Кіріспе..........................................................................................................3

І. ТАРАУ. Объектілі бағдарлы программалау және технологиясы.

1. Объект ұғымы...............................................................7
2. ОБП қасиеттерi.................................................................................13
3. Әдістер.........................................................................20
4. Виртуалды әдістер........................................................23
5. Кластар құру...................................................................................... 24
6. Жиындар құру ................................................................................... 40

ІІ.  ТАРАУ.  HTML және ОБПтехнологиясы жайлы Web түйін құру

1. HTML ...................................................................................................45
   1. Тегтер..........……………..................……............………....….46
   2. HTML тілінің құрылымы ……………..........………..….…...48
   3. Cілтемелер мен якорьлар …...........................…….….…...…50

          2. 1.4  Бетке суреттер енгізу ..……………….....................…...……52

          2. 1. 5 Тізімдермен жұмыс …………………..............................….. 57

2. 1. 6 HTML-дің мультимедиялық объектілері...................... …...58

         2. 1. 7  Кесте құру …………….......................................................... 60

         2. 1. 8  Фреймдер  ………...….............................…...………….….. 62

2. 2   Front  Page программасы ………............………………….…..…... 63

         2. 2. 1 Стандартты аспаптар тақтасы .……...............................….. 64

1. 2. 2    Жұмыс режимі тақтасы …......................…………….….66

         2. 2. 3 Графикалық форматтардың түрлері …............................... .68

         2. 2. 4 Динамикалық  эффектілер..............................................…... 70

         2. 2. 5 Жүгіртпе жол…………...………........................…….…….. 70

         2. 2. 4  Бетке анимация ұйымдастыру........................................….. 71

2.3 ОБП жайлы Web түйіннің дайындалу технологиясы........................72

ҚОРЫТЫНДЫ ...........................................................................................79

Пайдаланылған әдебиеттер...................................................80

Қосымша...............................................................................81

Кіріспе

       Компьютердің құрылымына байланысты төменгі, ортаңғы,жоғары деңгейлі программалау тілдері болады.

        Төменгі деңгейлі тілдерге нақты  компьютрлік машиналық командалар тілі енеді. Машиналық тілдегі программа командалар тізбегі, яғни, операцияға қатысатын операндтардың адрестерімен орындалатын операцияның кодтарының тізбегі. Мұндай программалау тілдерін компьютерге тәуелді тілдер деп те атайды. Машиналық тілде жазылған программа басқа компьютерде орындалмауы мүмкін. Себебі бұл компьтердің командалар жүйесі мен құрылысы өзгеше болуы мүмкін. Машиналық тілде жазылған программа жылдам орындалады және жадтан аз орын бөледі. Машиналық тілде программалау көп уақыт жұмсауды қажет етеді.

       Машиналық тілдердің кемшіліктері ортаңғы деңгейлі машинаға бағдарланған. Машиналық тілде программалау кезінде жадты бөліктеу жұмысын программалаушы орындайды. Ал бағдарланған тілдерде  ол жұмысты компьютр атқарады. Нақты адрестермен оперция кодтары орнына символдар және бір командаға біріктірілген командалар тізбегі ретінде орындалады. Ортаңғы деңгейлі тілдерге автокодтар мен ассемблер тілі кіреді. Ассемблер автокодтарды машиналық тілге  аударады. Мұндай программалау тілдерін трансляторлар деп атайды. Ассемблер тілінде программалау копьютр құрылымының барлық мүмкіндіктерін қолдануға мүмкіндік береді.

        Жоғары деңгейлі программалау тілдері компьютрге тәуелсіз деп те атайды. Яғни, траслятор прогаммалау тілінде жазылған командаларды машиналық тілге аударады. Жоғары деңгейлі программалау тілде проблемалық- бағдарланған және универсады болып бөлінеді.

         Проблемаға бағдарланған тілде тек бір салада есептерді тиімді шешуге арналған. Мысалы, экономикалық есептерді, лисп, пролог- жасанды интеллект саласындағы есептерді шешуге арналған. Универсал программалау тілдері кез келген сипаттағы, кез келген саладағы есептерді шешуге мүмкіндік береді. Универсал программалау тілдері процедуралық және объектілі – бағдарлы болып екіге бөлінеді. 1950 жылы алғашқы процедураға бағдарланған Фортран программалау тілі пайда болды, ал 1960 жылы Алгол программалау тілі, кейін Basic, Pascal, Cu, APL т.с.с. программалау тілдері шығарыла бастады.

        Паскаль программалау тілі 1969 жылы жасалынған, оны Швейцария ғалымы Никлаус Вирт құрастырып шығарған. Алғашқы кездері тіл программалау негіздерін оқытуға арналып шығарылғаны уақыт өте келе кәсіби программалаушының арасында кеңінен қолданыс тапты. Қазіргі кезге дейін программалау тілі жетілдіріліп, оның көптеген нұсқалары пайда болды. Паскаль тілі программалауға ыңғайлы құрал болғандықтан қазіргі кездерде де көптеген мәселерді шешуге пайдаланады.

        Паскаль тілінің соңғы нұсқасын 1991 жылы Borland International компаниясы жасап шығарды.

