинг юзини топиш алгоритми, учбурчакнинг кандай бўлишидан катъий назар, унинг юзини хисоблаб бераверади.

5.Натижавийлик. Хар бир алгоритм чекли сондаги кадамлардан сўнг албатта натижа бериши шарт. Бажариладиган амаллар кўп бўлса хам барибир натижага олиб келиши керак. Чекли кадамдан сўнг кўйилган масала йечимга эга эмаслигини аниклаш хам натижа хисобланади. Агар кўрилаётган жараён чексиз давом этиб натижа бермаса, уни алгоритм деб атай олмаймиз.

Маъруза 6
Программалаш тиллари.
Тил синтаксиси. Бекус - Науре шакли

Энг кенг таркалган метатиллардан бири Бекус-Наурнинг металингвистик формулалари ва синтактик диаграммаларидир. Бир алгоритмик тилнинг конун коидаларини аник ва бир кийматли аниклаш учун махсус тушинча ва белгилар зарур бўлади. Тилнинг хар бир тушинчаси учун ягона метоформула мавжуд бўлиши керак ва унинг гап кисмида киритилаётган тушинча, яъни метаўзгарувчи кўрсатилади. Ўнг томонда эса, метоўзгарувчининг кабул килиши мумкин бўлган кийматлар тўплами келтирилади. Одатда метоўзгарувчилар махсус <> кавслар ичида ёзилади. Масалан: <сон>, <арифметик ифода>. Метоформуланинг чап ва ўнг кисмлари махсус метосимвол билан ажратилади ва у “таъриф бўйича” деган маънони англатади. Масалан, куйидаги метоформула

<ўзгарувчи>::=A|B

ўзгарувчи таъриф бўйича A ёки B харфидир деган маънони ифодалайди.

<ифода>::=<ўзгарувчи>| <ўзгарувчи>+<ўзгарувчи> |<ўзгарувчи> - <ўзгарувчи>

метоформула эса, юкоридаги <ўзгарувчи> метоформуласига боглик холда <ифода> сифатида куйидаги 10 та ифодадан ихтиёрий биттаси бўлиши мумкин деган маънони англатади:

A, B, A + A, A + B, B + A, B + B, A-A, A-B, B-A, B-B.

Эслатиб ўтамиз вертикал чизик ёки деган маънони ифодалайди. Фараз килайлик биз <иккилик код> деган тушунчасини киритмокчимиз ва иккилик код деганда 0 ва 1 ракамлардан ташкил топган ихтиёрий кетма-кетликни назарда тутамиз. Умуман олганда, 0 ва 1 нинг ўзлари хам иккилик код ва уларнинг ёки 0 ва 1 ракамларидан бирортасини ёзсак, яна иккилик код пайдо бўлади юкорида келтирилган фикрларни куйидаги метаформулалар ёрдамида оддий ва киска кўринишда ифодалаш мумкин.

<иккилик ракам>::=0 | 1

<иккилик код>::=<иккилик ракам>+<иккилик код><иккилик ракам>

метоформулаларда ишлатиладиган фигурали кавс { }, унинг ичидаги конструксиянинг кўп марта такрорланишини ифодалайди. Юкоридаги иккилик код тушунчаси фигурали кавслар ёрдамида куйидагига киритилиши мумкин.

<иккилик ракам> ::=0 | 1

<иккилик код>::=<иккилик ракам> {<иккилик ракам>}

Маъруза 7
С++ тили ва унинг лексик асоси

С++ тилида программа яратиш бир нечта босқичлардан иборат бўлади. Дастлаб, матн таҳририда (одатда программалаш муҳитининг таҳририда) программа матни терилади, бу файлнинг кенгайтмаси «.cpp» бўлади, Кейинги босқичда программа матн ёзилган файл компиляторга узатилади, агарда программада хатоликлар бўлмаса, компилятор «.obj» кенгайтмали объект модул файлини ҳосил қилади. Охирги қадамда компоновка (йиғувчи) ёрдамида «.exe» кенгайтмали бажарилувчи файл - программа ҳосил бўлади. Босқичларда юзага келувчи файлларнинг номлари бошланғич матн файлининг номи билан бир хил бўлади.

Компиляция жараёнининг ўзи ҳам иккита босқичдан ташкил топади. Бошида препроцессор ишлайди, у матндаги компиляция директиваларини бажаради, хусусан #include директиваси бўйича кўрсатилган кутубхоналардан С++ тилида ёзилган модулларни прог-рамма таркибига киритади. Шундан сўнг кенгайтирилган программа матни компиляторга узатилади. Компилятор ўзи ҳам программа бўлиб, унинг учун кирувчи маълумот бўлиб, С++ тилида ёзилган программа матни ҳисобланади. Компилятор программа матнини лексема (атомар) элементларга ажратади ва уни лексик, кейинчалик синтаксик таҳлил қилади. Лексик таҳлил жараёнида у матнни лексемаларга ажратиш учун «пробел ажратувчисини» ишлатади. Пробел ажратувчисига - пробел белгиси (‘␣’), ‘\t’ - табуляция белгиси, ‘\n’- кейинги қаторга ўтиш белгиси, бошқа ажратувчилар ва изоҳлар ҳисобланади.

Программа матни тушунарли бўлиши учун изоҳлар ишлатилади. Изоҳлар компилятор томонидан «ўтказиб» юборилади ва улар программа амал қилишига ҳеч қандай таъсир қилмайди.

С++ тилида изоҳлар икки кўринишда ёзилиши мумкин.

Биринчисида “**/*”** дан бошланиб, “***/”** белгилар оралиғида жойлашган барча белгилар кетма-кетлиги изоҳ ҳисобланади, иккинчиси «сатрий изоҳ» деб номланади ва у “**//”** белгилардан бошланган ва сатр охиригача ёзилган белгилар кетма-кетлиги бўлади. Изоҳнинг биринчи кўринишида ёзилган изоҳлар бир неча сатр бўлиши ва улардан кейин С++ операторлари давом этиши мумкин.

Мисол.

```
int main()
{
```

```
 // бу қатор изоҳ ҳисобланади
 int a=0; // int d;
 int c;
 /* int b=15 */
 /* - изоҳ бошланиши
 a=c;
 изоҳ тугаши */
 return 0;
}
```

Программада d, b ўзгарувчилар эълонлари инобатга олинмайди ва a=c амали бажарилмайди.

Қуйида C++ тилидаги содда программа матни келтирилган.

```
# include <iostream.h> // сарлавҳа файлни қўшиш
int main ()                // бош функция тавсифи
{                          // блок бошланиши
 cout << "Salom Olam!\n";// сатрни чоп этиш
 return 0;               // функция қайтарадиган қиймат
}                          // блок тугаши
```

Программа бажарилиши натижасида экранга “Salom Olam!” сатри чоп этилади.

Программанинг 1-сатрида #include.. препроцессор директиваси бўлиб, программа кодига оқимли ўқиш/ёзиш функциялари ва унинг ўзгарувчилари эълони жойлашган «iostream.h» сарлавҳа файлини қўшади. Кейинги қаторларда программанинг ягона, асосий функцияси - main() функцияси тавсифи келтирилган. Шуни қайд этиш керакки, C++ программасида албатта main() функцияси бўлиши шарт ва программа шу функцияни бажариш билан ўз ишини бошлайди.

Программа танасида консол режимида белгилар кетма-кетлигини оқимга чиқариш амали қўлланилган. Маълумотларни стандарт оқимга (экранга) чиқариш учун қуйидаги формат ишлатилган:

cout << <ифода>;

Бу ерда <ифода> сифатида ўзгарувчи ёки синтаксиси тўғри ёзилган ва қандайдир қиймат қабул қилувчи тил ифодаси келиши мумкин (*кейинчалик, бурчак қавс ичига олинган ўзбекча сатр остини тил таркибига кирмайдиган тушунча деб қабул қилиш керак)*.

Масалан:

```
int uzg=324;
cout<<uzg;     // бутун сон чоп этилади
```

Берилганларни стандарт оқимдан (одатда клавиатурадан) ўқиш қуйидаги форматда амалга оширилади:

cin >> <ўзгарувчи>;

Бу ерда <ўзгарувчи> қиймат қабул қилувчи ўзгарувчининг номи.

Мисол:

```
int Yosh;
cout<<"Yoshingizni kiriting_" ;
cin>>Yosh;
```

Бутун турдаги Yosh ўзгарувчиси киритилган қийматни ўзлаш-тиради. Киритилган қийматни ўзгарувчи турига мос келишини текшириш масъулияти программа тузувчисининг зиммасига юкланади.

Бир пайтнинг ўзида пробел воситасида бир нечта ва ҳар хил турдаги қийматларни оқимдан киритиш мумкин. Қиймат киритиш <enter> тугмасини босиш билан тугайди.

Агар киритилган қийматлар сони ўзгарувчилар сонидан кўп бўлса, «ортиқча» қийматлар буфер хотирада сақланиб қолади.

```
#include <iostream.h>
int main()
{
 int x,y;  float z;
 cin>>x>>y>>z;
 cout<<"O'qilgan qiymatlar\n";
 cout<<x<<'\t'<<y<<'\t'<<z;
 return 0;
}
```

Ўзгарувчиларга қиймат киритиш учун клавиатура орқали

```
10 20 3.14 <enter>
```

ҳаракати амалга оширилади. Шуни қайд этиш керакки, оқимга қиймат киритишда пробел ажратувчи ҳисобланади. Ҳақиқий соннинг бутун ва каср қисмлари '.' белгиси билан ажратилади.

Маъруза 8
C++ тилида берилганлар ва уларнинг турлари

Ўзгармаслар

Ўзгармас (*литерал*) - бу фиксирланган сонни, сатрни ва белгини ифодаловчи лексемадир.

Ўзгармаслар бешта гуруҳга бўлинади - *бутун*, *ҳақиқий* (*сузувчи нуқтали*), *санаб ўтилувчи*, *белги* (*литерли*) ва *сатр* («*стринг», литерли сатр*).

Компилятор ўзгармасни лексема сифатида аниқлайди, унга хотирадан жой ажратади, кўриниши ва қийматига (турига) қараб мос гуруҳларга бўлади.

Бутун ўзгармаслар. Бутун ўзгармаслар қуйидаги форматларда бўлади:

- ўнлик сон;
- саккизлик сон;
- ўн олтилик сон.

Ўнлик ўзгармас 0 рақамидан фарқли рақамдан бошланувчи рақамлар кетма-кетлиги ва 0 ҳисобланади: **`0; 123; 7987; 11.`**

Манфий ўзгармас - бу ишорасиз ўзгармас бўлиб, унга фақат ишорани ўзгартириш амали қўлланилган деб ҳисобланади.

Саккизлик ўзгармас 0 рақамидан бошланувчи саккизлик саноқ системаси (0,1,..,7) рақамларидан ташкил топган рақамлар кетма-кетлиги:

```
023; 0777; 0.
```

Ўн олтилик ўзгармас 0x ёки 0X белгиларидан бошланадиган ўн олтилик саноқ системаси рақамларидан иборат кетма-кетлик ҳисоб-ланади:

```
0x1A; 0X9F2D; 0x23.
```

Ҳарф белгилар ихтиёрий регистрларда берилиши мумкин.

Компилятор соннинг қийматига қараб унга мос турни белгилайди. Агар тилда белгиланган турлар программа тузувчини қаноатлантирмаса, у ошкор равишда турни кўрсатиши мумкин. Бунинг учун бутун ўзгармас рақамлари охирига, пробелсиз l ёки L (long), u ёки U (unsigned) ёзилади. Зарур ҳолларда битта ўзгармас учун бу белгиларнинг иккитасини ҳам ишлатиш мумкин:

```
45lu, 012Ul, 0xA2L.
```

Ҳақиқий ўзгармаслар. Ҳақиқий ўзгармаслар - сузувчи нуқтали сон бўлиб, у икки хил форматда берилиши мумкин:

- ўнлик фиксирланган нуқтали форматда. Бу кўринишда сон нуқта орқали ажратилган бутун ва каср қисмлар кўринишида бўлади. Соннинг бутун ёки каср қисми бўлмаслиги мумкин, лекин нуқта албатта бўлиши керак. Фиксирланган нуқтали ўзгармасларга мисоллар: **`24.56; 13.0; 66.; .87`**;
- экспоненциал шаклда ҳақиқий ўзгармас 6 қисмдан иборат бўлади:
  1) бутун қисми (ўнли бутун сон);
  2) ўнли каср нуқта белгиси;
  3) каср қисми (ўнлик ишорасиз ўзгармас);
  4) экспонента белгиси ‘e’ ёки ‘E’;
  5) ўн даражаси кўрсаткичи (ўнли бутун сон);
  6) қўшимча белгиси (‘F’ ёки ‘f’ , ‘L’ ёки ‘l’).

Экспоненциал шаклдаги ўзгармас сонларга мисоллар: **`1e2; 5e+3; .25e4; 31.4e-1`** .

Белги ўзгармаслар. Белги ўзгармаслар қўштирноқ (‘,’-апострофлар) ичига олинган алоҳида белгилардан ташкил топади ва у char калит сўзи билан аниқланади. Белги ўзгармас учун хотирада бир байт жой ажратилади ва унда бутун сон кўринишидаги белгининг ASCII коди жойлашади. Қуйидагилар белги ўзгармасларга мисол бўлади: **`’e’, ’@’, ’7’, ’z’,’W’,’+’,’ш’,’*’,’a’, ’s’.`**

1.1-жадвал. C++ тилида escape -белгилар жадвали

Escape белгилари	Ички код (16 сон)	Номи	Амал
\\	0x5C	\	Тескари ён чизиқни чоп этиш
\’	0x27	‘	Апострофни чоп этиш
\”	0x22	“	Қўштирноқни чоп этиш
\?	0x3F	?	Сўроқ белгиси
\a	0x07	bel	Товуш сигналини бериш
\b	0x08	bs	Курсорни 1 белги ўрнига орқага қайтариш
\f	0x0C	ff	Саҳифани ўтказиш
\n	0x0A	lf	Қаторни ўтказиш
\r	0x0D	cr	Курсорни айни қаторнинг бошига қайтариш
\t	0x09	ht	Навбатдаги табуляция жойига ўтиш
\v	0x0D	vt	Вертикал табуляция (пастга)
\000	000		Саккизлик коди
\xNN	0xNN		Белги ўн олтилик коди билан берилган

Айрим белги ўзгармаслар ‘\’ белгисидан бошланади, бу белги биринчидан, график кўринишга эга бўлмаган ўзгармасларни белги-лайди, иккинчидан, махсус вазифалар юкланган белгилар - апостроф белгиси(‘), савол белгисини (‘?’), тескари ён чизиқ белгисини (‘\’) ва иккита қўштирноқ белгисини (‘“’) чоп қилиш учун ишлатилади. Ундан ташқари, бу белги орқали белгини кўринишини эмас, балки ошкор равишда унинг ASCII кодини саккизлик ёки ўн олтилик шаклда ёзиш мумкин. Бундай белгидан бошланган белгилар escape кетма-кетликлар дейилади (1.1-жадвал).

C++ тилида қўшимча равишда wide ҳарфли ўзгармаслар ва кўп белгили ўзгармаслар аниқланган.

wide ҳарфли ўзгармаслар тури миллий кодларни белгилаш учун киритилган бўлиб, у wchar_t калит сўзи билан берилади, ҳамда хотирада 2 байт жой эгаллайди. Бу ўзгармас L белгисидан бошланади:

```
L’\013\022’, L’cc’
```

Кўп белгили ўзгармас тури int бўлиб, у тўртта белгидан иборат бўлиши мумкин:

```
’abc’, ’\001\002\003\004’.
```

Сатр ўзгармаслар. Иккита қўштирноқ (“,”) ичига олинган белгилар кетма-кетлиги *сатр ўзгармас* дейилади:

```
”Bu satr o’zgarmas va uning nomi string\n”
```

Сатр ичида escape кетма-кетлиги ҳам ишлатилиши мумкин, фақат бу кетма-кетлик апострофсиз ёзилади.

Пробел билан ажратиб ёзилган сатрлар компилятор томонидан ягона сатрга уланади (конкантенация):

```
”Satr - bu belgilar massivi” /* бу сатр кейинги сатрга қўшилади */ ”, uning turi char[]”;
```

Бу ёзув

```
”Satr - bu belgilar massivi, uning turi char[]”;
```

ёзуви билан эквивалент ҳисобланади.

Узун сатрни бир нечта қаторга ёзиш мумкин ва бунинг учун қатор охирида ‘\’ белгиси қўйилади:

```
”Kompilyator har bir satr uchun kompyuter xotirasida\
satr uzunligiga teng sondagi baytlardagi alohida \
xotira ajratadi va bitta - 0 qiymatli bayt qo’shadi”;
```

Маъруза 9
Ифодалар ва операторлар. Арифметик амаллар. Қиймат бериш оператори

Берилганларни қайта ишлаш учун C++ тилида амалларнинг жуда кенг мажмуаси аниқланган. *Амал* - бу қандайдир ҳаракат бўлиб, у битта (унар) ёки иккита (бинар) операндлар устида бажарилади, ҳисоб натижаси унинг қайтарувчи қиймати ҳисобланади.

Таянч арифметик амалларга қўшиш (+), айириш (-), кўпайтириш (*), бўлиш (/) ва бўлиш қолдиғини олиш (%) амалларини келтириш мумкин.

Амаллар қайтарадиган қийматларни ўзлаштириш учун қиймат бериш амали (=) ва унинг турли модификациялари ишлатилади: қўшиш, қиймат бериш билан (+=); айириш, қиймат бериш билан (−=); кўпайтириш, қиймат бериш билан (*=); бўлиш, қиймат бериш билан (/=); бўлиш қолдиғини олиш, қиймат бериш билан (%=) ва бошқалар. Бу ҳолатларнинг умумий кўриниши:

<ўзгарувчи><амал>=<ифода>;

Қуйидаги программа матнида айрим амалларга мисоллар келтирилган.

```
#include <iostream.h>
int main()
{
 int a=0,b=4,c=90; char z=’\t’;
 a=b; cout<<a<<z;          // a=4
 a=b+c+c+b; cout<<a<<z;// a= 4+90+90+4 = 188
 a=b-2; cout<<a<<z; // a=2
 a=b*3; cout<<a<<z; // a=4*3 = 12
 a=c/(b+6); cout<<a<<z;// a=90/(4+6) =9
 cout<<a%2<<z;      // 9%2=1
 a+=b;    cout<<a<<z;    // a=a+b = 9+4 =13
 a*=c-50; cout<<a<<z;    //a=a*(c-50)=13*(90-50)=520
 a-=38;   cout<<a<<z;    // a=a-38=520-38=482
 a%=8;    cout<<a<<z;    // a=a%8=482%8=2
 return 0;
}
```

Программа бажарилиши натижасида экранга қуйидаги сонлар қатори пайдо бўлади:

```
4    188    2   12   9    1    482  2
```

Ифода тушунчаси

C++ тилида *ифода* - амаллар, операндлар ва пунктация белги-ларининг кетма-кетлиги бўлиб, компилятор томонидан берилганлар устида маълум бир амалларни бажаришга кўрсатма деб қабул қилинади. Ҳар қандай ‘;’ белги билан тугайдиган ифодага *тил кўрсатмаси* дейилади.

C++ тилидаги тил кўрсатмасига мисол:

```
x=3*(y-2.45);
y=Summa(a,9,c);
```

Инкремент ва декремент амаллари

C++ тилида операнд қийматини бирга ошириш ва камайтириш-нинг самарали воситалари мавжуд. Булар инкремент (++) ва декремент (--) унар амаллардир.

Операндга нисбатан бу амалларнинг префикс ва постфикс кўри-нишлари бўлади. Префикс кўринишда амал тил кўрсатмаси бўйича иш бажарилишидан олдин операндга қўлланилади. Постфикс ҳолатда эса амал тил кўрсатмаси бўйича иш бажарилгандан кейин операндга қўлланилади.