         Паскаль тілі – компелятор.     Паскаль тілінде алфавит Бейсик тіліндегідей, мұнда программалау ортасын іске қосу үшін бірнеше фактор жиынтығы керек:

1. Turbo.exe – программаларды құру үшін интегралданған орта.
2. Turbo.hlp – көмекші мәліметтер орналасқан файл.
3. Turbo.tp – Turbo.exe файлы қолданатын конфигу- рациялық файл.
4. Turbo.tpl- Turbo Pascal-дің резидентті модульдері.
5. Tptour.exe- программалау ортасы алғаш іске қосылған кезде орта жайлы мәлімат беретін программа.
6. Graph.tpu – ортаның барлық графикалық программасын іске қосатын файл.

        Қазіргі кезде Паскаль тілінің Turbo Pascal нұсқасы қолданылады. Бұл ортаны іске қосу үшін Turbo.exe файлын орындау қажет. Файл орындауға жіберілгеннен соң программалау ортасы экранда ашылады. Экран үш бөліктен тұрады:

1. Мәзір қатары.
2. Программа мәтіні терілетін жұмыс облысы.
3. Көмекші мәлімет беретін қатар.

          Паскальда программа құрылымы :

1. Label – белгілемелер.
2. Const- тұрақтытылар.
3. Type – типтер.
4. Procedure, function – көмекші программалар.
5. Var- айнымалыларды сипаттау.
6. Begin……end – программаның негізгі бөлігі.

        Бұл дипломдық жұмыстың мақсаты ­­– Объектілі-бағдарлы программалаудың теориясын меңгеру, оның ішінде программалар құруды үйрену және Turbo Pascal-да мәзір арқылы ұйымдастыру. Бұл мақсатқа жету үшін келесі міндеттер орындалуы тиіс :

1. Объектілі-бағдарлы программалауға қажетті мәліметтер жинау;
2. Инкапсуляция, мұрагерлік, полиморфизмнің мағынала-рын түсіну;
3. Turbo Pascal-да файлдармен жұмыс жасау;
4. Тестік сұрақтарды енгізетін программа құрып үйрену және оны іске қосу;
5. Графикалық режимнің жұмысын игеру;
6. Turbo Pascal-дан мәзір программасының жұмысын игеру;
7. Turbo Pascal-да құрылған программаларды мәзір программасына кірістіру.

        Дипломдық жұмыс үш тараудан тұрады. Бірінші тарау объектілі  бағдарлы программалау негіздеріне, оның технологиясына  арналған. Екінші тарау WWW және HTML ұғымдарына, объектілі бағдарлы программалау технологиясы жайлы Web түйін құру технологияларына арналған.

        Web түйін  Front Page бағдарламасының көмегімен құрылған. Оның фреймдік, жүгіртпе жол жасау т.б. мүмкіндіктері пайдаланылған. Суреттер Photoshop – тың көмегімен дайындалған.

**I ТАРАУ. ОБЪЕКТІЛІ БАҒДАРЛЫ ПРОГРАММАЛАУ**

1. **Объект ұғымы**

**Объектілі бағдарлы программалаудың ең үздік технологиясы  және модификациясы жеңіл программа жасауға мүмкіндік береді. Оның мүмкіндіктерін ОБП тілінде қолдануға болады. Ондай тілдердің бірі Simula 67 , ол 60 – шы жылдардың екінші жартысында пайда болды. Қазіргі программалау тілдері ОБП-ны қолдайды. Оларға С++ , Java, Delphi т.б жатады. Бұл методология Turbo Pascal- дың 5.5 версиясынан бастап пайда болды. Мынаны айта кету керек, тек қана  ОБП тілдерінде емес басқа тілдерде де, мысалы, С және Фортран тілдерінде ОБП- ны қолданады. Бірақ оның жұмысын іске асыру өте қиынға түседі. Кейбір тапсырмалар класы дәстүрлі жолмен оңай шешіледі.**

        ОбП- ны пайдаланып, программаны өңдеу проектілі жүйенің объектілі моделін өңдеуден басталады. Ол үшін проблемаларды объектілерге бөліп, олардың арасындағы бір- біріне тәуелділігін ашады. Бұл шығармашылық процесс болашақ жүйені қарастырушы программалаушының біліктілігіне сүйенеді. Егер, мысалы, беттің кескіні туралы сөз етсек, онда бастапқы беттің жылтыры тегіс фрагменттердің жиынтығымен, үшбұрыштармен нақты көрінеді. Бұндай бөлінуді триангуляция деп атайды. Бұл жағдайда үшбұрыштар объект болып есептелінеді. Үшбұрыштар қабырғаларымен,бұрыштың өлшемдерімен анықталады. Бірақ олардың ұқсастықтары бар болғанымен үшбұрыштарды жалпы бір класқа біріктіруге мүмкіндік береді. Объектінің сипаттамасы оның атрибуттарының жиынтығы болып табылады. Оларды тұрақты және тұрақсыз атрибуттар деп екіге бөледі. Тұрақты атрибуттар оның класындағы объектілерді сипаттайды. Мысалы, көпбұрыштың төбелерінің санын айтуға болады. Ал, тұрақсыз атрибуттар объектінің жағдайын сипаттайды – осы атрибуттың мағынасын өзгерту арқылы, біз объектінің жағдайын өзгерте аламыз. Мысалы, көпбұрыштың төбелеріндегі бұрыштардың өзгеруін айтуға болады. Кейде атрибуттардың типтері көрсетіледі, себебі, әрқайсысы әртүрлі мағына береді және тұрақсыз атрибуттың бастапқы мағынасын көрсетеді. Осы атрибуттың бастапқы мағыналары объектінің бастапқы жағдайын көрсетеді. Мысалы, үшбұрышты алатын болсақ, оның атрибуттары болып қабырғалары болып есептеледі.

Объектілерді класқа біріктірудің белгілері болып тек қана атрибуттар ғана емес, сондай-ақ тапсырмалардың контекстісі де есептеледі, мысалы, тікбұрыштар және тікбұрышты үшбұрыштар бірдей атрибуттардың жиынтығымен сипаттала алады- іргелес екі қабырғалардың ұзындықтары. Солай бола тұра олар әртүрлі геометриялық қасиеттермен бірге әртүрлі математикалық объектілер бола алады.

        Объектілерді класқа біріктіру тапсырмаларды жалпы қойылым ретінде көрсетуге мүмкіндік береді. Осы кластың барлық объектілеріне қатысты өзіне тән аты болады. Сонымен қатар кластағы атрибуттардың аттары кейбір объектілерге байланысты қойылады. Класты сипаттау ‘’ жазу ‘’ түрін сипаттауды еске түсіреді, әр объект тұрақты немесе тұрақсыз типтің мағынасындай мағыны береді.

        ОБП- ның келесі негізгі түсінігі – операция. Операция дегеніміз берілген кластың объектісіне қолданатын функцияны айтамыз. Мысалы , үшбұрыштың геометриялық өзгерістері бір операцияны әртүрлі кластың объектісіне қолданады, оны полиморфтық операция деп атайды. Мысалы ,  “ вектор“ және “комплексті сан “ объектісіне көбейту операциясын қолдануға болады. Бұл полиморфтық операцияның мысалы болып табылады, себебі , векторды көбейту және комплексті сандарды көбейту - әртүрлі математикалық операциялар.

        Объект  берiлгендердiң сипатталынуы мен оларды өңдейтiн программаларды бiрiктiредi. Объектiге енетiн программалар :

Процедуралар;

Функциялар;

Конструкторлар;

Деструкторлар түрiнде iске асады.

|  |
| --- |
|  |
|  |  |

Объектке енгiзiлген программалар әдiстер деп аталынады және сонымен қатар:

Объект Turbo Pascal-да берiлгендердi сипаттау бөлiмiнде жазылады. Ал айнымалыларды сипаттау бөлiмiнде объект экземплярлары берiледi.

Type  Pozicia=Object

          X,Y:integer;

        Procedure Init(NewX,NewY:integer);

        Procedure Locat( var XL,YL:integer);

  End;

Объектi сипаттау үшiн Object кiлiттi сөзi қолданылады. Pozicia объектiсiне  берiлгендер ретiнде X және Y бѕтiн айнымалылары және әдiс ретiнде Init және Locate процедуралары енгiзiлген.  NewX, NewY, XL,YL  айнымалалары объектiнiң берiлгенi емес, Init және Locate процедураларының формальды параметрi болып табылыды. Әдiстердiң мәтiнi объектен тыс берiледi, сондықтан сипатталыну кезiнде қандай объектiге жататыны анықталынуы тиiс. Егер әдiстер әртүрлi объектiлерге тиiстi болса, онда олардың аттарын бiрдей етiп жаза беруiмiзге болады.Сондықтан әдiс атына оның қай объектiге жататынын көрсететiн префикс қосылып жазылады, мысал: Pozicia.Init, Circle.Init.  Әдiстермен жұмыс iстеу үшiн айнымалы енгiземiз:

Var

MyPoz:Pozicia;

Бұл сипатталынудан кейiн MyPoz.Init(18,12) деп жазуымызға болады, яғни MyPoz объект экземплярының Init процедурасын шақыру арқылы  объектiнi инициализациялау болып табылады, нәтижесiнде MyPoz объект экземплярының X,Y  айнымалылары 18 және 32 мәнiне ие болады. Әрбiр объект экземплярының барлық айнымалыларына меншiктi жады алаңы болады және  олар арасында мәндердiң алмасуы үнсiз жүргiзiлмейдi.   Объект айнымалаларымен тура байланысуға болады. Төменде көрсетiлген мысалда программадағы екi нұсқаданың қай қайсысының орындалу MyPoz.Init процедурасының шақырғандағы нәтиженi бередi.

1-шi нұсқа:

MyPoz.X:=18;

MyPoz.Y:=32;

2-шi нұсқа:

With MyPoz do

X:=18;

Y:-32;

End;

ОБП әдiстемесi берiлгендермен тура байланысуға кеңес бермейдi. Объект берiлгендерiн сол объектiнiң әдiсi арқылы өңдеген жөн.

ОБП- ны қолданып жазылған программа бір-бірімен қарым – қатынаста болатын объектіден тұрады. Объектіні программалық жүзеге асыру- олардың өңдеу процедуралары мен берілгендерін біріктіру деген сөз. Турбо Паскальда Object типі бар, оны record құрылымды типінің жалпыламасы деп есептеуге болады. Тұрақсыз объек типін объект экземпляры деп атайды.

        Объект типінің жазу типінен айырмашылығы тек қана берілгендерді сипаттайтын бөлімі ғана бар емес, тағы да объектіде сипатталатын процедуралар мен функциялар бар. Бұл процедуралар мен функциялар әдіс болып табылады.Объектінің әдістеріне олардың бөлімдері қатысты. Әдістері мен олардың параметрлері объектіні сипаттауда анықталады, ал оларды жүзеге асыру басқа жерде, яғни, программаның сол әдісті шақыру женрінде орындалады. Объектіні сипаттау әдіске қарайтын шаблондардан тұрады. Олар параметрлермен типтердің сандарының сәйкес екендігі компелятор үші н қажет.