Префикс ёки постфикс амал тушунчаси фақат қиймат бериш билан боғлиқ ифодаларда ўринли:

```
x=y++;        // постфикс
index =--i; // префикс
count++;      // унар амал, ”++count;” билан эквивалент
abc-- ;       // унар амал, ”--abc; ” билан эквивалент
```

Бу ерда у ўзгарувчининг қийматини х ўзгарувчисига ўзлаштири-лади ва кейин биттага оширилади, і ўзгарувчининг қиймати биттага камайтириб, index ўзгарувчисига ўзлаштирилади.

sizeof амали

Ҳар хил турдаги ўзгарувчилар компьютер хотирасида турли сондаги байтларни эгаллайди. Бунда, ҳаттоки бир турдаги ўзгарувчилар ҳам қайси компьютерда ёки қайси операцион системада амал қилинишига қараб турли ўлчамдаги хотирани банд қилиши мумкин.

C++ тилида ихтиёрий (таянч ва ҳосилавий) турдаги ўзгарув-чиларнинг ўлчамини sizeof амали ёрдамида аниқланади. Бу амални ўзгармасга, турга ва ўзгарувчига қўлланиши мумкин.

Қуйида келтирилган программада компьютернинг платформа-сига мос равишда таянч турларининг ўлчамлари чоп қилинади.

```
int main()
{
 cout<<”int тури ўлчами:”<<sizeof(int)<<”\n”;
 cout<<”float тури ўлчами:”<<sizeof(float)<<”\n”;
 cout<<”double тури ўлчами:”<<sizeof(double) <<”\n”;
 cout<<”char тури ўлчами:”<<sizeof(char)<<”\n”;
 return 0;}
```

Разрядли мантиқий амаллар

Программа тузиш тажрибаси шуни кўрсатадики, одатда қўйилган масалани ечишда бирор ҳолат рўй берган ёки йўқлигини ифодалаш учун 0 ва 1 қиймат қабул қилувчи *байроқлардан* фойдала-нилади. Бу мақсадда бир ёки ундан ортиқ байтли ўзгарувчилардан фойдаланиш мумкин. Масалан, bool туридаги ўзгарувчини шу мақсадда ишлатса бўлади. Бошқа томондан, байроқ сифатида байт-нинг разрядларидан

фойдаланиш ҳам мумкин. Чунки разрядлар фақат иккита қийматни - 0 ва 1 сонларини қабул қилади. Бир байтда 8 разряд бўлгани учун унда 8 та байроқни кодлаш имконияти мавжуд.

Фараз қилайлик, қўриқлаш тизимига 5 та хона уланган ва тизим тахтасида 5 та чироқча (индикатор) хоналар ҳолатини билдиради: хона қўриқлаш тизими назоратида эканлигини мос индикаторнинг ёниб туриши (разряднинг 1 қиймати) ва хонани тизимга уланмаган-лигини индикатор ўчганлиги (разряднинг 0 қиймати) билдиради. Тизим ҳолатини ифодалаш учун бир байт етарли бўлади ва унинг кичик разрядидан бошлаб бештасини шу мақсадда ишлатиш мумкин:

7	6	5	4	3	2	1	0
			ind5	ind4	ind3	ind2	ind1

Масалан, байтнинг қуйидаги ҳолати 1, 4 ва 5 хоналар қўриқлаш тизимига уланганлигини билдиради:

7	6	5	4	3	2	1	0
x	x	x	1	1	0	0	1

Қуйидаги жадвалда C++ тилида байт разрядлари устида мантиқий амаллар мажмуаси келтирилган.

3.1-жадвал. Байт разрядлари устида мантиқий амаллар

Амаллар	**Мазмуни**
&	Мантиқий ВА (кўпайтириш)
|	Мантиқий ЁКИ (қўшиш)
^	Истисно қилувчи ЁКИ
~	Мантиқий ИНКОР (инверсия)

Разрядли мантиқий амалларнинг бажариш натижаларини жадвал кўринишида кўрсатиш мумкин.

3.2-жадвал. Разрядли мантиқий амалларнинг бажариш натижалари

A	**B**	**C=A&B**	**C=A|B**	**C=A^B**	**C=~A**
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

Юқоридаги келтирилган мисол учун қўриқлаш тизимини ифода-ловчи бир байтли char туридаги ўзгарувчини эълон қилиш мумкин:

```
char q_taxtasi=0;
```

Бу ерда q_taxtasi ўзгарувчисига 0 қиймат бериш орқали барча хоналар қўриқлаш тизимига уланмаганлиги ифодаланади:

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

Агар 3-хонани тизимга улаш зарур бўлса

```
q_taxtasi=q_taxtasi|0x04;
```

амалини бажариш керак, чунки $0x04_{16}=00000100_2$ ва мантиқий ЁКИ амали натижасида q_taxtasi ўзгарувчиси байти қуйидаги кўринишда бўлади:

7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	0

Худди шундай йўл билан бошқа хоналарни тизимга улаш мумкин, зарур бўлса бирданига иккитасини (зарур бўлса барчасини):

```
q_taxtasi=q_taxtasi|0x1F;
```

Мантиқий кўпайтириш орқали хоналарни қўриқлаш тизимидан чиқариш мумкин:

q_taxtasi=q_taxtasi&0xFD; // $0xFD_{16}=11111101_2$

Маъруза 10
Программа бажарилишини бошкариш. Оператор тушунчаси

Программалаш тили операторлари ечилаётган масала алгорит-мини амалга ошириш учун ишлатилади. Операторлар *чизиқли* ва *бошқарув операторларига* бўлинади. Аксарият ҳолатларда оператор-лар «нуқта-вергул» (‘;’) белгиси билан тугалланади ва у компилятор томонидан алоҳида оператор деб қабул қилинади (for операторининг қавс ичида турган ифодалари бундан мустасно). Бундай оператор *ифода оператори* дейилади. Қиймат бериш амаллари гуруҳи, хусусан, қиймат бериш операторлари ифода операторлари ҳисобланади:

```
I++; --j; k+=I;
```

Программа тузиш амалиётида бўш оператор - ‘;’ ишлатилади. Гарчи бу оператор ҳеч нима бажармаса ҳам, ҳисоблаш ифодаларини тил қурилмаларига мос келишини таъминлайди. Айрим ҳолларда юзага келган «боши берк» ҳолатлардан чиқиб кетиш имконини беради.

Ўзгарувчиларни эълон қилиш ҳам оператор ҳисобланади ва уларга *эълон оператори* дейилади.

Маъруза 11
Шарт операторлари

Олдинги бобда мисол тариқасида келтирилган программаларда амаллар ёзилиш тартибида кетма-кет ва фақат бир марта бажариладиган ҳолатлар, яъни чизиқли алгоритмлар келтирилган. Амалда эса камдан-кам масалалар шу тариқа ечилиши мумкин. Аксарият масалалар юзага келадиган турли ҳолатларга боғлиқ равишда мос қарор қабул қилишни (ечимни) талаб этади. C++ тили программанинг алоҳида бўлакларининг бажарилиш тартибини бошқаришга имкон берувчи қурилмаларнинг етарлича катта мажмуасига эга. Масалан, программа бажарилишининг бирорта қадамида қандайдир шартни текшириш натижасига кўра бошқарувни программанинг у ёки бу бўлагига узатиш мумкин (тармоқланувчи алгоритм). Тармоқланишни амалга ошириш учун шартли оператордан фойдаланилади.

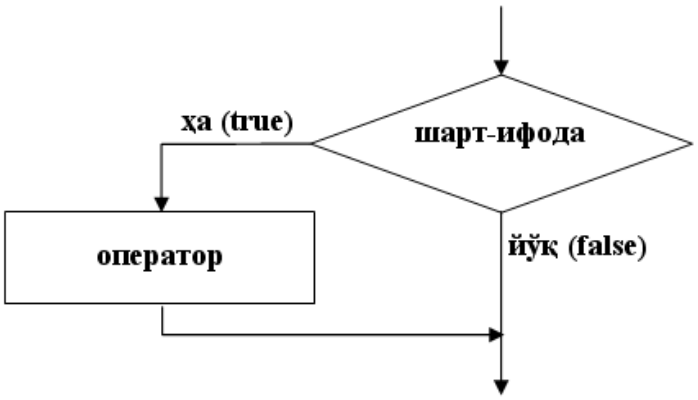
if оператори

if оператори қандайдир шартни ростликка текшириш натижасига кўра программада тармоқланишни амалга оширади:

if (<шарт>)<оператор>;

Бу ерда <шарт> ҳар қандай ифода бўлиши мумкин, одатда у таққослаш амали бўлади.

Агар шарт 0 қийматидан фарқли ёки рост (true) бўлса, <оператор> бажарилади, акс ҳолда, яъни шарт 0 ёки ёлғон (false) бўлса, ҳеч қандай амал бажарилмайди ва бошқарув if операторидан кейинги операторга ўтади (агар у мавжуд бўлса). Ушбу ҳолат 4.1-расмда кўрсатилган.



4.1-расм. if() шарт операторининг блок схемаси

C++ тилининг қурилмалари операторларни блок кўринишида ташкил қилишга имкон беради. *Блок* - ‘{‘ ва ‘}’ белги оралиғига олинган операторлар кетма-кетлиги бўлиб, у компилятор томонидан яхлит бир оператор деб қабул қилинади. Блок ичида эълон операторлари ҳам бўлиши мумкин ва уларда эълон қилинган ўзгарувчилар фақат шу блок ичида кўринади (амал қилади), блокдан ташқарида кўринмайди. Блокдан кейин ‘;’ белгиси қўйилмаслиги мумкин, лекин блок ичидаги ҳар бир ифода ‘;’ белгиси билан якунланиши шарт.

Қуйида келтирилган программада if операторидан фойдаланиш кўрсатилган.

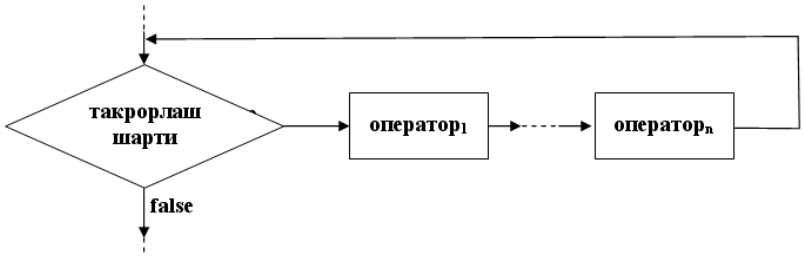
```
#include <iostream.h>
int main()
{
 int b;
 cin>>b;
 if (b>0)
 {              // b>0 шарт бажарилган ҳолат
 ...
 cout<<"b - musbat son";
 ...
 }
 if (b<0)
 cout<<"b - manfiy son"; // b<0 шарт бажарилган ҳолат
 return 0;
}
```

Программа бажарилиши жараёнида бутун турдаги b ўзгарувчи эълон қилинади ва унинг қиймати клавиатурадан ўқилади. Кейин b қийматини 0 сонидан катталиги текширилади, агар шарт бажарилса (true) , у ҳолда ‘{‘ ва ‘}’ белгилар ичидаги операторлар бажарилади ва экранга “b - мусбат сон” хабари чиқади. Агар шарт бажарилмаса, бу операторлар чеклаб ўтилади. Навбатдаги шарт оператори b ўзгарувчи қиймати манфийликка текширади, агар шарт бажарилса, ягона cout кўрсатмаси бажарилади ва экранга “b - манфий сон” хабари чиқади.

Маъруза 12
Такрорлаш операторлари

Программа бажарилишини бошқаришнинг бошқа бир кучли механизмларидан бири - такрорлаш операторлари ҳисобланади.

Такрорлаш оператори «такрорлаш шарти» деб номланувчи ифоданинг рост қийматида программанинг маълум бир қисмидаги операторларни (такрорлаш танасини) кўп марта такрор равишда бажаради (итаратив жараён) (4.2-расм).



4.2-расм. Такрорлаш операторининг блок схемаси

Такрорлаш ўзининг кириш ва чиқиш нуқталарига эга, лекин чиқиш нуқтасининг бўлмаслиги мумкин. Бу ҳолда такрорлашга *чексиз такрорлаш* дейилади. Чексиз такрорлаш учун такрорлашни давом эттириш шарти доимо рост бўлади.

Такрорлаш шартини текшириш такрорлаш танасидаги оператор-ларни бажаришдан олдин текширилиши мумкин (for, while такрорлашлари) ёки такрорлаш танасидаги операторлари бир марта бажарилгандан кейин текширилиши мумкин (do-while).

Такрорлаш операторлари ичма-ич жойлашган бўлиши мумкин.

for такрорлаш оператори

for такрорлаш операторининг синтаксиси қўйидаги кўринишга эга:

for (<ифода$_1$>; <ифода$_2$>;<ифода$_3$>) <оператор ёки блок>;

Бу оператор ўз ишини <ифода$_1$> ифодасини бажаришдан бошлайди. Кейин такрорлаш қадамлари бошланади. Ҳар бир қадамда <ифода$_2$> бажарилади, агар натижа 0 қийматидан фарқли ёки true бўлса, такрор-лаш танаси - <оператор ёки блок> бажарилади ва охирида <ифода$_3$> бажарилади. Агар <ифода$_2$> қиймати 0 (false) бўлса, такрорлаш жара-ёни тўхтайди ва бошқарув такрорлаш операторидан кейинги операторга ўтади. Шуни қайд қилиш керакки, <ифода$_2$> ифодаси вергул билан ажратилган бир нечта ифодалар бирлашмасидан иборат бўлиши мумкин, бу ҳолда охирги ифода қиймати такрорлаш шарти ҳисобланади. Такрорлаш танаси сифатида битта оператор, жумладан бўш оператор бўлиши ёки операторлар блоки келиши мумкин.

Мисол учун 10 дан 20 гача бўлган бутун сонлар йиғиндисини ҳисоблаш масаласини кўрайлик.

```
#include <iostream.h>
int main()
{
 int Summa=0;
 for (int i=10; i<=20; i++)
 Summa+=i;
 cout<<"Yig'indi=" <<Summa;
 return 0;
}
```

Программадаги такрорлаш оператори ўз ишини, і такрорлаш параметрига (такрорлаш санагичига) бошланғич қиймат - 10 сонини беришдан бошлайди ва ҳар бир такрорлаш қадамидан (итарациядан) кейин қавс ичидаги учинчи оператор бажарилиши ҳисобига унинг қиймати биттага ошади. Ҳар бир такрорлаш қадамида такрорлаш танасидаги оператор бажарилади, яъни Summa ўзгарувчисига і қиймати қўшилади. Такрорлаш санагичи і қиймати 21 бўлганда “i<=20” такрорлаш шарти false (0-қиймати) бўлади ва такрорлаш тугайди. Натижада бошқарув такрорлаш операторидан кейинги cout операторига ўтади ва экранга йиғинди чоп этилади.

Юқорида келтирилган мисолга қараб такрорлаш операторлари-нинг қавс ичидаги ифодаларига изоҳ бериш мумкин:

<ифода$_1$> - такрорлаш санагичи вазифасини бажарувчи ўзгарув-чига бошланғич қиймат беришга хизмат қилади ва у такрорлаш жараёни бошида фақат бир марта ҳисобланади. Ифодада ўзгарувчи эълони учраши мумкин ва бу ўзгарувчи такрорлаш

оператори танасида амал қилади ва такрорлаш операторидан ташқарида «кўринмайди» (C++ Builder компилятори учун);

<ифода$_2$> - такрорлашни бажариш ёки йўқлигини аниқлаб берувчи мантиқий ифода, агар шарт рост бўлса, такрорлаш давом этади, акс ҳолда йўқ. Агар бу ифода бўш бўлса, шарт доимо рост деб ҳисобланади;

<ифода$_3$> - одатда такрорлаш санагичининг қийматини ошириш (камайтириш) учун хизмат қилади ёки унда такрорлаш шартига таъсир қилувчи бошқа амаллар бўлиши мумкин.

Такрорлаш операторида қавс ичидаги ифодалар бўлмаслиги мумкин, лекин синтаксис ‘;’ бўлмаслигига рухсат бермайди. Шу сабабли, энг содда кўринишдаги такрорлаш оператори қуйидагича бўлади:

```
for (;;)cout <<"Cheksiz takrorlash…" ;
```

Агар такрорлаш жараёнида бир нечта ўзгарувчиларнинг қий-мати синхрон равишда ўзгариши керак бўлса, такрорлаш ифодаларида зарур операторларни ‘,’ билан ёзиш орқали бунга эришиш мумкин:

```
for(int i=10,j=2;i<=20;i++,j=i+10) {...};
```

Такрорлаш операторининг ҳар бир қадамида j ва i ўзгарувчи-ларнинг қийматлари мос равишда ўзгариб боради.

for операторида такрорлаш танаси бўлмаслиги ҳам мумкин. Масалан, программа бажарилишини маълум бир муддатга «тўхтаб» туриш зарур бўлса, бунга такрорлашни ҳеч қандай қўшимча ишларни бажармасдан амал қилиши орқали эришиш мумкин:

```
#include <iostream.h>
 int main()
 {int delay;
  ...
  for (delay=5000; delay>0; delay--); // бўш оператор
  ...
  return 0;}
```

Юқорида келтирилган 10 дан 20 гача бўлган сонлар йиғиндисини бўш танали такрорлаш оператори орқали ҳисоблаш мумкин:

```
...
for (int i=10; i<=20; Summa+=i++);
...
```

Такрорлаш оператори танаси сифатида операторлар блоки ишлатишини факториални ҳисоблаш мисолида кўрсатиш мумкин:

```
#include <iostream.h>
int main()
{ int a;
 unsigned long fact=1;
 cout <<"Butun sonni kiriting:_ ";
 cin >>a;
 if ((a>=0)&&(a<33))
 { for (int i=1; i<=a; i++) fact*=i;
  cout<<a<<"!= "<<fact<<'\n'; }
 return 0; }
```

Программа фойдаланувчи томонидан 0 дан 33 гача ораликдаги сон киритилганда амал қилади, чунки 34! қиймати unsigned long учун ажратилган разрядларга сиғмайди.

Маъруза 13
Кўрсатгичлар ва адресни олувчи узгарувчилар

Программа матнида ўзгарувчи эълон қилинганда, компилятор ўзгарувчига хотирадан жой ажратади. Бошқача айтганда, программа коди хотирага юкланганда берилганлар учун, улар жойлашадиган сегментнинг бошига нисбатан силжишини, яъни нисбий адресини аниқлайди ва объект код ҳосил қилишда ўзгарувчи учраган жойга унинг адресини жойлаштиради.

Умуман олганда, программадаги ўзгармаслар, ўзгарувчилар, функциялар ва синф объектлар адресларини хотиранинг алоҳида жойида сақлаш ва улар устидан амаллар бажариш мумкин. Қиймат-лари адрес бўлган ўзгарувчиларга *кўрсаткич ўзгарувчилар* дейилади.

Кўрсаткич уч хил турда бўлиши мумкин:

- бирорта объектга, хусусан ўзгарувчига кўрсаткич;
- функцияга кўрсаткич;
- void кўрсаткич.

Кўрсаткичнинг бу хусусиятлари унинг қабул қилиши мумкин бўлган қийматларида фарқланади.

Кўрсаткич албатта бирорта турга боғланган бўлиши керак, яъни у кўрсатган адресда қандайдир қиймат жойланиши мумкин ва бу қийматнинг хотирада қанча жой эгаллаши олдиндан маълум бўлиши шарт.

Функцияга кўрсаткич. Функцияга кўрсаткич программа жой-лашган хотирадаги функция кодининг бошланғич адресини кўрса-тади, яъни функция чақирилганда бошқарув айни шу адресга узатила-ди. Кўрсаткич орқали функцияни оддий ёки воситали чақириш амалга ошириш мумкин. Бунда функция унинг номи бўйича эмас, балки функцияга кўрсатувчи ўзгарувчи орқали чақирилади. Функцияни бошқа функцияга аргумент сифатида узатиш ҳам функция кўрсаткичи орқали бажарилади. Функцияга кўрсаткичнинг ёзилиш синтаксиси қуйидагича:

<тур> (* <ном>) (<параметрлар рўйхати>);

Бунда <тур>- функция қайтарувчи қиймат тури; *<ном> - кўрсаткич ўзгарувчининг номи; <параметрлар рўйхати> - функция параметр-ларининг ёки уларнинг турларининг рўйхати.

Масалан:

```
int (*fun)(float,float);
```

Бу ерда бутун сон турида қиймат қайтарадиган fun номидаги функцияга кўрсаткич эълон қилинган ва у иккита ҳақиқий турдаги параметрларга эга.

Масала. Берилган бутун n=100 ва a,b - ҳақиқий сонлар учун $f_1(x)=5\sin(3x)+x$, $f_2(x)=\cos(x)$ ва $f_3(x)=x^2+1$ функциялар учун $\int_a^b f(x)dx$ интегралини тўғри тўртбурчаклар формуласи билан тақрибан ҳисоблансин:

$$\int_a^b f(x)dx \approx h[f(x_1)+f(x_2)+...+f(x_n)],$$

бу ерда $h=\frac{b-a}{n}$, $x_i=a+ih-h/2, i=1..n$.

Программа бош функция, интеграл ҳисоблаш ва иккита математик функциялар - $f_1(x)$ ва $f_3(x)$ учун аниқланган функциялардан ташкил топади, $f_2(x)=\cos(x)$ функциянинг адреси «math.h» сарлавҳа файлидан олинади. Интеграл ҳисоблаш функциясига кўрсаткич орқали интеграли ҳисобланадиган функция адреси, a ва b - интеграл чегаралари қийматлари узатилади. Оралиқни бўлишлар сони - n глобал ўзгармас қилиб эълон қилинади.

```
#include <iostream.h>
#include <math.h>
const int n=100;
```

```
double f1(double x){return 5*sin(3*x)+x;}
double f3(double x){return x*x+1;}
double Integral(double(*f)(double),double a,double b)
{
 double x,s=0;
 double h=(b-a)/n;
 x=a-h/2;
 for(int i=1;i<=n; i++) s+=f(x+=h);
 s*=h;
 return s;
}
int main()
{
 double a,b;
 int menu;
 while(1)
 {
  cout<<"\nIsh regimini tanlang:\n";
  cout<<"1:f1(x)=5*sin(3*x)+x integralini\
  hisoblash\n";
  cout<<"2:f2(x)=cos(x) integralini hisoblash\n";
  cout<<"3:f3(x)=x^2+1 integralini hisoblash\n";
  cout<<"0:Programmadan chiqish\n";
  do
  {
   cout<<" Ish regimi-> ";
   cin>>menu;
  }
   while (menu<0 || menu>3);
  if(!menu)break;
  cout<<"Integral oralig'ining quyi chegarasi a=";
  cin>>a;
  cout<<"Integral oralig'ining yuqori chegarasi b=";
  cin>>b;
  cout<<"Funksiya integrali S=";
  switch (menu)
  {case 1 : cout<<Integral(f1,a,b)<<endl; break;
   case 2 : cout<<Integral(cos,a,b)<<endl; break;
   case 3 : cout<<Integral(f3,a,b)<<endl; }
 } return 0;
}
```

Программанинг иши чексиз такрорлаш оператори танасини бажаришдан иборат. Такрорлаш танасида фойдаланувчига иш режи-мини танлаш бўйича меню таклиф қилинади:

```
Ish regimini tanlang:
1: f1(x)=5*sin(3*x)+x integralini hisoblash
2: f2(x)=cos(x) integralini hisoblash
3: f3(x)=x^2+1 integralini hisoblash
0: Programmadan chiqish
Ish regimi->
```

Фойдаланувчи 0 ва 3 оралиғидаги бутун сонни киритиши керак. Агар киритилган сон (menu ўзгарувчи қиймати) 0 бўлса, break опера-тори ёрдамида такрорлашдан, кейин программадан чиқилади. Агар menu қиймати 1 ва 3 оралиғида бўлса, интегралнинг қуйи ва юқори чегараларини киритиш сўралади, ҳамда Integral() функцияси мос функция

адреси билан чақирилади ва натижа чоп этилади. Шунга эътибор бериш керакки, интеграл чегараларининг қийматларини тўғри киритилишига фойдаланувчи жавобгар.

Маъруза 14
Массивлар. Берилганлар массиви тушунчаси

Хотирада кетма-кет (регуляр) жойлашган бир хил турдаги қийматларга *массив* дейилади.

Одатда массивларга зарурат, катта ҳажмдаги, лекин чекланган миқдордаги ва тартибланган қийматларни қайта ишлаш билан боғлиқ масалаларни ечишда юзага келади. Фараз қилайлик, талабалар гуруҳининг рейтинг баллари билан ишлаш масаласи қўйилган. Унда гуруҳнинг ўртача рейтингини аниқлаш, рейтингларни камайиши бўйича тартиблаш, конкрет талабанинг рейтинги ҳақида маълумот бериш ва бошқа масала остиларини ечиш зарур бўлсин. Қайд этилган масалаларни ечиш учун берилганларнинг (рейтингларнинг) тартиб-ланган кетма-кетлиги зарур бўлади. Бу ерда тартибланганлик маъноси шундаки, кетма-кетликнинг ҳар бир қиймати ўз ўрнига эга бўлади (биринчи талабанинг рейтинги массивда биринчи ўринда, иккинчи талабаники - иккинчи ўринда ва ҳакоза). Берилганлар кетма-кет-лигини икки хил усулда ҳосил қилиш мумкин. Биринчи йўл - ҳар бир рейтинг учун алоҳида ўзгарувчи аниқлаш: Reyting1,…,ReytingN. Лекин, гуруҳдаги талабалар сони етарлича катта бўлганда, бу ўзгарув-чилар қатнашган программани тузиш катта қийинчиликларни юзага келтиради. Иккинчи йўл - берилганлар кетма-кетлигини ягона ном билан аниқлаб, унинг қийматларига мурожаатни, шу қийматларнинг кетма-кетликда жойлашган ўрнининг номери (индекси) орқали амалга оширишдир. Рейтинглар кетма-кетлигини Reyting деб номлаб, ундаги қийматларига $Reyting_1$,…,$Reyting_N$ кўринишида мурожаат қилиш мумкин. Одатда берилганларнинг бундай кўринишига массивлар дейилади. Массивларни математикадаги сонлар векторига ўхшатиш мумкин, чунки вектор ҳам ўзининг индивидуал номига эга ва у фиксирланган миқдордаги бир турдаги қийматлардан - сонлардан иборатдир.

Демак, массив - бу фиксирланган миқдордаги айрим қийматлар-нинг (массив элементларининг) тартибланган мажмуасидир. Барча элементлар бир хил турда бўлиши керак ва бу тур *элемент тури* ёки массив учун *таянч тур* деб номланади. Юқоридаги келтирилган мисолда Reyting - ҳақиқий турдаги *вектор* деб номланади.

Программада ишлатиладиган ҳар бир конкрет массив ўзининг индивидуал номига эга бўлиши керак. Бу номни *тўлиқ ўзгарувчи* дейилади, чунки унинг қиймати массивнинг ўзи бўлади. Массивнинг ҳар бир элементи массив номи, ҳамда квадрат қавсга олинган ва *элемент селектори* деб номланувчи индексни кўрсатиш орқали ошкор равишда белгиланади. Мурожаат синтаксиси:

<массив номи >[<индекс>]

Бу кўринишга *хусусий ўзгарувчи* дейилади, чунки унинг қиймати мас-сивнинг алоҳида элементидир. Бизнинг мисолда Reyting массивининг алоҳида компоненталарига Reyting[1],…,Reyting[N] хусусий ўзгарув-чилар орқали мурожаат қилиш мумкин. Бошқача бу ўзгарувчилар *индексли ўзгарувчилар* дейилади.

Массив индекси сифатида бутун сон қўлланилади. Умуман олганда индекс сифатида бутун сон қийматини қабул қиладиган ихтиёрий ифода ишлатилиши мумкин ва унинг қиймати массив элементи номерини аниқлайди. Ифода сифатида ўзгарувчи ҳам олиниши мумкинки, ўзгарувчининг қиймати ўзгариши билан муро-жаат қилинаётган массив элементини аниқловчи индекс ҳам ўзгаради. Шундай қилиб, программадаги битта индексли ўзгарувчи орқали массивнинг барча элементларини белгилаш (аниқлаш) мумкин бўлади. Масалан, Reyting[I] ўзгарувчиси орқали I ўзгарувчининг қийматига боғлиқ равишда Reyting массивининг ихтиёрий элементига мурожаат қилиш мавжуд.

Ҳақиқий турдаги (float, double) қийматлар тўплами чексиз бўлганлиги сабабли улар индекс сифатида ишлатилмайди.

C++ тилида индекс доимо 0 дан бошланади ва унинг энг катта қиймати массив эълонидаги узунликдан биттага кам бўлади.

Массив эълони қуйидагича бўлади:

<тур> <ном> [<узунлик>]={бошланғич қийматлар}.

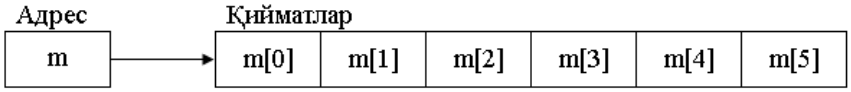
Бу ерда <узунлик> - ўзгармас ифода. Мисоллар:

```
int m[6]={1,4,-5,2,10,3];
float a[4];
```

Массив статик ва динамик бўлиши мумкин. Статик массивнинг узунлиги олдиндан маълум бўлиб, у хотирада маълум адресдан бошлаб кетма-кет жойлашади. Динамик массивни узунлиги прог-рамма бажарилиш жараёнида аниқланиб, у динамик хотирадаги айни пайтда бўш бўлган адресларга жойлашади. Масалан,

```
int m[6];
```

кўринишида эълон қилинган бир ўлчамли массив элементлари хотирада қуйидагича жойлашади:



7.1-расм. Бир ўлчамли массивнинг хотирадаги жойлашуви

Массивнинг i- элементига m[i] ёки *(m+i) - воситали мурожаат қилиш мумкин. Массив узунлигини sizeof(m) амали орқали аниқлади.

Массив эълонида унинг элементларига бошланғич қийматлар бериш мумкин ва унинг бир нечта вариантлари мавжуд.

1) ўлчами кўрсатилган массив элементларини тўлиқ инициали-зациялаш:

```
int t[5]={-10,5,15,4,3};
```

Бунда 5 та элементдан иборат бўлган t номли бутун турдаги бир ўлчамли массив эълон қилинган ва унинг барча элементларига бошланғич қийматлар берилган. Бу эълон қуйидаги эълон билан эквивалент:

```
int t[5];
t[0]=-10; t[1]=5; t[2]=15; t[3]=4; t[4]=3;
```

2) ўлчами кўрсатилган массив элементларини тўлиқмас инициа-лизациялаш:

```
int t[5]={-10,5,15};
```

Бу ерда фақат массив бошидаги учта элементга бошланғич қийматлар берилган. Шуни айтиб ўтиш керакки, массивнинг бошидаги ёки ўртасидаги элементларига қийматлар бермасдан, унинг охиридаги элементларга бошланғич қиймат бериш мумкин эмас. Агарда массив элементларига бошланғич қиймат берилмаса, унда келишув бўйича static ва extern модификатори билан эълон қилинган массив учун элементларининг қиймати 0 сонига тенг деб, automatic массивлар элементларининг бошланғич қийматлари номаълум ҳисобланади.

3) ўлчами кўрсатилмаган массив элементларини тўлиқ инициа-лизациялаш:

```
int t[]={-10,5,15,4,3};
```

Бу мисолда массивни барча элементларига қийматлар берилган ҳисобланади, массив узунлиги компилятор томонидан бошланғич қийматлар сонига қараб аниқланади. Агарда массив узунлиги берилмаса, бошланғич қиймати берилиши шарт.

Маъруза 15
Статик ва динамик массивлар. Кўп ўлчамли статик массивлар

C++ тилида массивлар элементининг турига чекловлар қўйил-майди, лекин бу турлар чекли ўлчамдаги объектларнинг тури бўлиши керак. Чунки компилятор массивнинг хотирадан қанча жой (байт) эгаллашини ҳисоблай олиши керак. Хусусан,

массив компонентаси массив бўлиши мумкин («векторлар-вектори»), натижада *матрица* деб номланувчи икки ўлчамли массив ҳосил бўлади.

Агар матрицанинг элементи ҳам вектор бўлса, уч ўлчамли массивлар - *куб* ҳосил бўлади. Шу йўл билан ечилаётган масалага боғлиқ равишда ихтиёрий ўлчамдаги массивларни яратиш мумкин.

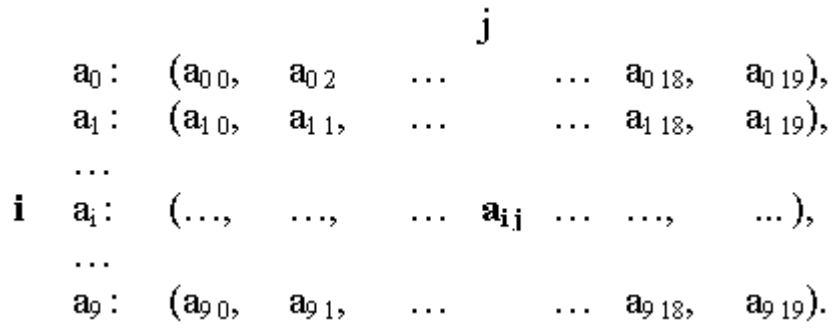
Икки ўлчамли массивнинг синтаксиси қуйидаги кўринишда бўлади:

<тур> <ном> [<узунлик >] [<узунлик>]

Масалан, 10×20 ўлчамли ҳақиқий сонлар массивининг эълони:

```
float a[10][20];
```

Эълон қилинган А матрицани кўриниши 7.2-расмда келтирилган.



7.2-расм. Икки ўлчамли массивнинг хотирадаги жойлашуви

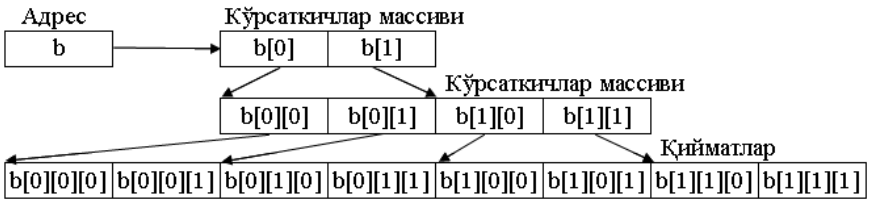
Энди адрес нуқтаи - назаридан кўп ўлчамли массив элемент-ларига мурожаат қилишни кўрайлик. Қуйидаги эълонлар берилган бўлсин:

```
int a[3][2];
float b[2][2][2];
```

Биринчи эълонда икки ўлчамли массив, яъни 2 сатр ва 3 устундан иборат матрица эълон қилинган, иккинчисида уч ўлчамли - 3 та 2x2 матрицадан иборат бўлган массив эълон қилинган. Унинг элементларига мурожаат схемаси:

7.3-расм. Икки ўлчамли массив элементларига мурожаат

Бу ерда a[i] кўрсаткичда i-чи сатрнинг бошланғич адреси жойла-шади, массив элементига a[i][j] кўринишидаги асосий мурожаатдан ташқари воситали мурожаат қилиш мумкин: *(*(a+i)+j) ёки *(a[i]+j).



7.3-расм. Уч ўлчамли массивнинг хотирада ташкил бўлиши

Массив элементларига мурожаат қилиш учун номдан кейин квадрат қавсда ҳар бир ўлчам учун индекс ёзилиши керак, масалан b[i][j][k]. Бу элементга воситали мурожаат ҳам қилиш мумкин ва унинг вариантлари:

((*(b+i)+j)+k) ёки *(*(b[i]+j)+k) ёки *(b[i][j] +k);

Динамик массивлар билан ишлаш

Статик массивларнинг камчиликлари шундаки, уларнинг ўлчамлари олдиндан маълум бўлиши керак, бундан ташқари бу ўлчамлар берилганларга ажратилган хотира сегментининг ўлчами билан чегараланган. Иккинчи томондан, етарлича катта ўлчамдаги массив эълон қилиб, конкрет масала ечилишида ажратилган хотира тўлиқ ишлатилмаслиги мумкин. Бу камчиликлар динамик массивлар-дан фойдаланиш орқали бартараф этилади, чунки улар программа ишлаши жараёнида керак бўлган ўлчамдаги массивларни яратиш ва зарурат қолмаганда йўқотиш имкониятини беради.

Динамик массивларга хотира ажратиш учун malloc(), calloc() функцияларидан ёки new операторидан фойдаланиш мумкин. Дина-мик объектга ажратилган хотирани бўшатиш учун free() функцияси ёки delete оператори ишлатилади.

Юқорида қайд қилинган функциялар «alloc.h» кутубхонасида жойлашган.

malloc() функциясининг синтаксиси

```
void * malloc(size_t size);
```

кўринишида бўлиб, у хотиранинг уюм қисмидан size байт ўлчамидаги узлуксиз соҳани ажратади. Агар хотира ажратиш муваффақиятли бўлса, malloc() функцияси ажратилган соҳанинг бошланиш адресини қайтаради. Талаб қилинган хотирани ажратиш муваффақиятсиз бўлса, функция NULL қийматини қайтаради.

Синтаксисдан кўриниб турибдики, функция void туридаги қиймат қайтаради. Амалда эса конкрет турдаги объект учун хотира ажратиш зарур бўлади. Бунинг учун void турини конкрет турга келтириш технологиясидан фойдаланилади. Масалан, бутун турдаги узунлиги 3 га тенг массивга жой ажратишни қуйидагича амалга ошириш мумкин:

```
int * pInt=(int*)malloc(3*sizeof(int));
```

calloc() функцияси malloc() функциясидан фарқли равишда массив учун жой ажратишдан ташқари массив элементларини 0 қиймати билан инициализация қилади. Бу функция синтаксиси

```
void * calloc(size_t num, size_t size);
```

кўринишда бўлиб, num параметри ажратилган соҳада нечта элемент борлигини, size ҳар бир элемент ўлчамини билдиради.

free() хотирани бўшатиш функцияси ўчириладиган хотира бўла-гига кўрсаткич бўлган ягона параметрга эга бўлади:

```
void free(void * block);
```

free() функцияси параметрининг void турида бўлиши ихтиёрий турдаги хотира бўлагини ўчириш имконини беради.

Маъруза 16
Символли массивлар

Стандарт C++ тили икки хилдаги белгилар мажмуасини қўллаб-қувватлайди. Биринчи тоифага, анъанавий, «тор» белгилар деб номла-нувчи 8-битли белгилар мажмуаси киради, иккинчисига 16-битли «кенг» белгилар киради. Тил кутубхонасида ҳар бир гуруҳ белгилари учун махсус функциялар тўплами аниқланган.

C++ тилида сатр учун махсус тур аниқланмаган. Сатр char тури-даги белгилар массиви сифатида қаралади ва бу белгилар кетма-кетлиги *сатр терминатори* деб номланувчи 0 кодли белги билан тугайди (‘\0’). Одатда, нол-терминатор билан тугайдиган сатрларни *ASCIIZ-сатрлар* дейилади.

Қуйидаги жадвалда C++ тилида белги сифатида ишлатилиши мумкин бўлган ўзгармаслар тўплами келтирилган.