              Type

                      Location=obgect

                            x,y:integer;

                            Procedure init( initX, InitY: integer);

                           Function getX: integer;

                           Function getY: integer;

                           End;

Обьект резервтегі object\_end сөздерінің көмегімен сипатталады. Олардың арасында әдістер мен  бөлімдердің сипаттамасы бар. Келтірілген мысалда объект графикалық координаталардың белгілерін сақтау үшін екі бөлімнен, берілген объектілердің әдістері сипатталған процедура мен екі функциялардан құралған. Объекті типтердің сипатталауының басқа сипатталудан айырмашылығы – ол программалық бірліктің  (программа, модуль) ең жоғарғы сатысында тұрады. Әдістер мен функциялар, процедуралардың сипатталу бөлімінде мұндай сипаттау болмайды. Резевтегі private сөзі объект бөлімдеріне кіре алмайды. Келесі мысалда X,Y айнымалысына location объектілі типінің getX, getY әдістері арқылы ғана кіреді. Мысалы:

 Type

     Location= object

          Procedure init(initX, initY : integer);

          Function getX: integer;

          Function getY : integer;

     Private      X, Y : integer;

     End;

Private секциясында объект әдістері де бола алады. Әдістерді толық сипаттау, яғни, оларды жүзеге асыру сипатталуы объектіні сипаттаудан кейін тұруы керек. Әдістердің аттары құрамалы және нүкте арқылы бөлінетін әдістердің аттары мен

объектілердің аттарынан тұрады .    Мысалы :

          Procedure Location.init(initX, initY : integer);

Begin

         X:= initX;

         Y:= initY;

End;

Function Location. getX: integer;

Begin

       GetX:=X;

End;

Function Location.getY : integer;

Begin

GetY := Y;

End;

        Объект сипатталып болғаннан кейін прграммада оның экземплярын қолдануға болады. Яғни, берілген объектілі типтерінің айнымалысын :

     Var

         Grmarker : location;

**1.2 ОБП қасиеттерi**

ОБП инкапсуляция, мұра, полиморфизм деген қасиеттерден  тұрады.

**Инкапсуляция** – дегеніміз ол объектінің өзіне қолданылатын қажетті детальді жасыру.инкапсуляция тек өзіне тән әдістері арқылы ғана берілгендерге кіруді ұйымдастыратын бірден бір тәсіл. *Инкасуляция-* жаңа объект түрiн өңдеудегi  берiлгендердiң процедура және функциялармен бiрiгуi болып табылады.  *"*Инкапсуляция"  терминi- бұл жалпы бiр "қабыққа" (Object ... End) берiлгендер мен әдiстердi бiрiктiру мағанасын бередi.

 Жоғарыда айтылғандай, ОБП әдiстемесiмен келiсе отырып, берiлгендердiң мәндерiн өзгерту мақсатында, берiлгендермен тура байланысуға болмайды. Объект берiлгендерi әдiстер арқылы өңделуi тиiст. Бұл инкапсуляция болып табылады. Private cипаттауы арқылы берiлгендер мен әдiстердiң бiр бөлiгiн объектен тыс пайдалануға мүмкiншiлiк туғызбауға болады.

  Type

   NewObject=object

   Public

     Берiлгендердi сипаттау{ашық бөлiгi}

     Әдiстердi сипаттау {ашық бөлiгi}

   Private

     Берiлгендердi сипаттау{жабық бөлiгi}

     Әдiстердi сипаттау {жабық бөлiгi}

   End;

Жабық берiлгендердi тек қана осы объектiнiң әдiстерi ғана өңдей алады алады.

Инкапсуляция принципі бойынша объект және оған орындалатын іс-әрекеттер біріктіріліп объектінің ерекше бір түрінде қарастырылады. Яғни, әрбір объект және оған орындалатын іс-әрекеттер жеке класс ретінде ұйымдастырылады.

        Біздің мысалда объектіні сипаттау init-инициялизациясының процедурасы және getX, getY функциялары бөлек өзінше объект бола алмайды. Ол - location объекті типінің бөлінбейтін бөлігі. Егер программада берілген типтердің бірнеше айнымалыларының сипаттамасы келтірілген болса, онда әр айнымалы берілгендерді сақтау үшін өзінің есте сақтау облысы болады, ал функциялар мен процедураларға кіретін нүктелердің нұсқаулары үшін жалпы есте сақтау облысы болады. Әр әдісті шақыру мүмкіндігі тек қана өзінің құрама атының көмегімен ғана орындалады.

        Өзінің бұрынғы қасиеттерін пайдаланып, дамыта және өзгерте отырып, мұрагерлік жаңа объектілерді  анықтауға мүмкіндік береді.

**Мұрагерлік принцип-** белгілі бір объектіден келесі бір объект туындап, туынды объекті бастапқы объектінің барлық қасиеттеріне ие болуы. Сонымен қатар туынды объект бастапқы объектіде болмайтын жаңа қасиетке ие болуы да мүмкін.  Туынды объект түбірлік объектіден барлық өрістер мен  әдістерді қабылдап және оған өзінің өрістері мен әдістерін қосады немесе түбірлік объектінің әдістерін өзінікіне айырбастай алады. Туанды объектінің сипаттамасы мына мысалда келтірілген.