8.1-жадвал. C++ тилидаги белги ўзгармаслар

Белгилар синфлари	Белги ўзгармаслар
Катта ҳарфлар	‘A’ …’Z’, ‘A’…’Я’
Кичик ҳарфлар	‘a’ …’z’, ‘a’…’я’
Рақамлар	‘0’ …’9’
Бўш жой	горизонтал табуляция (ASCII коди 9), сатрни ўтказиш (ASCII коди 10), вертикал табуляция (ASCII коди 11), формани ўтказиш (ASCII коди 12), кареткани қайтариш (ASCII коди 13)
Пунктуация белгилари (ажратувчилар)	! ” # $ & ‘ ( ) * + - , . / : ; < = > ? @ [\ ] ^ _ { | } ~
Бошқарув белгилари	ASCII коди 0...1Fh оралиғида ва 7Fh бўлган белгилар
Пробел	ASCII коди 32 бўлган белги
Ўн олтилик рақамлар	‘0’…’9’, ‘A’…’F’, ‘a’…’f’

Сатр массиви эълон қилинишида, сатр охирига терминатор қўйилиши ва натижада сатрга қўшимча битта байт бўлишини инобатга олиниши керак:

```
char satr[10];
```

Ушбу эълонда satr сатри учун жами 10 байт ажратилади - 9 сатр ҳосил қилувчи белгилар учун ва 1 байт терминатор учун.

Сатр ўзгарувчилар эълон қилинишида бошланғич қийматларни қабул қилиши мумкин. Бу ҳолда компилятор автоматик равишда сатр узунлиги ҳисоблайди ва сатр охирига терминаторни қўшиб қўяди:

```
char Hafta_kuni[]=”Juma”;
```

Ушбу эълон қуйидаги эълон билан эквивалент:

```
char Hafta_kuni[]={’J’,’u’,’m’,’a’,’\0’};
```

Сатр қийматини ўқишда оқимли ўқиш оператори “>>” ўрнига getline() функциясини ишлатган маъқул ҳисобланади, чунки оқимли ўқишда пробеллар инкор қилинади (гарчи улар сатр белгиси ҳисоб-ланса ҳам) ва ўқилаётган белгилар кетма-кетлиги сатрдан «ошиб» кетганда ҳам белгиларни киритиш давом этиши мумкин. Натижада сатр ўзига ажратилган ўлчамдан ортиқ белгиларни «қабул» қилади. Шу сабабли, getline() функцияси иккита параметрга эга бўлиб, биринчи параметр ўқиш амалга оширилаётган сатрга кўрсаткич, иккинчи параметрда эса ўқилиши керак бўлган белгилар сони кўрсатилади. Сатрни getline() функцияси орқали ўқишга мисол кўрайлик:

```
#include <iostream.h>
int main()
{
 char satr[6];
 cout<<”Satrni kiriting: “<<’\n’;
 cin.getline(satr,6);
 cout<<”Siz kiritgan satr: “<<satr;
 return 0;
}
```

Программада ишлатилган satr сатри 5 та белгини қабул қилиши мумкин, ортиқчалари ташлаб юборилади. getline() функциясига муро-жаатда иккинчи параметр қиймати ўқилаётган сатр узунлигидан катта бўлмаслиги керак.

Сатр билан ишлайдиган функцияларнинг аксарияти «string.h» кутубхонасида жамланган. Нисбатан кўп ишлатиладиган функциялар-нинг тавсифини келтирамиз.

Сатр узунлигини аниқлаш функциялари

Сатрлар билан ишлашда, аксарият ҳолларда сатр узунлигини билиш зарур бўлади. Бунинг учун «string.h» кутубхонасида strlen() функцияси аниқланган бўлиб, унинг синтаксиси қуйидагича бўлади:

```
size_t strlen(const char* string)
```

Бу функция узунлиги ҳисобланиши керак бўлган сатр бошига кўрсаткич бўлган ягона параметрга эга ва у натижа сифатида ишорасиз бутун сонни қайтаради. strlen() функцияси сатрнинг реал узун-лигидан битта кам қиймат қайтаради, яъни нол-терминатор ўрни ҳисобга олинмайди.

Худди шу мақсадда sizeof() функциясидан ҳам фойдаланиш мумкин ва у strlen() функциясидан фарқли равишда сатрнинг реал узунлигини қайтаради. Қуйида келтирилган мисолда сатр узунлигини ҳисоблашнинг ҳар иккита варианти келтирилган:

```
#include <iostream.h>
#include <string.h>
int main()
{
 char Str[]="1234567890";
 cout <<"strlen(Str)="<<strlen(Str)<<endl;
 cout<<"sizeof(Str)="<<sizeof(Str)<<endl;
 return 0;
}
```

Программа ишлаши натижасида экранга

```
strlen(Str)=10
sizeof(Str)=11
```

хабарлари чиқади.

Одатда sizeof() функциясидан getline() функциясининг иккинчи аргументи сифати ишлатилади ва сатр узунлигини яққол кўрсатмаслик имконини беради:

```
cin.getline(Satr, sizeof(Satr));
```

Маъруза 17
Куп улчовли массивлар

Динамик массивларга хотира ажратиш учун malloc(), calloc() функцияларидан ёки new операторидан фойдаланиш мумкин. Дина-мик объектга ажратилган хотирани бўшатиш учун free() функцияси ёки delete оператори ишлатилади.

Юқорида қайд қилинган функциялар «alloc.h» кутубхонасида жойлашган.

malloc() функциясининг синтаксиси

```
void * malloc(size_t size);
```

кўринишида бўлиб, у хотиранинг уюм қисмидан size байт ўлчамидаги узлуксиз соҳани ажратади. Агар хотира ажратиш муваффақиятли бўлса, malloc() функцияси ажратилган соҳанинг бошланиш адресини қайтаради. Талаб қилинган хотирани ажратиш муваффақиятсиз бўлса, функция NULL қийматини қайтаради.

Синтаксисдан кўриниб турибдики, функция void туридаги қиймат қайтаради. Амалда эса конкрет турдаги объект учун хотира ажратиш зарур бўлади. Бунинг учун void турини конкрет турга келтириш технологиясидан фойдаланилади. Масалан, бутун турдаги узунлиги 3 га тенг массивга жой ажратишни қуйидагича амалга ошириш мумкин:

```
int * pInt=(int*)malloc(3*sizeof(int));
```

calloc() функцияси malloc() функциясидан фарқли равишда массив учун жой ажратишдан ташқари массив элементларини 0 қиймати билан инициализация қилади. Бу функция синтаксиси

```
void * calloc(size_t num, size_t size);
```

кўринишда бўлиб, num параметри ажратилган соҳада нечта элемент борлигини, size ҳар бир элемент ўлчамини билдиради.

free() хотирани бўшатиш функцияси ўчириладиган хотира бўла-гига кўрсаткич бўлган ягона параметрга эга бўлади:

```
void free(void * block);
```

free() функцияси параметрининг void турида бўлиши ихтиёрий турдаги хотира бўлагини ўчириш имконини беради.

Қуйидаги программада 10 та бутун сондан иборат динамик массив яратиш, унга қиймат бериш ва ўчириш амаллари бажарилган.

```
#include <iostream.h>
#include <alloc.h>
int main()
{
 int * pVector;
 if ((pVector=(int*)malloc(10*sizeof(int)))==NULL)
 {
  cout<<"Xotira etarli emas!!!";
  return 1;
 }
 // ажратилган хотира соҳасини тўлдириш
 for(int i=0;i<10;i++) *(pVector+i)=i;
 // вектор элементларини чоп этиш
 for(int i=0; i<10; i++) cout<<*(pVector+i)<<endl;
 // ажратилган хотира бўлагини қайтариш (ўчириш)
 free(pVector);
 return 0;
}
```

Кейинги программада $n \times n$ ўлчамли ҳақиқий сонлар массиви-нинг бош диагоналидан юқорида жойлашган элементлар йиғинди-сини ҳисоблаш масаласи ечилган.

```
#include <iostream.h>
#include <alloc.h>
int main()
{
 int n;
 float * pMatr, s=0;
 cout<<"A(n,n): n=";
 cin>>n;
 if((pMatr=(float*)malloc(n*n*sizeof(float)))==NULL)
 {
  cout<<"Xotira etarli emas!!!";
  return 1;
 }
 for(int i=0;i<n;i++)
  for(int j=0;j<n;j++)cin>>*(pMatr+i*n+j);
 for(int i=0;i<n;i++)
  for(int j=i+1;j<n;j++)s+=*(pMatr+i*n+j);
 cout<<"Matritsa bosh diagonalidan yuqoridagi ";
 cout<<"elementlar yig`indisi S="<<s<<endl;
```

```
 return 0;
}
```

new оператори ёрдамида, массивга хотира ажратишда объект туридан кейин квадрат қавс ичида объектлар сони кўрсатилади. Масалан, бутун турдаги 10 та сондан иборат массивга жой ажратиш учун

```
pVector=new int[10];
```

ифодаси ёзилиши керак. Бунга қарама-қарши равишда, бу усулда ажратилган хотирани бўшатиш учун

```
delete [] pVector;
```

кўрсатмасини бериш керак бўлади.

Икки ўлчамли динамик массивни ташкил қилиш учун

```
int **a;
```

кўринишидаги «кўрсаткичга кўрсаткич» ишлатилади.

Бошда массив сатрлари сонига қараб кўрсаткичлар массивига динамик хотирадан жой ажратиш керак:

```
a=new int *[m] // бу ерда m массив сатрлари сони
```

Кейин, ҳар бир сатр учун такрорлаш оператори ёрдамида хотира ажратиш ва уларнинг бошланғич адресларини а массив элементларига жойлаштириш зарур бўлади:

```
for(int i=0;i<m;i++)a[i]=new int[n];//n устунлар сони
```

Шуни қайд этиш керакки, динамик массивнинг ҳар бир сатри хотиранинг турли жойларида жойлашиши мумкин (7.1 ва 7.3-расмлар).

Икки ўлчамли массивни ўчиришда олдин массивнинг ҳар бир элементи (сатри), сўнгра массивнинг ўзи йўқотилади:

```
for(i=0;i<m;i++) delete[]a[i];
delete[]a;
```

Маъруза 18
Функция тушунчаси. Кўриниш соҳаси. Локал ва глобал узгарувчилар

Программа таъминотини яратиш амалда мураккаб жараён ҳисобланади. Программа тузувчи программа комплексини бир бутун-ликдаги ва унинг ҳар бир бўлагининг ички мазмунини ва уларнинг сезилмас фарқларини ҳисобга олиши керак бўлади.

Программалашга тизимли ёндошув шундан иборатки, програм-ма тузувчи олдига қўйилган масала олдиндан иккита, учта ва ундан ортиқ нисбатан кичик масала остиларга бўлинади. Ўз навбатида бу масала остилари ҳам яна кичик масала остиларига бўлиниши мумкин. Бу жараён токи майда масалаларни оддий стандарт амаллар ёрдамида ечиш мумкин бўлгунча давом этади. Шу йўл билан масалани декомпозициялаш амалга оширилади.

Иккинчи томондан, программалашда шундай ҳолатлар кузатила-дики, унда программанинг турли жойларида мазмунан бир хил алго-ритмларни бажаришга тўғри келади. Алгоритмнинг бу бўлаклари асосий ечилаётган масаладан ажратиб олинган қандайдир масала остини ечишга мўлжалланган бўлиб, етарлича мустақил қийматга (натижага) эгадир. Мисол учун қуйидаги масалани кўрайлик:

Берилган $a_0,a_1,...,a_{30}$, $b_0,b_1,...,b_{30}$, $c_0,c_1,...,c_{30}$ ва x,y,z ҳақиқий сонлар учун

$$\frac{\left(a_0x^{30}+a_1x^{29}+...+a_{30}\right)^2-\left(b_0y^{30}+b_1y^{29}+...+b_{30}\right)}{c_0(x+z)^{30}+c_1(x+z)^{29}+...+c_{30}}$$

ифоданинг қиймати ҳисоблансин.

Бу мисолни ечишда касрнинг сурат ва махражидаги ифодалар бир хил алгоритм билан ҳисобланади ва программада ҳар бир ифодани (масала ости) ҳисоблаш учун бу алгоритмни 3 марта ёзишга тўғри келади. Масаладаги 30-даражали кўпҳадни ҳисоблаш алгоритмини, масалан, Горнер алгоритмини алоҳида, битта нусхада ёзиб, унга турли параметрлар - бир сафар a вектор ва x қийматини, иккинчи сафар b вектор ва y қийматини, ҳамда c вектор ва (x+z) қийматлари билан мурожаат қилиш орқали асосий масалани ечиш мумкин бўлади. Функциялар қўлланишининг яна бир сабабини қуйидаги масалада кўришимиз мумкин - берилган чизиқли тенгламалар системасини Гаусс, Крамер, Зейдел усулларининг бирортаси билан ечиш талаб қилинсин. У ҳолда асосий программани қуйидаги бўлакларга бўлиш мақсадга мувофиқ бўлар эди: тенглама коэффицентларини киритиш бўлаги, ечиш усулини танлаш бўлаги, Гаусс, Крамер, Зейдел усулларини амалга ошириш учун алоҳида бўлаклар, натижани чоп қилиш бўлаги. Ҳар бир бўлак учун ўз функциялар мажмуаси яратиб, зарур бўлганда уларга бош функция танасидан мурожаатни амалга ошириш орқали бош масала ечиш самарали ҳисобланади.

Бундай ҳолларда программани ихчам ва самарали қилиш учун C++ тилида программа бўлагини алоҳида ажратиб олиб, уни функция кўринишида аниқлаш имкони мавжуд.

Функция бу - C++ тилида масала ечишдаги калит элементларидан биридир.

Функция параметрлари ва аргументлари

Программада ишлатиладиган ҳар қандай функция эълон қили-ниши керак. Одатда функциялар эълони сарлавҳа файлларда эълон қилинади ва #include директиваси ёрдамида программа матнига қўшилади.

Функция эълонини *функция прототипи* тавсифлайди (айрим ҳолларда *сигнатура* дейилади). Функция прототипи қуйидаги кўринишда бўлади:

<қайтарувчи қиймат тури> <функция номи>(<параметрлар рўйхати >);

Бу ерда <қайтарувчи қиймат тури> - функция ишлаши натижасида у томонидан қайтарадиган қийматнинг тури. Агар қайтариладиган қиймат тури кўрсатилмаган бўлса, келишув бўйича функция қайтара-диган қиймат тури int деб ҳисобланади, <параметрлар рўйхати>- вергул билан ажратилган функция параметрларининг тури ва номлари рўйхати. Параметр номини ёзмаса ҳам бўлади. Рўйхат бўш бўлиши ҳам мумкин. Функция прототипларига мисоллар:

```
int almashsin(int,int);
double max(double x,double y);
void func();
void chop_etish(void);
```

Функция прототипи тушириб қолдирилиши мумкин, агар прог-рамма матнида функция аниқланиши уни чақирадиган функциялар матнидан олдин ёзилган бўлса. Лекин бу ҳолат яхши услуб ҳисоб-ланмайди, айниқса ўзаро бир-бирига мурожаат қилувчи функциялар-ни эълон қилишда муаммолар юзага келиши мумкин.

Функция аниқланиши - функция сарлавҳаси ва фигурали қавсга (‘{‘,’}’) олинган қандайдир амалий мазмунга эга танадан иборат бўлади. Агар функция қайтарувчи тури void туридан фарқли бўлса, унинг танасида албатта мос турдаги параметрга эга return оператори бўлиши шарт. Функция танасида биттадан ортиқ return оператори бўлиши мумкин. Уларнинг ихтиёрий бирортасини бажариш орқали функциядан чиқиб кетилади. Агар функциянинг қиймати програм-мада ишлатилмайдиган бўлса, функциядан чиқиш учун параметрсиз return оператори ишлатилиши мумкин ёки умуман return ишлатил-майди. Охирги ҳолда функциядан чиқиш - охирги ёпилувчи қавсга етиб келганда рўй беради.

Функция программанинг бирорта модулида ягона равишда аниқланиши керак, унинг эълони эса функцияни ишлатадиган модулларда бир неча марта ёзилиши мумкин. Функция аниқланишида сарлавҳадаги барча параметрлар номлари ёзилиши шарт.

Одатда программада функция маълум бир ишни амалга ошириш учун чақирилади. Функцияга мурожаат қилганда, у қўйилган масала-ни ечади ва ўз ишини тугатишида қандайдир қийматни натижа сифатида қайтаради.

Функцияни чақириш учун унинг номи ва ундан кейин қавс ичида аргументлар рўйхати берилади:

<функция номи>(<аргумент$_1$>, <аргумент$_2$>,…, <аргумент$_n$ >);

Бу ерда ҳар бир <аргумент> - функция танасига узатиладиган ва кейинчалик ҳисоблаш жараёнида ишлатиладиган ўзгарувчи, ифода ёки ўзгармасдир. Аргументлар рўйхати бўш бўлиши мумкин.

Функциялар ҳам ўз танасида бошқа функцияларни, ўзини ҳам чақириши мумкин. Ўз танасида ўзини чақирадиган функцияларга *рекурсив функциялар* дейилади.

Олдинги бобларда таъкидлаб ўтилганидек, C++ тилидаги ҳар қандай программада албатта main() бош функцияси бўлиши керак. Айни шу функцияни юклагич томонидан чақирилиши билан програм-ма бажарилиши бошланади.

5.1- расмда бош функциядан бошқа функцияларни чақириш ва улардан қайтиш схемаси кўрсатилган.

Программа main() функциясини бажаришдан бошланади ва «f1(x,y);» - функция чақиришгача давом этади ва кейинчалик бошқарув f1(x,y) функция танасидаги амалларни бажаришга ўтади. Бунда Radius параметрининг қиймати сифатида функция x ўзгарувчи қийматини, Symbol параметри сифатида y ўзгарувчисининг қиймати ишлатилади.

Маъруза 19
Рекурсив функциялар

Юқорида қайд қилингандек *рекурсия* деб функция танасида шу функциянинг ўзини чақиришига айтилади. Рекурсия икки хил бўлади:

1) *оддий* - агар функция ўз танасида ўзини чақирса;
2) *воситали* - агар биринчи функция иккинчи функцияни чақирса, иккинчиси эса ўз навбатида биринчи функцияни чақирса.

Одатда рекурсия математикада кенг қўлланилади. Чунки аксарият математик формулалар рекурсив аниқланади. Мисол тариқасида факториални ҳисоблаш формуласини

$$n! = \begin{cases} 1, & \text{агар } n = 0; \\ n*(n-1)!, & \text{агар } n > 0, \end{cases}$$

ва соннинг бутун даражасини ҳисоблашни кўришимиз мумкин:

$$x^n = \begin{cases} 1, & \text{агар } n = 0; \\ x * x^{n-1}, & \text{агар } n > 0. \end{cases}$$

Кўриниб турибдики, навбатдаги қийматни ҳисоблаш учун функциянинг «олдинги қиймати» маълум бўлиши керак. C++ тилида рекурсия математикадаги рекурсияга ўхшаш. Буни юқоридаги мисоллар учун тузилган функцияларда кўриш мумкин. Факториал учун:

```
long F(int n)
{
 if(!n) return 1;
 else return n*F(n-1);
}
```

Берилган ҳақиқий x сонинг n- даражасини ҳисоблаш функцияси:

```
double Butun_Daraja(double x, int n)
{
 if(!n) return 1;
```

```
 else return x*Butun_Daraja(x,n-1);
}
```

Агар факториал функциясига n>0 қиймат берилса, қуйидаги ҳолат рўй беради: шарт операторининг else шохидаги қиймати (n қиймати) стекда эслаб қолинади. Ҳозирча қиймати номаълум n-1 факториални ҳисоблаш учун шу функциянинг ўзи n-1 қиймати билан билан чақирилади. Ўз навбатида, бу қиймат ҳам эслаб қолинади (стекка жойланади) ва яна функция чақирилади ва ҳакоза. Функция n=0 қиймат билан чақирилганда if операторининг шарти (!n) рост бўлади ва «return 1;» амали бажарилиб, айни шу чақириш бўйича 1 қиймати қайтарилади. Шундан кейин «тескари» жараён бошланади - стекда сақланган қийматлар кетма-кет олинади ва кўпайтирилади: охирги қиймат - аниқлангандан кейин (1), у ундан олдинги сақланган қийматга 1 қийматига кўпайтириб F(1) қиймати ҳисобланади, бу қиймат 2 қийматига кўпайтириш билан F(2) топилади ва ҳакоза. Жараён F(n) қийматини ҳисоблашгача «кўтарилиб» боради. Бу жараённи, n=4 учун факториал ҳисоблаш схемасини 5.2-расмда кўриш мумкин:

↓	F(4)=4*F(3)	↓	F(4)=4*F(3)	↓	F(4)=4*F(3)	↓	F(4)=4*F(3)	↑	F(4)=4*6
↓	F(3)=3*F(2)	↓	F(3)=3*F(2)	↓	F(3)=3*F(2)	↑	F(3)=3*2		
↓	F(2)=2*F(1)	↓	F(2)=2*F(1)	↑	F(2)=2*1				
↓	F(1)=1*F(0)	↑	F(1)=1*1						
↑	F(0)=1								

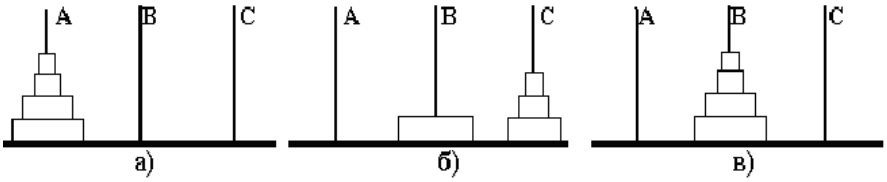
5.2-расм. 4! ҳисоблаш схемаси

Рекурсив функцияларни тўғри амал қилиши учун рекурсив чақиришларнинг тўхташ шарти бўлиши керак. Акс ҳолда рекурсия тўхтамаслиги ва ўз навбатида функция иши тугамаслиги мумкин. Факториал ҳисоблашида рекурсив тушишларнинг тўхташ шарти функция параметри n=0 бўлишидир (шарт операторининг рост шохи).

Ҳар бир рекурсив мурожаат қўшимча хотира талаб қилади - функцияларнинг локал объектлари (ўзгарувчилари) учун ҳар бир мурожаатда стекдан янгидан жой ажратилади. Масалан, рекурсив функцияга 100 марта мурожаат бўлса, жами 100 локал объектларнинг мажмуаси учун жой ажратилади. Айрим ҳолларда, яъни рекурсиялар сони етарлича катта бўлганда, стек ўлчами чекланганлиги сабабли (реал режимда 64Кб ўлчамгача) у тўлиб кетиши мумкин. Бу ҳолатда программа ўз ишини «Стек тўлиб кетди» хабари билан тўхтади.

Қуйида, рекурсия билан самарали ечиладиган «Ханой минораси» масаласини кўрайлик.

Масала. Учта A, B, C қозиқ ва n-та ҳар хил ўлчамли халқалар мавжуд. Халқаларни ўлчамлари ўсиш тартибида 1 дан n гача тартиб-ланган. Бошда барча халқалар A қозиққа 5.3а -расмдагидек жойлаш-тирилган. A қозиқдаги барча халқаларни B қозиққа, ёрдамчи C қозиқдан фойдаланган ҳолда, қуйидаги қоидаларга амал қилган ҳолда ўтказиш талаб этилади: халқаларни биттадан кўчириш керак ва катта ўлчамли халқани кичик ўлчамли халқа устига қўйиш мумкин эмас.



5.3-расм. Ханой минораси масаласини ечиш жараёни

Амаллар кетма-кетлигини чоп этадиган («Халқа q дан r га ўтказилсин» кўринишида, бунда q ва r - 5.3-расмдаги A,B ёки C халқалар). Берилган n та халқа учун масала ечилсин.

Кўрсатма: халқаларни A дан B га тўғри ўтказишда 5.3б –расмлар-даги ҳолат юзага келади, яъни n халқани A дан B ўтказиш масаласи n-1 халқани A дан C га ўтказиш, ҳамда битта халқани A дан B ўтказиш масаласига келади. Ундан кейин C қозиқдаги n-1 халқали A қозиқ ёрдамида B қозиққа ўтказиш масаласи юзага келади ва ҳакоза.

```
#include <iostream.h>
void Hanoy(int n,char a='A',char b='B',char c='C')
```

```
{
 if(n)
 {
  Hanoy(n-1,a,c,b);
  cout<<"Xalqa"<< a<<" dan "<<b<<" ga o'tkazilsin\n";
  Hanoy(n-1,c,b,a);
  }
}
int main()
{unsigned int Xalqalar_Soni;
 cout<<"Hanoy minorasi masalasi"<<endl;
 cout<<"Xalqalar sonini kiriting: ";
 cin>>Xalqalar_Soni;
 Hanoy(Xalqalar_Soni);
 return 0;
}
```

Халқалар сони 3 бўлганда (Xalqalar_Soni=3) программа экранга халқаларни кўчириш бўйича амаллар кетма-кетлигини чоп этади:

Маъруза 20
Сатрлар ва улар устида амаллар

C++ тилида стандарт сатр турига қўшимча сифатида string тури киритилган ва у string синфи кўринишида амалга оширилган. Бу турдаги сатр учун '\0' белгиси тугаш белгиси ҳисобланмайди ва у оддийгина белгилар массиви сифатида қаралади. string турида сатрлар узунлигининг бажарила-диган амаллар натижасида динамик равишда ўзгариб туриши, унинг таркибида бир қатор функциялар аниқланганлиги бу тур билан ишлашда маълум бир қулайликлар яратади.

string туридаги ўзгарувчилар қуйидагича эълон қилиниши мумкин:

```
string s1,s2,s3;
```

Бу турдаги сатрлар учун махсус амаллар ва функциялар аниқ-ланган.

string сатрга бошланғич қийматлар ҳар хил усуллар орқали бериш мумкин:

```
string s1="birinchi usul";
string s2("ikkinchi usul");
string s3(s2);
string s4=s2;
```

Худди шундай, string туридаги ўзгарувчилар устида қиймат бериш амаллари ҳам ҳар хил:

```
string s1,s2,s3; char *str="misol";
//сатрли ўзгармас қиймати бериш
s1="Qiymat berish 1-usul";
s2=str;        // char туридаги сатр юкланмоқда
s3='A';        // битта белги қиймат сифатида бериш
s3=s3+s1+s2+"0123abc"; //қиймат сифатида сатр ифода
```

8.2-жадвалида string туридаги сатрлар устидан амаллар келтирилган.

Сатр элементига индекс воситасидан ташқари at() функцияси орқали мурожаат қилиш мумкин:

```
string s1="satr misoli";
cout<<s.at(3) // натижада 'r' белгиси экранга чиқади
```

Шуни айтиб ўтиш керакки, string синфда шу турдаги ўзгарувчилар билан ишлайдиган функциялар аниқланган. Бошқача айтганда, string турида эълон қилинган

ўзгарувчилар (объектлар) ўз функцияларига эга ҳисобланади ва уларни чақириш учун олдин ўзгарувчи номи, кейин ‘.’ (нуқта) ва зарур функция номи (аргументлари билан) ёзилади.

8.2-жадвал. string туридаги сатрлар устидан амаллар

Амал	Мазмуни	Мисол
=, +=	Қиймат бериш амали	s=”satr01234” s+=”2satr000”
+	Сатрлар улаш амали (конкантенация)	s1+s2
==, !=, <, <=, >, >=	Сатрларни солиштириш амаллари	s1==s2 || s1>s2 && s1!=s2
[]	Индекс бериш	s[4]
<<	Оқимга чиқариш	cout << s
>>	Оқимдан ўқиш	cin >> s (пробелгача)

Сатр қисмини бошқа сатрга нусхалаш функцияси

Бир сатр қисмини бошқа сатрга юклаш учун куйидаги функция-ларни ишлатиш мумкин, уларни прототипи куйидагича:

```
assign(const string &str);
assign(const string &str,unsigned int pos,
       unsigned int n);
assign(const char *str, int n);
```

Биринчи функция қиймат бериш амал билан эквивалентдир: string туридаги str сатр ўзгарувчи ёки сатр ўзгармасни амални чақирувчи сатрга беради:

```
string s1,s2;
s1=”birinchi satr”;
s2.assign(s1); // s2=s1 амалга эквивалент
```

Иккинчи функция чақирувчи сатрга аргументдаги str сатрнинг pos ўрнидан n та белгидан иборат бўлган сатр қисмини нусхалайди. Агарда pos қиймати str сатр узунлигидан катта бўлса, хатолик ҳақида огоҳлантирилади, агар pos + n ифода қиймати str сатр узунлигидан катта бўлса, str сатрининг pos ўрнидан бошлаб сатр охиригача бўлган белгилар нусхаланади. Бу қоида барча функциялар учун тегишлидир.

Мисол:

```
string s1,s2,s3;
s1=”0123456789”;
s2.assign(s1,4,5);  // s2=”45678”
s3.assign(s1,2,20);      // s3=”23456789”
```

Учинчи функция аргументдаги char туридаги str сатрни string турига айлантириб, функцияни чақирувчи сатрга ўзлаштиради:

```
char * strold;
cin.getline(strold,100);//”0123456789” киритилади
string s1,s2;
s2.assign(strold,6);     // s2=”012345”
s3.assign(strold,20);    // s3=”0123456789”
```

Сатр қисмини бошқа сатрга қўшиш функцияси

Сатр қисмини бошқа сатрга қўшиш функциялари қуйидагича:

```
append(const string &str);
append(const string & str,unsigned int pos,
       unsigned int n);
append(const char *str, int n);
```

Бу функцияларни юқорида келтирилган мос assign функция-лардан фарқи - функцияни чақирувчи сатр охирига str сатрни ўзини ёки унинг қисмини қўшади.

```
char * sc;
cin.getline(sc,100);     //"0123456789" киритилади
string s1,s,s2;
s2=sc; s1="misol";
s="aaa";                  //s2="0123456789"
s2.append("abcdef"); //s2+="abcdef" амали
                          //ва s2="0123456789abcdef"
s1.append(s2,4,5);    //s1="misol45678"
s.append(ss,5);           // s="aaa012345"
```

Сатр қисмини бошқа сатр ичига жойлаштириш функцияси

Бир сатрга иккинчи сатр қисмини жойлаштириш учун қуйидаги функциялар ишлатилади:

```
insert(unsigned int pos1,const string &str);
insert(unsigned int pos1,const string & str,
       unsigned int pos2,unsigned int n);
insert(unsigned int pos1,const char *str, int n);
```

Бу фукциялар append каби ишлайди, фарқи шундаки, str сатрини ёки унинг қисмини функцияни чақирувчи сатрнинг кўрсатилган pos1 ўрнидан бошлаб жойлаштиради. Бунда амал чақирувчи сатрнинг pos1 ўриндан кейин жойлашган белгилар ўнга сурилади.

Маъруза 21
Структуралар

Маълумки, бирор предмет соҳасидаги масалани ечишда ундаги объектлар бир нечта, ҳар хил турдаги параметрлар билан аниқланиши мумкин. Масалан, текисликдаги нуқта ҳақиқий тур-даги х- абцисса ва у- ордината жуфтлиги - (х,у) кўринишида бери-лади. Талаба ҳақидаги маълумотлар: сатр туридаги талаба фамилия, исми ва шарифи, мутахассислик йўналиш, талаба яшаш адреси, бутун турдаги туғилган йили, ўқув босқичи, ҳақиқий турдаги рейтинг бали, мантиқий турдаги талаба жинси ҳақидаги маълумот ва бошқалардан шаклланади.

Программада ҳолат ёки тушунчани тавсифловчи ҳар бир берилганлар учун алоҳида ўзгарувчи аниқлаб масалани ечиш мумкин. Лекин бу ҳолда объект ҳақидаги маълумотлар «тарқоқ» бўлади, уларни қайта ишлаш мураккаблашади, объект ҳақидаги берилганларни яхлит ҳолда кўриш қийинлашади.

С++ тилида бир ёки ҳар хил турдаги берилганларни жамланмаси *структура* деб номланади. Структура фойдаланувчи томонидан аниқланган берилганларнинг янги тури ҳисобланади. Структура қуйидагича аниқланади:

struct <структура номи>
{
  <тур$_1$ > <ном$_1$>;
  <тур$_2$ > <ном$_2$>;
  …
  <тур$_n$> <ном$_n$>;
};

Бу ерда <структура номи> - структура кўринишида яратилаёт-ган янги турнинг номи, "<тур$_i$> <ном$_i$>;" - структуранинг і-майдо-нининг (ном$_i$) эълони.

Бошқача айтганда, структура эълон қилинган ўзгарувчилардан (майдонлардан) ташкил топади. Унга ҳар хил турдаги берилган-ларни ўз ичига олувчи *қобиқ* деб қараш

мумкин. Қобиқдаги берилганларни яхлит ҳолда кўчириш, ташқи қурилмалар (бинар файлларга) ёзиш, ўқиш мумкин бўлади.

Талаба ҳақидаги берилганларни ўз ичига олувчи структура тури-нинг эълон қилинишини кўрайлик.

```
struct Talaba
{
 char FISh[30];
 unsigned int Tug_yil;
 unsigned int Kurs;
 char Yunalish[50];
 float Reyting;
 unsigned char Jinsi[5];
 char Manzil[50];
 bool status;
};
```

Программада структуралардан фойдаланиш, шу турдаги ўзга-рувчилар эълон қилиш ва уларни қайта ишлаш орқали амалга оширилади:

```
Talaba talaba;
```

Структура турини эълонида турнинг номи бўлмаслиги мумкин, лекин бу ҳолда структура аниқланишидан кейин албатта ўзгарувчилар номлари ёзилиши керак:

```
 struct
 {
  unsigned int x,y;
  unsigned char Rang;
 } Nuqta1, Nuqta2;
```

Келтирилган мисолда структура туридаги Nuqta1, Nuqta2 ўзгарувчилари эълон қилинган.

Структура туридаги ўзгарувчилар билан ишлаш, унинг майдон-лари билан ишлашни англатади. Структура майдонига мурожаат қилиш ‘.’ (нуқта) орқали амалга оширилади. Бунда структура тури-даги ўзгарувчи номи, ундан кейин нуқта қўйилади ва майдон ўзгарув-чисининг номи ёзилади. Масалан, талаба ҳақидаги структура майдон-ларига мурожаат қуйидагича бўлади:

```
talaba.Kurs=2;
talaba.Tug_yil=1988;
strcpy(talaba.FISh,”Abdullaev A.A.”);
strcpy(talaba.Yunalish,
”Informatika va Axborot texnologiyalari”);
strcpy(talaba.Jinsi,”Erk”);
strcpy(talaba.Manzil,
”Toshkent,Yunusobod 6-3-8, tel: 224-45-78”);
talaba.Reyting=123.52;
```

Келтирилган мисолда talaba структурасининг сон туридаги майдонларига оддий кўринишда қийматлар берилган, сатр туридаги майдонлар учун strcpy функцияси орқали қиймат бериш амалга оширилган.

Структура туридаги объектнинг хотирадан қанча жой эгаллаган-лигини sizeof функцияси (оператори) орқали аниқлаш мумкин:

```
int i=sizeof(Talaba);
```

Айрим ҳолларда структура майдонлари ўлчамини битларда аниқлаш орқали эгалланадиган хотирани камайтириш мумкин. Бунинг учун структура майдони қуйидагича эълон қилинади:

<майдон номи> : <ўзгармас ифода>

Бу ерда <майдон номи>- майдон тури ва номи, <ўзгармас ифода>- майдоннинг битлардаги узунлиги. Майдон тури бутун турлар бўлиши керак (int, long, unsigned, char).

Агар фойдаланувчи структуранинг майдони фақат 0 ва 1 қийма-тини қабул қилишини билса, бу майдон учун бир бит жой ажратиши мумкин (бир байт ёки икки байт ўрнига). Хотирани тежаш эвазига майдон устида амал бажаришда разрядли арифметикани қўллаш зарур бўлади.

Мисол учун сана-вақт билан боғлиқ структурани яратишнинг иккита вариантини кўрайлик. Структура йил, ой, кун, соат, минут ва секунд майдонларидан иборат бўлсин ва уни қуйидагича аниқлаш мумкин:

```
struct Sana_vaqt
{
 unsigned short Yil;
 unsigned short Oy;
 unsigned short Kun;
 unsigned short Soat;
 unsigned short Minut;
 unsigned short Sekund;
};
```

Бундай аниқлашда Sana_vaqt структураси хотирада 6 майдон*2 байт=12 байт жой эгаллайди. Агар эътибор берилса структурада ортиқча жой эгалланган ҳолатлар мавжуд. Масалан, йил учун қиймати 0 сонидан 99 сонигача қиймат билан аниқланиши етарли (масалан, 2008 йилни 8 қиймати билан ифодалаш мумкин). Шунинг учун унга 2 байт эмас, балки 7 бит ажратиш етарли. Худди шундай ой учун 1..12 қийматларини ифодалашга 4 бит жой етарли ва ҳакоза.

Маъруза 22
Структура функция аргументи сифатида. Структуралар массифи

Структуралар функция аргументи сифатида ишлатилиши мумкин. Бунинг учун функция прототипида структура тури кўрсатилиши керак бўлади. Масалан, талаба ҳақидаги берилганларни ўз ичига олувчи Talaba структураси туридаги берилганларни Talaba_Manzili() функциясига параметр сифатида бериш учун функция прототипи қуйидаги кўринишда бўлиши керак:

```
void Talaba_Manzili(Talaba);
```

Функцияга структурани аргумент сифатида узатишга мисол сифатидаги программанинг матни:

```
#include <iostream.h>
#include <string.h>
struct Talaba
{
 char FISh[30];
 unsigned int Tug_yil;
 unsigned int Kurs;
 char Yunalish[50];
 float Reyting;
 unsigned char Jinsi[5];
 char Manzil[50];
 bool status;
};
void Talaba_Manzili(Talaba);
int main(int argc,char* argv[])
{
```

```
 Talaba talaba;
 talaba.Kurs=2;
 talaba.Tug_yil=1988;
 strcpy(talaba.FISh,"Abdullaev A.A.");
 strcpy(talaba.Yunalish,
 "Informatika va Axborot  texnologiyalari");
 strcpy(talaba.Jinsi,"Erk");
 strcpy(talaba.Manzil,
 "Toshkent, Yunusobod 6-3-8, tel: 224-45-78");
 talaba.Reyting=123.52;
 Talaba_Manzili(talaba);
 return 0;
}
void Talaba_Manzili(Talaba t)
{
 cout<<"Talaba FIO: "<<t.FIO<<endl;
 cout<<"Manzili: "<<t.Manzil<<endl;
}
```

Программа бош функциясида talaba структураси аниқланиб, унинг майдонларига қийматлар берилади. Кейин talaba структураси Talaba_Manzili() функциясига аргумент сифатида узатилади. Прог-рамма ишлаши натижасида экранга қуйидаги маълумотлар чоп этилади.

```
Talaba FIO: Abdullaev A.A.
Manzili: Toshkent, Yunusobod 6-3-8, tel: 224-45-78
```

Структуралар массиви

Ўз-ўзидан маълумки, структура туридаги ягона берилган билан ечиш мумкин бўлган масалалар доираси жуда тор ва аксарият ҳолатларда, қўйилган масала структуралар мажмуасини ишлатишни талаб қилади. Бу турдаги масалаларга берилганлар базасини қайта ишлаш масалалари деб қараш мумкин.

Структуралар массивини эълон қилиш худди стандарт массивларни эълон қилишдек, фарқи массив тури ўрнида фойдала-нувчи томонидан аниқланган структура турининг номи ёзилади. Масалан, талабалар ҳақидаги берилганларни ўз ичига олган массив яратиш эълони қуйидагича бўлади:

```
const int n=25;
Talaba talabalar[n];
```

Структуралар массивининг элементларига мурожаат одатдаги массив элементларига мурожаат усуллари орқали, ҳар бир элементнинг майдонларига мурожаат эса ‘.’ орқали амалга оширилади.