*Мұра* -бағынышты объектiлерден тұратын иерархиялық құрылымды құрайтын бағынышты объектiнi анықтау.  Әрбiр бағынышты объект ататегi объектiсiнiң берiлгендерiне, процедура мен функцияларына тiкелей байланыса алады.

Ататегi объектiсiнiң атын туылатын жаңа объектiнiң сипатталуына жазу арқылы *мұра*қалдырылынады. Осылай ағаш тәрiздi құрылым құрылады. Мұра-бұл объектiлер емес, объект түрлерi арасындағы байланыс болып табылады. Мысалы:

Type Pozicia=Object {Ата-ана объектiсi}

X,Y:Integer;

Procedure Init(NewX,NewY:integer);

Procedure Locate(Var XN,XL:Integer);

End;

Kom=Object(Pozicia)  {Бағынышты объект}

      Cvet,Raz:Word;

       Procedure Init(NewX,NewY:Integer;Color:Word);

       Procedure Zoom(Delt,Znak:Integer);

End;

Pozicia түрiне бағынышты  Kom түрi қабырғасы Raz (пиксел бойынша)   болып келген  боялған квадратты бередi. Мұнда мұрадан қалған X,Y  координаттары квадрат центiрiнiң координатасын бередi. Zoom процедурасы квадратты Delt және Znak аргументерi бойынша үлкейтедi (кiшiрейтедi).

**Әдiстердi мұра** **ету.**Жаңа алаңдарды сипатаған кезде, алаң аттарының қайталанып жазылмайтындылығын қадағалау керек. Объектiдегi жаңа алаң аты мұра арқылы алынған алаң атымен сәйкес болған жағдай программада қате туғызады. Ал әдiстер басқаша болып келедi. Объект (бағынышты объект) күрделi болған сайын, ол өзiн өрнектейтiн жаңа әдiстердi талап етiп ғана қоймай, ескi әдiстердi өзгеруiн талап етуi, оның "жүрiс-тұрысыда" күрделене түседi. Kom.Init  локальды әдiс Pozicia.Init  ата-ата әдiсiн жабады. Егер Kom.Init әдiсi болмаса, онда Pozicia.Init Kom түрi үшiн глобальды болып табылыды және  автоматты түрде осы әдiске программа назар аударады.

Функциональды ұқсас әдiстердiң бiрдей аталынуы объектiлер жүйесiн қабылдау мен программалауды жеңiлдетедi.

Мысалы :

Type

     Point= object(location)

          Visible: boolean;

          Procedure init(initX, initY: integer);

          Procedure show;

          Procedure hide;

          Function isvisible: boolean;

          Procedure moveto(newX,newY: integer);

      End;

        Бұл жерде point объектісі туынды объекті болады, ал түбірлік объект ретінде location объектісі болады. Туынды объектіде түбірлік объектінің өрістері мен әдістерінің сипаттамасы жоқ. Соңғысының ( түбірлік объектінің) аты дөңгелек жақшаның ішінде object сөзінен кейін көрсетілген. Туынды объект  әдісінен  түбірлік объект әдісін шақыруға болады. Туынды объект жасау үшін түбірлік объектінің алғашқы текстісінің болуын  қажет етпейді. Түбірлік объект трансляциядан өткен модульдің құрамында болуы мүмкін.

        Турбо Паскальда inherited деген сөз резервте болады. Ол түбірлік объект әдісін шақыруға қолданылады. Туынды объект сипаттамасында сәйкес келетін сөйлем орналасқан, мысалы:

     constructor nunfield.init(x.coord, y.coord, len:integer;

                                           lmin, lmax: longint);

        begin

            inherited.init(x\_coord, y\_coord,len);

                     min:=lmin;

                     max:=lmax;

         end;

        Инициализация дегеніміз массивті нөлдермен толтыру, бұл үшін fillchar процедурасы қолданылады. Біріші параметр белгіні өзіне иеленетін берілгендердің объектісін береді. Ал екінші параметр үшінші параметр арқылы берілген белгі кіргізілетін жадтағы аралас байттың саны. Бұл процедураны шақыру кезінде кіріктірілген Турбо Паскальдың self тұрақтысы қолданылады. Ол экземпляр объектісіне анық емес нұсқағыш параметрі болып табылады. Инициялизация кезінде массивтердің аты мен типі анықталынады. Массивтің құрылуы кезінде массив өлшемі шектен тыс үлкен болмауын және типтің дұрыс көрсетілуін қадағалайтын тексерулер жүргізіледі. Содан кейін массивке жадтан орын бөледі және олар нөлге айналады. Массивтермен жұмыс аяқталғаннан кейін close әдісі шақырылады.

**Полиморфизм** – туынды объектілердің бастапқы объектілер әдістерін басқа есептерді шешуге қолданылуы. Обект қасиеттеріне байланысты әдістер әртүрл іс- әрекеттер жасайды, осы кезде полиморфизм көрінеді. Полиморфизм виртуальды  әдістермен және соңғы байланыс механизмімен байланысты. Дәл осы ситуацияда қандай операция орындалатындығы жөніндегі шешім программа орындалу кезінде қабылданады.