Қуйидаги программада гуруҳидаги ҳар бир талаба ҳақидаги берилганларни клавиатурадан киритиш ва гуруҳ талабаларини фамилия, исми ва шарифини чоп қилинади.

```
#include <iostream.h>
#include <conio.h>
const int n=3;
struct Talaba
{
 char FISh[30];
 unsigned int Tug_yil;
 unsigned int Kurs;
 char Yunalish[50];
 float Reyting;
 char Jinsi[6];
 char Manzil[50];
```

```
 bool status;
};
void Talaba_Kiritish(Talaba t[]);
void Talabalar_FISh(Talaba t[]);
int main(int argc, char* argv[])
{
 Talaba talabalar[n];
 Talaba_Kiritish(talabalar);
 Talabalar_FISh(talabalar);
 return 0;
}
void Talabalar_FISh(Talaba t[])
{
 for(int i=0; i<n; i++)
 cout<<t[i].FISh<<endl;
}
void Talaba_Kiritish(Talaba t[])
{
 for(int i=0; i<n; i++)
 {
  cout<<i+1<<"-talaba malumotlarini kiriting:"<<endl;
  cout<<" Talaba FISh :";
  cin.getline(t[i].FISh,30);
  cout<<" Kurs:";
  cin>>t[i].Kurs;
  cout<<" Reyting bali:";
  cin>>t[i].Reyting;cout<<" Tug''ilgan yili:";
  cin>>t[i].Tug_yil;
  cout<<" Ta'lim yo'nalishi:";
  cin.getline(t[i].Yunalish,50);
  cout<<" Jinsi(erkak,ayol):";
  cin.getline(t[i].Jinsi,6);
  cout<<" Yashash manzili:";
  cin.getline(t[i].Manzil,50);
 }
}
```

Маъруза 23
Структураларга курсаткич

Структура элементларига кўрсаткичлар орқали мурожаат қилиш мумкин. Бунинг учун структурага кўрсаткич ўзгарувчиси эълон қилиниши керак. Масалан, юқорида келтирилган мисолда Talaba структурасига кўрсаткич қуйидагича эълон қилинади:

```
Talaba * k_talaba;
```

Кўрсаткич орқали аниқланган структура элементларига мурожаат «.» билан эмас, балки «->» воситасида амалга оширилади:

```
cout<<k_talaba ->FISh;
```

Структураларни кўрсаткич ва адресни олиш (&) воситасида функция аргументи сифатида узатиш мумкин. Қуйида келтирилган программа бўлагида структурани Talaba_Kiritish() функциясига кўрсаткич орқали, Talabalar_FISh() функциясига эса адресни олиш воситасида узатишга мисол келтирилган.

```
...
void Talaba_Kiritish(Talaba *t);
```

```
void Talabalar_FISh(Talaba & t);
int main( )
{
 Talaba * k_talaba;
 k_talaba=(Talaba*)malloc(n*sizeof(Talaba));
 Talaba_Kiritish(k_talaba);
 Talabalar_FISh(*k_talaba);
 return 0;
}
void Talabalar_FISh(Talaba & t)
{
 for(int i=0; i<n; i++)
 {cout<<(&t+i)->FISh<<endl;}
}
void Talaba_Kiritish(Talaba *t)
{
 for(int i=0; i<n; i++)
 {
  cout<<i+1<<"-talaba malumotlarini kiriting:"<<endl;
  cout<<" Talaba FISh :";
  cin.getline((t+i)->FISh,30);
  cout<<" Kurs:";
  cin>>(t+i)->Kurs;
  ...
 }
}
```

Шунга эътибор бериш керакки, динамик равишда ҳосил қилинган структуралар массиви элементи бўлган структуранинг майдонига мурожаатда «*» белгиси қўлланилмайди.

Масала. Футбол жамоалари ҳақидаги маълумотлар - жамоа номи, айни пайтдаги ютуқлар, дуранг ва мағлубиятлар сонлари, ҳамда рақиб дарвозасига киритилган ва ўз дарвозасидан ўтказиб юборилган тўплар сонлари билан берилган. Футбол жамоаларининг турнир жадвали чоп қилинсин. Жамоаларни жадвалда тартиблашда қуйидаги қоидаларга амал қилинсин:

1) жамоалар тўплаган очколарини камайиши бўйича тартибла-ниши керак;

2) агар жамоалар тўплаган очколари тенг бўлса, улардан нисбатан кўп ғалабага эришган жамоа жадвалда юқори ўринни эгаллайди;

3) агар иккита жамоанинг тўплаган очколари ва ғалабалар сони тенг бўлса, улардан нисбатан кўп тўп киритган жамоа жадвалда юқори ўринни эгаллайди.

Жамоа ҳақидаги берилганлар структура кўринишида, жадвал эса структура массиви сифати аниқланади:

```
struct Jamoa
{
 string Nomi;
 int Yutuq, Durang, Maglub, Urgan_tup, Utkazgan_tup;
 int Uyin, Ochko;
};
```

Бу ерда Uyin майдони Yutuq, Durang ва Maglub майдонлар йиғиндиси, жамоа тўплаган очколар - Ochko=3*Yutuq+1*Durang кўринишида аниқланади. Жамоалар массиви Ochko, Yutuq ва Urgan_tup майдонлари бўйича тартибланади.

Маъруза 24
Динамик структура

Берилганлар устида ишлашда уларнинг миқдори қанча бўлиши ва уларга хотирадан қанча жой ажратиш кераклиги олдиндан номаълум бўлиши мумкин. Программа ишлаш пайтида берилганлар учун зарурат бўйича хотирадан жой ажратиш ва уларни кўрсаткичлар билан боғлаш орқали ягона структура ҳосил қилиш жараёни *хотиранинг динамик тақсимоти* дейилади. Бу усулда ҳосил бўлган берилганлар мажмуасига *берилганларнинг динамик структураси* дейилади, чунки уларнинг ўлчами программа бажарилишида ўзгариб туради. Программалашда динамик структуралардан чизиқли рўйхат-лар (занжирлар), стеклар, навбатлар ва бинар дарахтлар нисбатан кўп ишлатилади. Улар бир - биридан элементлар ўртасидаги боғланиш-лари ва улар устида бажариладиган амаллари билан фарқланади. Программа ишлашида структурага янги элементлар қўшилиши ёки ўчирилиши мумкин.

Ҳар қандай берилганларнинг динамик структураси майдонлар-дан ташкил топади ва уларнинг айримлари қўшни элементлар билан боғланиш учун хизмат қилади.

Масала. Нолдан фарқли бутун сонлардан иборат чизиқли рўйхат яратилсин ва ундан кўрсатилган сонга тенг элемент ўчирил-син.

Бутун сонларнинг чизиқли рўйхат кўринишидаги динамик структураси қуйидаги майдонлардан ташкил топади:

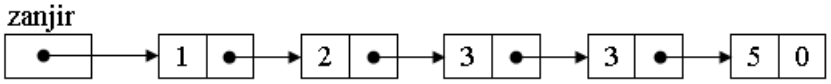
```
struct Zanjir
{
 int element;
 Zanjir * keyingi;
};
```

Программа матни:

```
#include <iostream.h>
struct Zanjir { int element; Zanjir * keyingi;};
Zanjir * Element_Joylash(Zanjir * z, int yangi_elem)
{
.......................
int main()
{
 Zanjir * zanjir=0;
 int son, del_element;
 do
 {
  cout<<"\nSonni kiriting (0-jaryon tugatish): ";
  cin>>son;
  if(son) zanjir=Element_Joylash(zanjir,son);
 } while (son);
 Zanjir_Ekranga(zanjir);
 cout<<"\nO'chiriladigan elementni kiriting: ";
 cin>>del_element;
 zanjir= Element_Uchirish(zanjir,del_element);
 Zanjir_Ekranga(zanjir);
 Zanjir = Zanjirni_Uchirish(zanjir);
 return 0;
}
```

Программанинг бош функциясида чизиқли рўйхат ҳосил қилиш учун Zanjir туридаги zanjir ўзгарувчиси аниқланган бўлиб, унга бўш кўрсаткич қиймати 0 берилган (унинг эквиваленти - NULL). Кейин такрорлаш оператори танасида клавиатурадан бутун сон ўқилади ва Element_Joylash() функциясини чақириш орқали бу сон рўйхатга охирига қўшилади. Функция янги ҳосил бўлган рўйхат бошининг адресини яна zanjir

ўзгарувчисига қайтаради. Агар клавиатурадан 0 сони киритилса рўйхатни ҳосил қилиш жараёни тугайди. Фараз қилайлик қуйидаги сонлар кетма-кетлиги киритилган бўлсин: 1,2,3,3,5,0. У ҳолда ҳосил бўлган рўйхат қуйидаги кўринишда бўлади (10.1-расм):



10.1-расм. Бешта сондан ташкил топган чизиқли рўйхат

Ҳосил бўлган рўйхатни кўриш учун Zanjir_Ekranga() функцияси чақирилади ва экранда рўйхат элементлари чоп этилади. Рўйхат устида амал сифатида берилган сон билан устма-уст тушадиган элементларни ўчириш масаласи қаралган. Бунинг учун ўчириладиган сон del_element ўзгарувчига ўқилади ва у Element_Uchirish() функция-си чақирилишида аргумент сифатида узатилади. Функция бу сон билан устма-уст тушадиган рўйхат элементларини ўчиради (агар бундай элемент мавжуд бўлса) ва ўзгарган рўйхат бошининг адреси-ни zanjir ўзгарувчисига қайтариб беради. Масалан, рўйхатдан 3 сони билан устма-уст тушадиган элементлар ўчирилгандан кейин у қуйидаги кўринишга эга бўлади (10.2-расм):

Маъруза 25
Бирлашма ва улар устида амаллар

Бирлашмалар хотиранинг битта соҳасида (битта адрес бўйича) ҳар хил турдаги бир нечта берилганларни сақлаш имконини беради.

Бирлашма эълони union калит сўзи, ундан кейин идентификатор ва блок ичида ҳар хил турдаги элементлар эълонидан иборат бўлади, масалан:

```
union Birlashma
 {
 int n;
 unsigned long N;
 char Satr[10];
 };
```

Бирлашманинг бу эълонида компилятор томонидан Birlashma учун унинг ичидаги энг кўп жой эгалловчи элементнинг - Satr сатрининг ўлчамида, яъни 10 байт жой ажратилади. Вақтнинг ҳар бир моментида бирлашмада, эълон қилинган майдонларнинг фақат биттасининг туридаги берилган мавжуд деб ҳисобланади. Юқоридаги мисол-да Birlashma устида амал бажарилишида унинг учун ажратилган хотирада ёки int туридаги n ёки unsigned long туридаги N ёки Satr сатр қиймати жойлашган деб ҳисобланади.

Бирлашма майдонларига худди структура майдонларига муро-жаат қилгандек ‘.’ орқали мурожаат қилинади.

Структуралардан фарқли равишда бирлашма эълонида фақат унинг биринчи элементига бошланғич қиймат бериш мумкин:

```
union Birlashma
{
 int n;
 unsigned long N;
 char Satr[10];
}
 birlashma={25};
```

Бу мисолда birlashma бирлашмасининг n майдони бошланғич қиймат олган ҳисобланади.

Бирлашма элементи сифатида структуралар келиши мумкин ва улар одатда берилганни «бўлакларга» ажратиш ёки «бўлаклардан» яхлит берилганни ҳосил қилиш

учун хизмат қилади. Мисол учун сўзни байтларга, байтларни тетрадаларга (4 битга) ажратиш ва қайтадан бирлаштириш мумкин.

Қуйида байтни катта ва кичик ярим байтларга ажратишда бирлашма ва структурадан фойдаланилган программани матни келти-рилган.

```
#include <iostream.h>
union BCD
{unsigned char bayt;
 struct
 {
  unsigned char lo:4;
   unsigned char hi:4;
  } bin;
} bcd;
int main()
{
 bcd.bayt=127;
 cout<<"\n Katta yarim bayt : "<<(int)bcd.bin.hi;
 cout<<"\n Kichik yarim bayt: "<<(int)bcd.bin.lo;
 return 0;
}
```

Программа бош функциясида bcd бирлашмасининг байт ўлчамида bayt майдонига 127 қиймати берилади ва унинг катта ва кичик ярим байтлари чоп этилади.

Программа ишлаши натижасида экранга қуйидаги натижалар чиқади:

```
Katta yarim bayt : 7
Kichik yarim bayt: 15
```

Масала. Ҳақиқий турдаги соннинг компьютер хотирасидаги ички кўринишини чоп қилиш. Ҳақиқий сон float турида деб ҳисобланади ва у хотирада 4 байт жой эгаллайди (1-иловага қаранг). Қўйилган масалани ечиш учун бирлашма хусусиятдан фойдаланилади, яъни хотиранинг битта адресига ҳақиқий сон ва белгилар массиви жойлаштирилади. Ҳақиқий сон хотирага ўқилиб, белгилар массивининг ҳар бир элементининг (байтининг) иккилик кўриниши чоп этилади.

Тест саволлари

1. **C++ тилида бутун константалар куйидаги форматларда булади:**
   a. 8, 10, 16
   b. 10
   c. 2, 8, 10, 16
   d. 2, 16
   e. 0..9
2. **Саккизлик санок системасида ёзилган узгармасни курсатинг**
   a. 0123
   b. 0x123
   c. 123
   d. 0789
   e. 1010101
3. **Хотирада 2 байт жой эгаллайдиган бутун турни курсатинг.**
   a. short int
   b. int
   c. long int
   d. unsigned int
   e. unsigned short int
4. **enum Hafta**
{dush=5, sesh, chorsh=0, paysh, juma, shanba=paysh-1, yaksh}
тур эълонида yaksh константаси киймати нимага тенг
булади?
   a. 1
   b. 6
   c. 7
   d. 0
   e. ноаник булади
5. **C++ тилининг киймати бутун сон булган турларининг**
руйхатини курсатинг.
   a. int, char, long, bool
   b. int, char, long, bool, float
   c. char, long, bool, byte
   d. float, double
   e. int, long, float, double
6. **Адресланувчи энг кичик маълумот бирлиги...**
   a. байт
   b. килобайт
   c. бит
   d. диск сектори
   e. параграф
7. **Санок системаларининг кайси бири замонавий компьютерда**
маълумотларни хотирадаги ички тасвирланиши учун
ишлатилади?
   a. 2 с/с
   b. 8 с/с
   c. 16 с/с
   d. 10 с/с
   e. 3 с/с
8. **Иккилик санок системасидаги 10011,101 сони 10 санок**
системасидаги кайси сонни ифодалайди?
   a. 19,625
   b. 19,25
   c. 45

d. 12,45
e. 15,625

9. Бир байтда ифодаланиши мумкин булган иккилик сонлар микдори канча?

a. 256
b. 512
c. 127
d. 255
e. 16

10. Бир килобайтда неча байт бор?

a. 1024
b. 256
c. 1000
d. 127
e. 512

11. 16 разрядли процессор платформасида C++ тилида int туридаги киймат учун хотирадан канча жой ажратилади?

a. 2 байт
b. 4 байт
c. 1 байт
d. 8 байт
e. Соннинг иккилик куринишидаги разряд улчамида

12. ЭХМ хотирасида хакикий сон кандай тасвирланади?

a. соннинг ишораси, мантиссаси ва тартиби куринишида
b. соннинг ишораси, бутун ва каср кисми куринишида
c. мантисса ва тартиб куринишида
d. соннинг ишораси,мантисса,тартиб ва тартиб ишораси билан
e. соннинг иккилик куринишида

13. Кандай сонлар устида яхлитлаш амали бажарилади?

a. бутун
b. бутун ва хакикий
c. хакикий
d. ихтиёрий турдаги
e. мусбат сонлар

14. Сонлар устида арифметик амаллар бажаришнинг кандай холларида сон тартибини ошиб кетиши руй беради?

a. Хакикий сон тартибининг киймати 2^k сонига тенг ёки катта булганда, бу ерда k - сон тартибини ёзилишидаги битлар сони
b. Сон тартиби нолга тенг булганда
c. Соннинг абсолют киймати тур учун чегаравий максимал кийматдан катта булганда.
d. Сон тартиби манфий булганда.
e. Сон тартибларини кушганда натижа нол булса.

15. ЭХМ хотирасида бутун манфий сон кандай тасвирланади?

a. мос мусбат соннинг иккилик куринишидаги битларкийматларини тескари кийматига алмаштириш вабирни кушишдан хосил булган сон куринишида
b. мусбат сон иккилик куринишидаги битлар кийматларинитескари кийматига алмаштирилган куринишида
c. соннинг ишора разрядига бир кийматини куйиш билан
d. мусбат сон иккилик куринишига бирни кушиш ва разряд кийматларини тескари кийматлар билан алмаштиришдан хосил булган сон куринишида.

16. Сузувчи нуктали сон хотирада кандай тасвирланади?

a. Соннинг ишораси, тартиби ва мантиссаси хакидаги маълумотларни уз ичига олувчи тузилма куринишида
b. Соннинг ишораси, тартиби, мантиссаси ва сузувчи нуктаурни хакидаги маълумотларни уз ичига олувчи тузилма куринишида

c. Экспоненциал шаклдаги сон куринишида
d. Манфий сонни тасвирлаш учун тескари коддан фойдаланган холдаги соннинг экспоненциал куринишида

17. ASCII жадвалининг кайси белгиларига 'бошкарув белгилари' дейилади?

a. 0-31 кодли белгиларга
b. 0-127 кодли белгиларга
c. 128-255 кодли белгиларга
d. Ракам ва лотин харфларидан бошка белгиларга
e. 0-255 кодли белгиларга

Программа синтаксиси ва семантикаси

18. C++ тилида ифода деб нимага айтилади?

a. ифода - операторлар, операндлар ва пунктация белгиларининг кетма-кетлиги булиб, компилятор томониданберилганлар устида маълум бир амалларни бажаришга курсатма хисобланади.
b. ифода - операторлар кетма-кетлиги булиб, компилятор томонидан берилганлар устида маълум бир амалларни бажаришга курсатма хисобланади.
c. ифода - буйруклар кетма-кетлиги булиб, компилятор томонидан берилганлар устида арифметик амалларни бажаришга курсатма хисобланади.
d. ифода - арифметик ва мантикий операторлар ва пунктация белгиларининг кетма-кетлиги.

19. Компьютер программаси бу...

a. Бирор масалани ечишга каратилган, машина ёки алгоритмик тилда ёзилган курсатмалар кетма-кетлиги
b. Ассемблер тилида ёзилган буйруклар кетма-кетлиги
c. Кирувчи ва чикувчи маълумотларнинг кодлаштирилган ёзуви
d. Алгоритмнинг график тасвири
e. Масала ечимига олиб келадиган машина буйрукларининг чекли кетма-кетлиги

20. Блок бу ?

a. " ва " белги оралигига олинган операторлар кетма-кетлиги булиб, у компилятор томонидан яхлит бироператор деб кабул килинади.
b. '(' ва ')' белги оралигига олинган операторлар кетма-кетлиги булиб, у компилятор томонидан яхлит бир оператор деб кабул килинади.
c. '/*' ва '*/' белги оралигига олинган операторлар кетма-кетлиги булиб, у компилятор томонидан яхлит бир оператор деб кабул килинади.
d. '[' ва ']' белги оралигига олинган операторлар кетма-кетлиги булиб, у компилятор томонидан яхлит бир оператор деб кабул килинади.

21. for (<ифода1>; <ифода2>;<ифода3>) учун нотугри тавсифни курсатинг

a. <ифода2> - такрорлаш санагичи вазифасини бажарувчи узгарувчисига бошлангич киймат беришга хизмат килади
b. <ифода2> - такрорлашни бажариш ёки йуклигини аниклаб берувчи мантикий ифода, агар шарт рост булса, такрорлаш давом этади, акс холда йук
c. <ифода3> - одатда такрорлаш санагичи кийматини ошириш(камайтириш) учун хизмат килади ёки бу ерда такрорлаш шартига таъсир килувчи бошка амаллар булиши мумкин.
d. <ифода1> - такрорлаш санагичи вазифасини бажарувчи узгарувчисига бошлангич киймат беришга хизмат килади
e. <ифода1> - одатда такрорлаш санагичи кийматини ошириш (камайтириш) учун хизмат килади ёки бу ерда такрорлаш шартига таъсир килувчи бошка амаллар булиши мумкин.