**Полиморфизм**("көпкөрiнiстi") класты немесе әр кластың өзiнiң функциональды ролi бар бiрнеше кластарды туысқан объектiлерге анықтауды ұсынады. Күрделi әдiстi жүрiстерi әр түрлi объектiнiң әр түрiне беру керек болатын тууы мүмкiн. Бұл жағдайда объектiлерге жалпы бiр әдiс қолданып, айырмашылықтарын ауыспалы бағынышты әдiстерге жүктеген жөн болып табылады. Бұл идея конструктивтi полиморфизм деп аталады. Бұл идея виртуальды бағынышты әдiстер арқылы iске асады. Объектiнi сипаттау барысында әрбiр "ауыспалы"  әдiстiң атынын кейiн Virtual  сөзi жазылуы керек. Ал оларды қосу үшiн конструкторге назар аударуымыз керек. **Конструктор-**бұл процедураның бiр түрi болып табылады және объектiнi сипаттау барысында Constructor сөзi процедура алдына жазылады. Бұл түсiнiктердi жақсы игеру үшiн мысал қарастырылған.

Қарастырылатын мысалда объектiлер ретiнде геометриялық фигуралар алынған. Кадарларды ауыстыра отырып "*көрiнiстi жасыру->позициясын өзгерту->жаңа орында пайда болу*" принципы бойынша фигураларды (хаосты) ұшыратын Fly  әдiсi объектiлермен жұмыс iстейдi. Мұнда көрсету (Show) және жасыру(Hide) фигураның әр түрiне арнайы арналған болып табылады. Бұлар виртуальды әдiстерге "кандидат" болады. Бұл мысалда, негiзгi ой- өлшемiн, түсiн және пропорциясына дейiн өзгерiп отыратын "ұшатын" фигуралар жиынын құру болып табылады.

**Абстрактылы ата-аналық түр**

Бастапқыда 3 түр алынған: Point("нүкте"), Circle( дөңгелек), Ring("Сақина"), бiрақ программа көрсетiлген кескiн бойынша  құрылған.

Point объектiсi қолданылмағандықтан, оның конструкторы жоқ. Circle түрiнiң конструкторы туылу блогiне қосалқы  жазылған. Басқа жағдай үшiн Ring түрiнде (қосалқы емес) конструктор  қарастырылған. Ол "бос " болып табылады. Виртуальды әдiстерi бар әр объектiге оның әдiстерiн  қолданбай тұрып, алдын ала оның конструкторiне   назар аударылуы керек. Конструкторла виртуалды бола алмайды.

TestRing конструкторiне фигура туылмас бұрын назар аударылған. Өйткенi Ring.Init блогi Circle.Init-ке назар аударады, ал соңында Show блогы орындалып фигура экранға шығады. Бiрақ  Show әдiсi виртуальды және алдын ала қосылуы керек. Circle түрiндегi виртуальды әдiстертердiң қосылуы Krug.Init конструкторының бiрiншi әректi болып табылады. Ал Show блогына ол жұмыстың соңында назар аудылады. Ring өзiнiң Hide виртуальды әдiсi жоқ болғандықтан, ататегiнiң әдiсiн қолданған. Виртуальды әдiстер мұра қалдырады.

Виртуальды емес әдiстер қарапайым блоктар сияқты программаны компиляциялаған кезде қосылады (ерте байланысу). Виртуальды әдiстер үшiн әр түрге компилятор  олардың адрестерi  жазылған ВӘК (виртуальды әдiстер кестесi) кестiн жадыда құрады. ВӘК кестесi орындалу кезiнде қолданылады ("қожайын"-әдiспен кеш байланысу). Конструкторге назар аударған кезде, ол объектiдегi арнайы алаңға осы түрдегi объектiнiң BӘК кетесiне нұсқау жазылады. Fly әдiсi Show және Hide әдiстерiмен ВӘК кестесi арқылы байланысады.

Обектiнi динамикалық жадыда орналастыруға болады. Бұл оларды құруға ыңғайлы жағдай туғызады. Мысалы динамикалық обектiлердi тудыра отырып оларды виртуальды әдiстерiмен де байланысуға болады. Ол үшiн New процедурасының екiншi параметiрiне конструкторге назар аудару жазылады. TestCircle түрiне ^Circle нұсқауын қолданамыз және "ұшатын" дөңгелектiң туылу келесi түрде жазылады:

New(Circle,Init(151,182,30,5));  Cicrle^.Fly(100);

New процедурасының екi параметiрi  болған жағдайда керектi алаңдарға жадыдан орын алуды конструктор бiрiншi болып iстейдi. Dispose процедурасы жадыдағы объектiлердi жадыдан босатады. Ол үшiн жадыдағы объектiлердi босататын арнайы процедура блогiн iске қосылады. Объектiдегi мұндай процедура -деструктор деп аталынады. Десторукторды консторуктор сияқты бағдарламашы өзi жазады. Деструктор виртуальды болып келедi , ал конструктор бола алмайды.

Point объектiсiнде үш X,Y, Visable берiлгендер алаңы бар, GetX,GetY функциялары X пен Y-тiң мәндерiн қайтарады. Show процедурасы берiлген координатада  нүктенi экранда көрсетедi, ал Hide  нѕктенi "сөндiредi". IsVisable  функциясы нүктенiң экранда көрiп тұрғанын немесе сөнiп тұрғанын анықтайды. Х нүктесiн (0,0) координатасында көрсету программасының бөлiгi.

Объктiлiердiң ағаш тәрiздi құрылымын құрғанда, бағынышты объектiде, оның  тегi болатын объектiсiндегi берiлгендердi, олардың қасиеттерiнiң өзгеру немесе өзгермеуiне тәуелсiз, қайта сипаттауға болмайтындығын ескеру керек. Ал әдiстердi қайта анықтауға болады. Егер бiр-бiрiне байланысқан объектiлердiң аттары бiрдей әртүрлi әдiстерi болса, онда оларды қандай объектiмен жұмыс iстеуiне байланысты   шақырылады.