22. Алгоритмик тил алфавити деб ... айтилади.

a. ... берилган тил учун чекланган белгилар тупламига...
b. ... тил курилмаларини аниклаш коидалари тупламига...
c. ...тил курилмаларига мазмун бериш коидалари тизимига...

d. ... лотин харфлари ва араб ракамлари тупламига ...
e. ... тилдаги хизматчи сузлар тупламига...

23. Алгоритмик тил синтаксиси деб ... айтилади.

a. ...тил курилмаларини аниклаш коидалари тупламига...
b. ...берилган тил учун чекланган белгилар тупламига...
c. ...тил курилмаларига мазмун бериш коидалари тизимига...
d. ...тил операторларига мос машина кодини аниклаш коидалари тизимига...
e. ... тилдаги хизматчи сузлар тупламига...

24. Алгоритмик тил семантикаси деб ... айтилади.

a. ...тил курилмаларига мазмун бериш коидалари тизимига...
b. ...берилган тил учун чекланган белгилар тупламига...
c. ...тил курилмаларини аниклаш коидалари тупламига...
d. ...тил операторлари учун бериладиган изохларга...
e. ...тилдаги хизматчи сузлар тупламига...

25. Алгоритмик тил операторининг тугри таърифини курсатинг.

a. Оператор алгоритмик тилнинг жумласи булиб, берилганларни кайта-ишлашнинг етарлича тугалланган боскичини билдиради.
b. Оператор алгоритмик тилнинг жумласи булиб, унинг ердамида узгарувчиларга киймат берилади.
c. Оператор алгоритмик тилнинг жумласи булиб, узгарувчиларнинг турлари эълон килинади.
d. Оператор алгоритмик тилнинг жумласи булиб,унинг ёрдамида берилганлар клавиатурадан укилади ва экранга чикарилади.
e. Оператор алгоритмик тилнинг жумласи булиб, сон берилганларни кайта-ишлашнинг етарлича тугалланган боскичини билдиради.

26. Узгарувчи бу -

a. программа объекти булиб, у киймат килиш хоссасига эга. Бу кийматни узгарувчи программа ишлаш жараёнида кабул килади.
b. программа объекти булиб, у киймат килиш хоссасига эга. Бу кийматни узгарувчи программа ишлашидан олдин кабул килади.
c. программа объекти булиб, у киймат килиш хоссасига эга. Бу кийматни узгарувчи программа ишлаш жараёнида факат бир марта кабул килади.
d. программа объекти булиб, у киймат килиш хоссасига эга. Бу кийматни узгарувчи программа ишлаш жараёнида киймат бериш оператори оркали кабул килади.
e. программа объекти булиб, у киймат килиш хоссасига эга. Бу кийматни узгарувчи программа ишлаш жараёнида бошка узгарувчилардан кабул килади.

27. Куйидаги программадаги хато оператор курсатилсин (агар у булса).

```
int main( ) { const char n=255; int x,y;  cin>>x>>x; y=n*x; n+=1; cout<<y<<' '<<n;
  return 0; }
```

a. n+=1;
b. cin>>x>>x;
c. cout<<y<<' '<<n;
d. n+=1; чунки n киймати 255 сонидан ортиб кетади.
e. Хато йук.

28. Программадаги мазмуний хато курсатилсин.

```
int main()  { int x,y; nishon1: cin>>x>>y; if(x>y) goto nishon1; else  x*=y; goto nishon2;
x+=y; nishon2: ; cout<<x;     return 0; }
```

a. x+=y; оператори хеч бир холатда бажарилмайди.
b. if оператори ичида goto операторини ишлатиб булмайди.
c. nishon2: ; буш операторга утиш мумкин эмас.
d. cin>>x; оператори чексиз такрорланиб колади
e. Хато йук.

29. Программадаги мумкин булмаган амалларни курсатинг.

```
int main() {   char a; bool b=123; int j; float x=123.999; j=x; a=x;
                  j=cos(b); x=j/2; j=(int)x;  return 0; }
```

a. барча амаллар уринли
b. j=x; бутун узгарувчига хакикий сонни бериб булмайди.
c. a=x; белги туридаги узгарувчига хакикий сонни бериб булмайди.
d. j=cos(b); мантикий узгарувчи cos функция аргументи булмайди.
e. x=j/2; хакикий узгарувчига бутун сонни бериб булмайди.

Ифодалар, операторлар ва амаллар

30. int n,m=2; n=1; m+=n++ + ++n; cout<<n<<' '<<m;
Амаллар бажарилиши натижасида экранга нима чоп этилади?

a. 3 6
b. 2 6
c. 3 7
d. 3 5
e. 2 7

31. A байтнинг 5 разрядига 1 кийматини урнатиш учун кайси амални бажариш керак?

a. A | = 32
b. A | =16
c. A ^ = 16
d. A & = 32
e. A & =16

32. A байтнинг 5 разрядида 1 киймати урнатилган ёки йуклигини кандай аниклаш мумкин?

a. if(A & 32)
b. if(A | 16)
c. if(A ^ 16)
d. if(A & 128)
e. if(~A)

33. A байтнинг 4 разрядини карама-карши кийматга алмаштиришни курсатинг ?

a. A ^= 0x08
b. A |= 0x10
c. A ^= 0x20
d. A &= 0xFF
e. A &= 0x16

34. char a=32>>6<<6; cout<<(int)a; Экранга нима чоп этилади?

a. 0
b. 16
c. 1
d. 32
e. хеч нима чоп этилмайди

35. int Son=1, Summa=0;

```
switch (Son)
{
 case 1: Summa+=Son; case 2 : Summa+=2*Son;
 case 3 : Summa+=3*Son; break;
 case 4 : Summa+=4*Son; break;
 default : Summa+=1;
}
```

36. switch оператори бажарилгандан кейин Summa узгарувчисининг киймати нимага тенг?

a. 6
b. 10
c. 1
d. 0
e. 2

int x=10; if(x-=x-- -1) x++; else x+=2; x узгарувчи киймати нимага тенг булади?

a. 2
b. 11
c. 8
d. 1
e. 0

37. Ифодани киймати нимага тенг?

24/(3*4)-24/3/4+24/3*4

a. 32
b. 4
c. 12
d. 16
e. 2

38. Хисоблашда бажариладиган амаллар тартиби курсатилсин?

a / b + a % b * c

a. /, %, * , +
b. %, /, * ,+
c. %, / , + , *
d. / , + , %, *
e. /, * , % , +

39. Ифоданинг киймати нимага тенг?

12/5+125 %(4+3*7)/2

a. 2
b. 0
c. 7
d. 1
e. -2

40. 12345 сонидаги '3' ракамини ажратиб олишнинг тугри амали кайси жавобда курсатилган.

a. 12345 / 10 / 10 % 10
b. 12345 % 100 % 10 % 10
c. 12345 / 1000 % 10 % 10
d. 12345 % 10 % 10
e. 12345 / 10 / 100 % 10

41. Куйидаги шартли оператор бажарилиши натижасида ва Y=1, Z=2, U=3 булганда X узгарувчи кандай кийматни кабул килади?

if (Y>0) if (Z<0) X:=0; else if (U<0) X:=1; else X:=2; else X:=3;

a. 2
b. 3
c. 1
d. 0
e. X узгарувчи киймат кабул килмайди.

Программа бажарилиши

42. Куйидаги программа ишлаганда жавобга нима чикади?

```
int main() { float x=1.5, y=-2.6;      y=y+x*y;      if (y<x) goto nishon;      y=y-x*2;
  nishon: cout<<y; return 0; }
```

a. -7.5
b. 0.5
c. -2.82
d. -6.5
e. 2.83

43. Куйидаги программа учун a узгарувчига 5.3 ва b узгарувчига 6.2 сонларини киймат сифатида киритилганда программа нима чоп этади?

```
int main() {int a; float b; cin>>a; cin>>b; cout<<a<<' '<<b;     return 0; }
```

a. 5 0.3
b. 5 6

c. 5 6.2
d. 6 0.2
e. Истисно холати руй беради ва программа уз ишини тухтатади.

44. Куйидаги программа учун 1,2,3 сонларини киймат сифатида берилса жавобга нима чикади?

```
int main() { int a,b; cin>>a>>b>>a; cout<<a<<' '<<b<<' '<<a; return 0; }
```

a. 3 2 3
b. 3 3 3
c. 1 3 1
d. 2 1 3
e. 1 2 3

45. Программа натижасида нима чоп этилади?

```
int main() { bool b=128;cout<<b;return 0;}
```

a. 1
b. 0
c. false
d. 128
e. true

46. Куйидаги программа бажарилиши натижасида экранга нима чикади?

```
int main() { unsigned int n=65535;      n+=1;      cout<<n;       return 0; }
```

a. 0
b. 65536
c. 65535
d. 256
e. Хатолик хакида хабар чикади

47. Куйидаги программа бажарилишида экранга кандай сон чикади?

```
int main() {short int i=32767;  i+=1;  cout<<i<<endl;  cin>>i;  return 0; }
```

a. -32768
b. 32768
c. 0
d. Хатолик хакида хабар
e. 32767

48. Куйидаги программа бажарилишида экранга кандай сон чоп этилади?

```
int main() {short int S=0;  for(int I=1; I<=10; I++)  I+=1;  S+=I;  cout<<I<<' '<<S;
 return 0; }
```

a. 11 30
b. 10 30
c. Хатолик хакида хабар чикади
d. 12 42
e. Экранга натижа чикмайди, чунки программада такрорлаш операторн чексиз ишлайди.

49. Куйидаги программадаги хато оператор курсатилсин (агар у булса).

```
int main() {const char n=255;  int x,y;  cin>>x>>x;  y=n*x;  n+=1;  cout<<y<<' '<<n;
 return 0; }
```

a. n+=1;
b. cin>>x>>x;
c. cout<<y<<' '<<n;
d. n+=1; чунки n киймати 255 сонидан ортиб кетади.
e. Хато йук.

50. Программадаги мазмуний хато курсатилсин.

```
int main()  {int x,y; nishon1: cin>>x>>y;  if(x>y) goto nishon1;   else  x*=y; goto nishon2;
 x+=y;  nishon2: ;  cout<<x;  return 0; }
```

a. x+=y; оператори хеч бир холатда бажарилмайди.
b. if оператори ичида goto операторини ишлатиб булмайди.
c. nishon2: ; буш операторга утиш мумкин эмас.
d. cin>>x; оператори чексиз такрорланиб колади

e. Хато йук.

#Такрорлаш операторлари

51. for (<ифода1>; <ифода2>;<ифода3>) учун нотугри тавсифни курсатинг

a. <ифода2> - такрорлаш санагичи вазифасини бажарувчи узгарувчисига бошлангич киймат беришга хизмат килади
b. <ифода2> - такрорлашни бажариш ёки йуклигини аниклаб берувчи мантикий ифода, агар шарт рост булса, такрорлаш давом этади, акс холда йук
c. <ифода3> - одатда такрорлаш санагичи кийматини ошириш (камайтириш) учун хизмат килади ёки бу ерда такрорлаш шартига таъсир килувчи бошка амаллар булиши мумкин.
d. <ифода1> - такрорлаш санагичи вазифасини бажарувчи узгарувчисига бошлангич киймат беришга хизмат килади
e. <ифода1> - одатда такрорлаш санагичи кийматини ошириш (камайтириш) учун хизмат килади ёки бу ерда такрорлаш шартига таъсир килувчи бошка амаллар булиши мумкин.

52. int n=10; while(n-=1, n2=n*n, n>0); Кавс ичидаги кайси амал такрорлаш операторининг тухташ шарти хисобланади?

a. n>0
b. n-=1
c. n2=n*n
d. n-=1, n2=n*n
e. n2=n*n, n>0

53. Чексиз такрорлаш операторидан кейинги операторга кайси оператор ёрдамида чикиб кетиш мумкин?

a. break;
b. continue;
c. return;
d. switch
e. if

54. C++ тилида шарт олдин текширилувчи такрорлаш операторини курсатинг.

a. do..while
b. for()
c. while..do
d. if()
e. if()else

55. Куйидаги такрорлаш оператори неча марта ишлайди?

int main() { unsigned short int i=0; do i+=1; while(1); return 0; }

a. Чексиз
b. i киймати 65535 сонидан ошганда такрорлаш тухтайди
c. 0
d. i киймати 32767 сонидан ошганда такрорлаш тухтайди

56. Куйидаги программа бажарилишида экранга кандай сон чоп этилади?

int main() { short int S=0; for(int I=1; I<=10; I++) I+=1; S+=I; cout<<I<<' '<<S; return 0; }

a. 11 30
b. 10 30
c. Хатолик хакида хабар чикади
d. 12 42
e. Экранга натижа чикмайди, чунки программада такрорлаш операторн чексиз ишлайди.

57. Хадлар сони иккитадан кам булмаган ва нол билан тугайдиган натурал сонлар кетма-кетлиги йигиндисини хисоблаш учун кайси такрорлаш операторини куллаган маъкул?

a. do..while
b. for
c. while..do

d. Ихтиёрий бирортасини куллаш самарали
e. олдин for кейин while..do

58. for(int i=100; i>=-1; i--) такрорлаш оператори неча марта ишлайди?

a. 102
b. 100
c. 101
d. 0
e. 99

59. for(int i=1; i<=10; i+=3)i--; такрорлаш оператори неча марта ишлайди?

a. 5
b. 10
c. чексиз
d. умуман ишламайди
e. 4

60. void Funksiya(void); Бу катор ёзилиши тугрими ва унда нима ёзилган?

a. Ха, бу каторларда функция аникланиши ёзилган.
b. Ха, бу каторларда функция прототипи ёзилган.
c. Ха, функцияга мурожаат ёзилган.
d. Йук, бу каторларда функция аникланиши хато ёзилган.
e. Йук, функция прототипи хато ёзилган.

61. Funksiya(int n) return n*n; функция эълони тугри ёзилганми?

a. Тугри
b. Хато, функцияни аниклаш синтаксиси бузилган.
c. Хато, функция тури берилмаганлиги учун return операторини ишлатиб булмайди
d. Тугри, чунки функция параметри int турида
e. Тугри ёки йуклиги, функцияга мурожаатга боглик булади

62. Качон функциядан кайтиш оператори "return;" куринишда булади?

a. Агар функция void турида эълон килинган булса
b. Агар функция тури эълон килинмаган булса
c. Агар функция int турида эълон килинган булса
d. Агар функция float турида эълон килинган булса
e. Агар функция char турида эълон килинган булса

63. Func_1(1, false, '\n'); Func_1(2, false); Func_1(3);Func_1();
Мурожаатлар учун функциянинг тугри эълонини курсатинг.

a. void Func_1(int n=4,bool bayroq=true,char belgi='\t');
b. void Func_1(int n, bool bayroq=true, char belgi= '\t');
c. void Func_1(int n=4, bool bayroq, char belgi= '\t');
d. void Func_1(int n=4, bool bayroq=true, char belgi);
e. void Func_1(int n, bool bayroq, char belgi);

64. Локал узгарувчилар -

a. функция танасида эълон килинган узгарувчилар
b. main() функцияси танасида эълон килинган узгарувчилар
c. программада main() функциясидан фаркли бошка функция танасида эълон килинган узгарувчилар
d. функция эълонидаги параметрлар
e. программадаги барча статик узгарувчилар

65. Программада функциялардан ташкарида эълон килинган узгарувчиларга ... дейилади.

a. . . . глобал узгарувчилар . . .
b. . . . локал узгарувчилар . . .
c. . . . ташки узгарувчилар . . .
d. . . . статик узгарувчилар . . .
e. . . . динамик узгарувчилар . . .

66. int y; int main() {int x=2; y=4; cout<<fun(x)<<" "<<x; return 0; } int fun(int a) {int x=a+y; return x; }

Программа бажарилганда экранга нима чоп этилади?

a. 6 2
b. 2 4
c. 6 4
d. 2 2
e. 4 4

67. int x, y; int main() {x=2; y=4; cout<<fun(x)<<" "; cout<<x; return 0; } int fun(int a) {x=a+y; return x; } Программа бажарилганда экранга нима чоп этилади?

a. 6 6
b. 2 4
c. 6 4
d. 2 2
e. 4 4

68. int x; int main() { int x=1; int x=2; cout<<::x; return 0; }Программа бажарилганда экранга нима чоп этилади?

a. 0
b. 1
c. 2
d. 65535
e. -32768

69. Хато функция эълонини кўрсатинг:

a. void f1(int i){int k=i+2;cout <<k; return 0;}
b. void f1(int i){int k=i+2;cout << k; return;}
c. void f1(int i) {int k=i+2;cout <<k;}

70. Функция киймат кайтармаса унинг тури ... булиш керак.

a. void
b. int
c. float
d. void int

71. void f(int i, int x=2,int j=5){ cout<<i+j+k;} void main(){ int a=5; f(a,3,6); f(a,3);f(a);} Программа ишлаши натижасида экранга чоп этилади:

a. 14 13 12
b. 12 13 14
c. 11 13 12
d. 12 11 14

72. Кайси функция сарлавхаси тугри ёзилган

a. int f(int I,int x=2,int j=2,int k=3)
b. int f(int I,int x=2,int j,int k=3)
c. int f(int i=1,int x,int j,int k=3)

73. void f(int i, float x=2.0, int j=5); функцияга хато мурожаатни курсатинг

a. f(4, ,3)
b. f(4)
c. f(2,3.0)
d. f(5,3.0,4)

74. Функциядан чикиш ва киймат кайтариш учун...операторн ишлатилади.

a. return
b. break
c. exit
d. goto

75. ...калит сузи функция сарлавхасида, функция киймат кайтармаслигини билдиради ёки параметрлар йуклигини билдиради

a. void
b. int
c. float
d. extern

76. Функция ...компиляторга функция параметрларнинг сони, тури ва кетма-кетлигини текшириш учун ишлатилади.

a. сарлавхаси
b. эълони
c. аникланиши

77. Кайси калит сузи аник бир хотира синфини курсатмайди

a. auto
b. registr
c. extern
d. static
e. void

78. Блок ичида ёки функция параметрлар руйхатида эълон килинган узгарувчилар ... хотира синфига тегишли булади, агар алохида холат курсатилмаган булса.

a. Автоматик
b. Регистр
c. Ташки
d. Статик

79. ... хотира модификатори компиляторга курсатилган узгарувчи регистрга жойлаштиришни курсатади

a. registr
b. auto
c. extern
d. static

80. Локал узгарувчи функцияга кейинги киришда уз кийматини саклаб колиши учун ... синфида деб эълон килиниши керак

a. static
b. auto
c. extern

Рейтинг баллари тақсимоти. Ўзлаштириш фоизларига мос баллар оралиқлари

2010-11 ўқув йили учун бакалавриат 1-курс ТМИ йўналиши бўйича «Программалаш асослари» фанидан баҳолаш турлари бўйича баллар тақсимоти

Машғулот ҳажми (1- семестр учун)

№	Машгулотлар	Аудитория соатлари	Мустакил иш	Умумий вақт сарфи
1	Маъруза	32	60	92
2	Амалий машгулот	84	52	136
Жами		116	112	228

Машғулот ҳажми (2- семестр учун)

№	Машгулотлар	Аудитория соатлари	Мустакил иш	Умумий вақт сарфи
1	Маъруза	30	40	70
2	Амалий машгулот	82	60	142
Жами		112	100	212

Рейтинг балларининг тақсимланиши

№	Баҳолаш турлари	Машғулот тури	Назоратлар сони				Жами
			1	2	3	4	
1	ЖН	Амалий машғулотлар	8	8	8	8	32
2	ОН	Маъруза	12	12	-	-	24
3	ЯН	Маъруза	30		-	-	30
4	МТБ	Мустақил	7	7	-	-	14
ЖАМИ							100

Ўзлаштириш фойизларига мос баллар оралиқлари

№	Баҳолаш турлари	Жами бали	Баҳолаш учун баллар							
			“аъло”		“яхши”		“қониқарли”		“қониқарсиз”	
			мах	мин	Мах	мин	мах	мин	мах	мин
1	ЖН	32	46	39,6	39,1	32,7	32,2	25,8	25,3	0
2	ОН	24	24	20,6	20,4	17	16,8	13,4	13,2	0
3	ЯН	30	30	25,8	25,5	21,3	21	16,8	16,5	0

Назорат шакллари бўйича баллар тақсимоти ва уни баҳолаш механизмалари

1. Амалий машғулотлар бўйича

№	Баҳолаш тури	Ўтказиш шакли	Бажариш механизми	Максимал бал	Бажариш вақти	Изоҳ
1	ЖБ	Ёзма	2 та	Ҳар бир	Дарс	Топшириқлар ёзма

			мисолдан иборат топшириқ	мисолга 4 бал, топшириқ учун жами 8 бал	жараёнида ва мустақил вазифа сифатида	иш тариқасида бажарилиши ёки уйда бажарилиб топширилиши мумкин.