**1.3**   **Әдістер**

**Әдіс** - объектінің қасиеттерін өзгертуге арналған процедуралар мен функциялар. Операция – бұл әдістің сипаттамасы, ал әдіс операцияны жүзеге асыру болып табылады. Логикалық жағынан қарағанда әдістер бірдей операцияларды орындайды, бірақ әртүрлі кодтың фрагменттерімен жүзеге асырылады. Әрбір операцияда анықталмаған аргумент болады, ол сол операция қолданатын объект, бірақ басқа да аргументтер болуы мүмкін, ол оны параметрлейтін параметрлер. Әдіс тек класс пен объектіге қатысты.

         Объектінің кейбір атрибуттарының мағынасы тек сол объектінің оперциясына ғана қатысты бола алады. Ондай атрибуттар жабық атрибуттар деп аталады. Объектілерде жабық операцияларды да анықтауға болады.

Осылайша класс тапсырмасы үшін сол кластың аты көрсетілуі керек, одан кейін оның атрибуттары мен операциялары ( әдістері) көрсетіледі. Жалпыламалау және мұрагерлік объектілер классификациясының көп сатылы екендігін және объектінің әртүрлі кластарының арасындағы ұқсастықтарды анықтауға мүмкіндік береді.Мысал ретінде көптеген геоматриялық фигураларды келтіруге болады.

Жалпыламалау мыналарды анықтауға мүмкіндік береді:

Класс “бір өлшемді фигуралар”, “түзу” және ”доға ” кластары “бір өлшемді фигуралар” кластарының ішкі класы болып есептеледіб ал бір өлшемді фигуралар класы түзу және доға класының суперкласы болып табылады. Егер осындай келісушілікте қарайтын болсақ, суперклассоперациясы мен атрибуттары сол ішкі класқа сәйкес болады ( сол атрибуттар мен операциялар ішкі кластарға мұрагер бола алады), онда доға және түзу кластарының ( ішкі кластар) атрибуттарымен операциялары бір өлшемді фигуралар класына ( суперкласс) көшіріле алады. Әр супер кластың атрибуттары мен оперциялары ішкі кластың барлық сатысындағы мұрагері болып табылады және бұл кластарды сипаттауды жеңілдетеді және қысқардады.

         Кейде белгілі бір суперкластағы ішкі кластың операциясын қайта анықтау қажет болады. Ол үшін мұрагерлік нәтижесінде суперкластан алынған операция қолданылады. Егер жаңа операция мұрагерлік операциясын кеңейтетін болса, онда қайта анықтау қолданыдады. Төмендегі суретте үш өлшемді нукте класының және екі өлшемді нүкте класының атрибуттары мен операцияларының мұрагерлеу схемасы  көрсетілген. Екі өлшемді нүктенің координаталары берілген. Ол оның атрибуттары, ал екі өлшемді нүкте объектілерінің орындайтын операциясы Х, У координаталарын және нүктенің арақашықтығын анықтау. Үш өлшемді нүкте класы ( ішкі класс) өзінің екі өлшемді нүкте суперкласының Х,У координатасын мұрагерлікке алады және өзіне тағы бір атрибут- Z координасын қосып алады. Х,У координатасын анықтау операциясы да мұрагерлікке алынады. Одан басқа Z координатасын анықтау операциясы қосылады және келесі үш өлшемді нүкте арасындағы арақашықтықты анықтау операциясын қайта орындайәды. Үш өлшемді нүкте арақашықтығын есептеу формуласының  екі өлшемді нұсқадан айырмашылығы болады.

                               Түбірлік объект

                                                             Түбірлік объект

|  |
| --- |
| Екі өлшемді нүкте |
| Х координатасы  У координатасы |
| Х координатасын анықтау  У координатасын анықтау  Келесі нүктеге дейінгі арақашықты анықтау |

Атрибуттар

Операциялар

                                         ¯

                                 Туынды объект

|  |
| --- |
| **Үш өлшемді нүкте** |
| Х координатасы  У координатасы  Z координатасы |
| Х координатасын анықтау  У координатасын  анықтау  Z координатасын анықтау  Келесі нүктеге дейінгі арақашықтықты анықтау |

Turbo Pascal программалау тiлiнде әдiстердiң статикалық және виртуальды деген екi түрi бар. Статикалық әдiстер онша күрделi емес, жадыдан аз орын алады және тез орындалады, бiрақ ОБП барлық артықшылығын толығымен көрсете алмайды. Жоғарыдағы программада пайдаланылған әдiстер - статикалық әдiстер болып табылады, яғни, егер әдiстiң түрi , онда ол статикалық болып табылады.

**1.4 Виртуалды әдістер.**

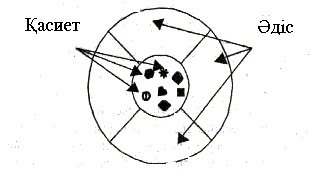
Виртуалды әдістермен оларды шақыратын процедуралар арасындағы байланыс программаны орындау барысында орнатылады. Бұл соңғы байланыс деп аталады. Оның алғашқы байланыстан айырмашылығы ол компеляция кезінде орнатылады. Виртуалды объектіні сипаттауда әдістің атынан кейін резервтегі virtual сөзі қолданылады. Туынды мен түбірлік объектінің виртуалды әдістерінің аттары бір- біріне сәйкес келуі керек. Виртуалды әдістері бар объектінің экземплярын инициализациялау арнайы әдіс – конструктордың көмегімен орындалады. Конструктор объект өрісіне алғашқы белгілерді иелендіреді және объектіні инициялизациялау кезіндегі керекті іс-әрекеттерді  орындайды. Әдіс- конструктор атында procedure сөзі  constructor сөзімен алмасады. Конструктор іс-әрекетіне қарама- қарсы іс- әрекет тағы да бір арнайы әдіс деструкторды орындайды. Ол destructor сөзімен сипатталады.