2. Маърузалар бўйича

№	Баҳолаш тури	Ўтказиш шакли	Бажариш механизми	Максимал бал	Бажариш вақти	Изоҳ
1	ОБ	Ёзма	3 та саволдан иборат вариантлар шаклида	Ҳар бир саволга 4 бал, вариант учун 12, жами 24 бал	Дарс жараёнида	

3. Мустақил ишлар бўйича (семестр учун)

№	Баҳолаш тури	Ўтказиш шакли	Бажариш механизми	Максимал бал	Бажариш вақти	Изоҳ
1	МТБ	Мустақил равишда	Берилган мавзу бўйича реферат (индивидуал)	Ҳар бир рефератга 7 бал, Макс.бал 14	Ҳар семестрнинг 20-хафтаси	компьютерда

4. Маъруза ва амалий кўникмалар бўйича (семестр учун)

№	Баҳолаш тури	Ўтказиш шакли	Бажариш механизми	Максимал бал	Бажариш вақти	Изоҳ
1	ЯН	Ёзма	30 та саволдан иборат топшириқ	Ҳар бир мисолга 1 бал, топшириқ учун жами 30 бал	Фан бўйича охирги дарс	Назорат компьютерда тест ёки қоғоз вариантлар кўринишида ўтказлиши мумкин

Программалаш асослари фанидан реферат мавзулари

1. С++ тилида сонларнинг ички кўриниши.
2. С++ тилининг кутубхоналари ва уларнинг программа тузишдаги аҳамияти.
3. С++ тилида берилганлар турлари ва уларнинг бошка программалаш тилларидан фарки.
4. Сонларнинг турли санок системаларида ифодаланиши.
5. Турли санок системаларида арифметик амалларни бажариш.
6. Мантикий ифодалар ва амаллар.
7. С++ тилида кўп ўлчовли динамик масссивлар.
8. С++ тилида массивлар ва кўрсаткичлар.
9. С++ тилида катта сонлар ва улар устида арифметик амаллар бажарувчи алгоритмлар.
10. Туб сонни топиш алгоритми.
11. ЭКУБ ва ЭКУК ни топиш алгоритми.
12. Берилган бутун сонни барча туб купайтувчиларини топиш алгоритми.
13. Берилган хакикий соннинг ракамлари йигиндисини топиш алгоритми.
14. С++ тилида 1 улчовли массивни тартиблаш алгоритмлари.
15. n та бутун соннинг ЭКУБ ва ЭКУК ини топиш алгоритми.
16. Купхадни купхадга купайтириш алгоритми.
17. Функциялар. Функция номига курсаткич.
18. С++ тилида string тури.
19. Динамик массивлар ва сатр турлари.
20. С++ тилида шифрлаш алгоритмлари.
21. С++ тили ва Паскал тили имкониятларини таққосланг.
22. С++ тилида тасодифий сонлар билан ишловчи алгоритмлар.
23. С++ тилида сонли усуллар алгоритмлари.(Нюътон, Зейдел ...)
24. С++ тилида бинар файллар билан ишлаш алгоритмлари.
25. С++ тилида икки каррали, уч каррали интегралларни ҳисоблаш алгоритмлари.
26. Структура, бирлашма ва уларнинг бошқа тиллардаги аналоглардан устунлиги ва камчиликлари.

Программалаш фани бўйича курс иши мавзулари

1. Бутун сонлар устида арифметик амаллар ургатувчи тринажёр программа тузинг.
2. Касрлар сонлар устида арифметик амаллар ургатувчи тринажёр программа тузинг.
3. Сонларнинг ЭКУК ва ЭКУБ ини топишни ургатувчи тринажёр программа тузинг.
4. Тригонометрик функцияларни ургатувчи тринажёр программа тузинг.
5. Квадрат тенгламани ечишни ургатувчи тринажёр программа тузинг.
6. Тенгламалар системасини ечишни ургатувчи тринажёр программа тузинг.
7. Бир санок системасидан иккинчи санок системасига утказишни ургатувчи тринажёр программа тузинг.
8. Берилган томонларига кура учбурчак турини аникловчи программа тузилсин.
9. Берилган координаталарига кура туртбурчак турини аникловчи программа тузилсин.

Малакавий битирув ишларининг мавзулари

1. "Талаба С" тизимида "Ўзлаштириш" тизим ойнасини амалга ошириш.

3. С++тилини ўргатувчи видео курс тизимларини яратиш (консол режими).

4. Хисоб-китоб натижаларини визуализациялаш

5. Минимал конфигуряцияли сунъий нейрон тўри ёрдамида жадвал кўринишидаги функция кийматларини аппроксимация килиш

6. Табиий тилларни математик моделлаштиришдаги кириш тили учун процессор "Талаба С" тизимидаги излашни амалга ошириш.

7. Икки ўлчовли мураккаб сохани дискретлаш программа таъминотини яратиш

8. Уч ўлчовли жисмнинг дискрет моделини тузиш жараёнини автоматлаштириш.

9.Академик лицейлар учун «Давомат» тизимини яратиш

10. "Талаба С" тизимининг "Берилганларни киритиш ва тахрирлаш"тизим остисини яратиш.

11. Температурани хисобга олгандаги паралапипеднинг мувозанати хакидаги масаланинг программа таъминоти.

12. Амалий масалаларни эчимларини визуаллаштириш.

13. Синфлардаги мантикий конуниятлар ва экстремал объектлар.

14. Профессор-ўкитувчиларнинг рейтинг бахосини аниклаш тизимининг берилганлар базасини шакллантириш.

15. С++ тилини ўргатувчи видео курс тизимларини яратиш (С++ Буилдер мухити).

16. Дискрет моделининг параметрларини минилизациялаштириш программа таъминоти.

17. "Талаба С" тизимнинг "Давомат"тизим остини яратиш.

18. С++ Builder мухитида математик формулаларининг аналитик куринишини хосил килиш моделини яратиш

Голоссарий

Байт
Компъютер хотираси ўлчови бирлиги.

Бит
Бир ёки ноль кабул қилувчи ўзгарувчи.

Процессор
Компьютер курилмаларини бошқарувчи ва берилганларни қайта ишловчи қурилма

Хотира сегменти
Оператив хотира қисми бўлиб хажми(ўлчами) ўзгарувчи бўлади.

Оператив хотира
Программа бажарилиши учун вақтинча фойдаланиладиган хотира бўлиб программа бажарилиш тезлигини аниқлайди.

Киритиш чикариш қурилмаси
Компьютер ва ташқи қурилмалар ўртасида маълумот алмашинувини таъминлаб берувчи қурилма.

Регистр
Хотиранинг тез мурожаат қилиш мумкин бўлган қисм бўлиб, хажми оператив хотирадан кичик бўлади.

Буфер
Киритиш - чикариш амаллари бажарилиш вақтида вақтинчалик берилганларни сақлаш учун ишлатладиган хотира қисми.

Таянч турлар
C++ тилида мавжуд турлар. Масалан int , float, double, char…

Структура
фойдаланувчи томонидан аниқланган таркибига бошқа турдаги майдонларни ўз ичига олган таркибли тур.

Машина буйруғи
Маълум қоида асосида микропроцессорга брор амалн бажариш учун мўлжалланган буйруқ.

O`zbekiston Milliy Universiteti

ТАКРОРЛАШ ОПЕРАТОРЛАРИ

Программалаш ва тармоқ технологиялари кафедраси
доц. Мадрахимов Ш.Ф.

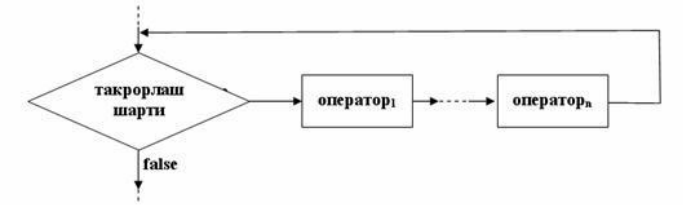
2011 йил

O`zbekiston Milliy Universiteti

ТАКРОРЛАШ ОПЕРАТОРИ

Программа бажарилишини бошқаришнинг бошқа бир кучли механизмларидан бири - такрорлаш операторлари ҳисобланади.

Такрорлаш оператори «такрорлаш шарти» деб номланувчи ифоданинг рост қийматида программанинг маълум бир қисмидаги операторларни (такрорлаш танасини) кўп марта такрор равишда бажаради (итаратив жараён)



O`zbekiston Milliy Universiteti

ТАКРОРЛАШ ОПЕРАТОРЛАРИ

Такрорлаш ўзининг кириш ва чиқиш нуқталарига эга, лекин чиқиш нуқтасининг бўлмаслиги мумкин. Бу ҳолда такрорлашга ***чексиз такрорлаш*** дейилади. Чексиз такрорлаш учун такрорлашни давом эттириш шарти доимо рост бўлади.

Такрорлаш шартини текшириш:

- такрорлаш танасидан олдин
  - ✓ for
  - ✓ while
- такрорлаш танасидан бир марта бажарилгач
  - ✓ do-while

O`zbekiston Milliy Universiteti

for такрорлаш оператори

for (<ифода$_1$>; <ифода$_2$>;<ифода$_3$>) <оператор>;

Бажаришдан қадамлари:

- **<ифода$_1$>** бажаришдан бошланади
- такрорлаш қадамлари бошланади
- **<ифода$_2$>** бажарилади
- такрорлаш танаси - **<оператор>** бажарилади
- **<ифода$_3$>** бажарилади
- агар **<ифода$_2$>** қиймати **false** бўлса, жараён тўхтайди
- агар **true** бўлса такрорланади

O`zbekiston Milliy Universiteti

while такрорлаш оператори

```
while (<ифода>) <оператор>;
```

Бажаришдан қадамлари:

- **<ифода> true** бўлса бажариш бошланади
- **<оператор>** бажарилади
- агар **<ифода>** қиймати **false** бўлса, жараён тўхтайди
- **акс ҳолда** яна такрорланади

O`zbekiston Milliy Universiteti

do-while такрорлаш оператори

```
do  <оператор>;  while (<ифода>);
```

Бажаришдан қадамлари:

- **<оператор>** бажарилади
- **<ифода> true** бўлса жараён тўхтайди
- **акс ҳолда** яна такрорланади

O`zbekiston Milliy Universiteti

Масала

Ҳар қандай 7дан катта бутун сондаги пул миқдорини 3 ва 5 сўмликларда бериш мумкинлиги исботлансин.

Ечиш усули

Қўйилган масала p=3n+5m тенгламани қаноатлантирувчи m, n сонлар жуфтликларини топиш масаласидир (p-пул миқдори). Бу тенглама бажарилиши учун m ва n ўзгарувчиларининг мумкин бўлган барча комбинациялари текширилиши зарур бўлади.

O`zbekiston Milliy Universiteti

Программа (асосий қисм)

```
#include <iostream.h>
int main()
{
unsigned int Pul; //Pul- киритиладиган пул миқдори
unsigned int n3, m5;//n-3 сўмликлар,m-5 сўмликлар сони
  ...
  ...
  return 0;
}
```

O`zbekiston Milliy Universiteti

Қиймат киритиш

```
...
  do
  {
    cout << "\nPul qiymatini kiriting (>8): ";
    cin >> Pul;
    if (Pul <= 7)
      cout << "Pul qiymati 8 dan kichik!";
  }
  while(Pul <= 7); // токи 7 катта сон киритилгунча
...
```

O`zbekiston Milliy Universiteti

n ва m ни ҳисоблаш

```
...
  n3 = 0; //бирорта ҳам 3 сўмлик йўқ
  do
  { m5 = 0; //бирорта ҳам 5 сўмлик йўқ
    do
    { if (3*n3+5*m5 == Pul)
        cout << n3 << " ta 3 so'mlik + "
             << m5 << " ta 5 so'mlik \n";
      m5++; // 5 сўмликлар биттага оширилади
    } while (3*n3+5*m5 <= Pul);
    n3++; //3 сўмликлар биттага оширилади
  } while(3*n3 <= Pul);
...
```

O`zbekiston Milliy Universiteti

Программа бажарилиши

```
#include <iostream.h>
int main()
{
  unsigned int Pul;
  //Pul- кир.ган пул миқдори
  unsigned int n3,m5;
  //n-3 сўмлар,m-5 сўмлар
  ...
  ...
  return 0;
}
```

Ҳотира

Pul	n3	m5	

Экран

O`zbekiston Milliy Universiteti

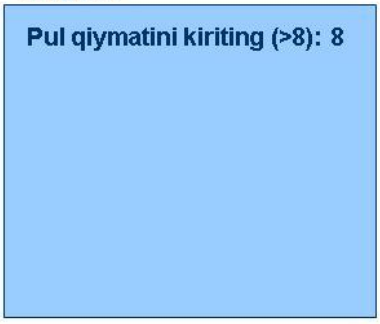
Программа бажарилиши

```
...
  do
  {
    cout << "\nPul qiymatini kiriting (>8): ";
    cin >> Pul;
    if (Pul <= 7)
      cout << "Pul qiymati 8 dan kichik!";
  }
  while(Pul <= 7);
  // токи 7 катта сон киритилгунча
...
```

Ҳотира

Pul	n3	m5	
8			

Экран



O`zbekiston Milliy Universiteti

Программа бажарилиши

```
...
  n3 = 0; //бирорта ҳам 3 сўмлик йўқ
  do
  { m5 = 0; //бирорта ҳам 5 сўмлик йўқ
    do
    { if (3*n3+5*m5 == Pul)
        cout << n3 << " ta 3 so'mlik + "
             << m5 << " ta 5 so'mlik \n";
      m5++; // 5 сўмлик бирга оширилади
    } while (3*n3+5*m5 <= Pul);
    n3++; //3 сўмликлар биттага оширилади
  } while(3*n3 <= Pul);
...
```

Ҳотира

Pul	n3	m5	
8	0	0	

Экран

```
Pul qiymatini kiriting (>8): 8
```

O`zbekiston Milliy Universiteti

Программа бажарилиши

```
...
  n3 = 0; //бирорта ҳам 3 сўмлик йўқ
  do
  { m5 = 0; //бирорта ҳам 5 сўмлик йўқ
    do
    { if (3*n3+5*m5 == Pul)
        cout << n3 << " ta 3 so'mlik + "
             << m5 << " ta 5 so'mlik \n";
      m5++; // 5 сўмлик бирга оширилади
    } while (3*n3+5*m5 <= Pul);
    n3++; //3 сўмликлар биттага оширилади
  } while(3*n3 <= Pul);
...
```

Ҳотира

Pul	n3	m5	
8	0	1	

Экран

```
Pul qiymatini kiriting (>8): 8
```

O`zbekiston Milliy Universiteti

Программа бажарилиши

```
...
  n3 = 0; //бирорта ҳам 3 сўмлик йўқ
  do
  { m5 = 0; //бирорта ҳам 5 сўмлик йўқ
    do
    { if (3*n3+5*m5 == Pul)
        cout << n3 << " ta 3 so'mlik + "
             << m5 << " ta 5 so'mlik \n";
      m5++; // 5 сўмлик бирга оширилади
    } while (3*n3+5*m5 <= Pul);
    n3++; //3 сўмликлар биттага оширилади
  } while(3*n3 <= Pul);
...
```

Ҳотира

Pul	n3	m5	
8	1	0	

Экран

Pul qiymatini kiriting (>8): 8
1 ta 3 so'mlik + 1 ta 5 so'mlik

O`zbekiston Milliy Universiteti

Программа бажарилиши

```
...
  n3 = 0; //бирорта ҳам 3 сўмлик йўқ
  do
  { m5 = 0; //бирорта ҳам 5 сўмлик йўқ
    do
    { if (3*n3+5*m5 == Pul)
        cout << n3 << " ta 3 so'mlik + "
             << m5 << " ta 5 so'mlik \n";
      m5++; // 5 сўмлик бирга оширилади
    } while (3*n3+5*m5 <= Pul);
    n3++; //3 сўмликлар биттага оширилади

...
```

Ҳотира

Pul	n3	m5	
8	1	2	

Экран

Pul qiymatini kiriting (>8): 8
1 ta 3 so'mlik + 1 ta 5 so'mlik

Фойдаланиладиган асосий дарсликлар ва ўқув қўлланмалар рўйхати

1. Б. Страуструп. Язык программирования С++. Специальное издание.-М.:ООО «Бином-Пресс», 2006.-1104 с.
2. Павловская Т.А. С++. Программирование на языке высокого уровня – СПб.: Питер. 2005.- 461 с.
3. Подбельский В.В. Язык СИ++.- М.; Финансы и статистика- 2003 562с.
4. Павловская Т.С. Щупак Ю.С. С/С++. Структурное программирование. Практикум.-СПб.: Питер,2002-240с
5. Павловская Т.С. Щупак Ю.С. С++. Объектно- ориентированное программирование. Практикум.-СПб.: Питер,2005-265с
6. Глушаков С.В., Коваль А.В., Смирнов С.В. Язык программирования С++: Учебный курс.- Харьков: Фолио; М.: ООО «Издательство АСТ», 2001.-500с.
7. Юров В., Хорошенко С. Assembler: Учебный курс- СПб, “Питер”,2000.-672с.
8. Финогенов К.Г. Основы языка Assemblera.-М.: Радио и связь, 2001. - 288 с.
9. Пильщиков В.Н. Упражнения по языку Паскаль-М.: МГУ, 1986.
10. Абель П. Assembler для IBM PC и программирования. 1991. М.: “Высшая школа”, 1992.- 447 с.
11. Скенлон Л. Персональный ЭВМ IBM PC и XT. Программирование на языке Assemblera. -М.: Радио и связь. 1991.- 336 с.
12. Гофман В. Э., Хомоненко А.Д. Delphi 5. - СПб.: БХВ-Санкт-Петербург, 2000. - 800с.
13. Немнюгин С.А. Turbo pascal, учебник. Изд. Питер., 2001, -496 с.
14. Абрамов С.А.,Гнезделова Капустина Е.Н.и др. Задачи по программированию. - М.: Наука, 1988.
15. Вирт Н. Алгоритмы + структуры данных = программа.-М.:Мир,1985.-405с.
16. Нортон П. Программно-аппаратная организация IBM PC.-М.:Мир,1991.-327с.
17. Юров В. Assembler: практикум. -СПб.: Питер, 2002.- 400с.
18. Informatika va programmalsh.O’quv qo’llanma. Mualliflar: A.A.Xaldjigitov, Sh.F.Madraximov, U.E.Adamboev, O’zMU, 2005 yil, 145 bet.
19. Мадрахимов Ш.Ф., Гайназаров С.М. С++ тилида программалаш асослари// Тошкент, ЎзМУ, 2009, 196 бет.