Конструктор виртуалды әдістер таблицасына нұсқағыш жасайды және ол таблицаны іздеу үшін пайдаланылады. Таблицаға виртуалды әдістер объектісінің балық адрестері кіреді. Виртуалды әдістерді шақыру кезінде оның аты бойынша адресі анықталады. Одан кейін сол адрес бойынша басқару беріледі. Әр объект типінің өзіне тән виртуалды әдістерінің таблицасы болады. Бұл операцияға әр түрлі процедураларды шақыру мүмкіндігін береді. Егер бір типті объектінің бірнеше экземпляры болса, онда конструктор әр экземпляр үшін шақырылуы керек. Ондай болмаған жағдайда программа жұмысы бірден тоқтатылады. Конструтор мен деструктор бос болуы мүмкін, яғни, оперциялар болмайды. Бұл жағдайда барлық керекті код  конструктор және деструктор сөздерін трансляциялау кезінде құрылады.

**Статикалық әдiс** - дегеніміз статикалық айнымалыларда программаның немесе процедураның іске қосылып жұмыс істегеннен бастап жадыдан адресі анықталып, программа немесе процедура жұмысын аяқтағанша айнымалы тек сол адресте сақталуды айтамыз. Мысалы: var  бөліміне жазылған айнымалыларды статикалық айнымалыларға жатқызамыз.

**1.5 Кластар құру**

   Айналаға қарап, өмірдің көптеген заттарын байқауға болады. Мысалы,  ит, үстел, теледидар, автомобиль. Мұнда кейбір геометриялық фигураларды да жатқызуға болады. Мысалы, шеңбер, тіктөртбұрыш және т.б. Қоршаған ортаның объектілерін екі типке бөлуге болады: олардың бәрінің **қалпы**және **мінез – құлқы** болады. Мысалы, автомобильдің қалпы бар (түсі, жүру жылдамдығы, типі және дөңгелек көлемі) және мінез – құлқы (тежеуі, жылдамдатуы).  Шеңбердің өзіне тән қасиеттері бар. Мысалы, орталық координаттары, радиусы, түсі. Және де өзіне тән мінез – құлқы (салу, орын алмастыру, түсін өзгерту).



1.1. сурет. Жалпы белгіленген объектінің программалық бейнесі.

программалық объектілер, қоршаған орта объектілеріне тән өзіндік қасиеттері де бар. программалық объектілер өзінің қалпын өзінің **қасиетінде** сақтайды. Қасиеті – идентификаторы бар, белгілі бір мән. программалық объектілердің мінез – құлқын **әдіс**арқылы іске асырады. **Әдіс –**бұл объектімен байланысқан астыртын программа (процедура). Сонымен, **объект** дегеніміз – бұл құрамында әдіс пен қасиеті бар программалық бірлік (1.1 сурет).

ООП-да класс үшін мұрагерлік, инкапсуляция және полиморфизм принциптері енгізілгек.

Жоғарыда айтылып өткен мәселе жайында мысалы келтірейік:

Мысалы ретінде кез келген программа құру қажет дейік, ол программа шеңберді салып оны кез келген түске бояуы тиіс делік. Бұл үшін оған радиус, орталық координат және де ол шеңбер боялатын түс қажет. Егер стандартты түрде даярлайтын болсақ, онда бізге келесі мәндер керек болады:

R – шеңбер радиусы;

X, Y – шеңбер орталығының координаттары;

Color – шеңбер түсі.

Бұл шеңберді салу үшін бізге процедура қажет, демек оны Draw деп атайық. Ол R, X, Y, Color  координаталы шеңберді бейнелеп тұрады.

Стандартты даярлаудың кемшілігі бұл параметрлердің бір – бірімен байланыссыз болуы. Мысалы, бір емес үш шеңбер салу керек (мысалы аққала).  Стандартты программалау жағдайында бізге екі амалдың бірін таңдау қажет болады. Біріншісі - шеңбер үшін R, X, Y, Color, онан соң осы шеңбердің әрқайсысы үшін Draw командасын  шақыру қажет.  Бұл тәсілдің кемшілігі егер көп шеңбер салу керек болса, онда бізге көптеген ауыспалы мәндер қажет болады. Бұл программаның үлкен размерділі болуына әкеліп соғады. Екінші тәсіл - әрбір шеңбер үшін R, X, У, Color, бірақ екінші шеңбер салу барысында, алғашқы шеңбердің  мәні жоғала береді. Бұл тәсілдің кемшілігі  соңғы шеңберден басқа барлық шеңберлердің мәнін жоғалтып аламыз.

Егер бұл суретте тек аққала  салумен шектеле қойсақ, онда кемшілігі көрінбей қалады. Көлемі жағынан үлкен программа құратын болсақ, онда бізге программаға елеулі өзгерістер енгізу қажет болады. Аққаламен қарастырылған мысалда бірнеше өзгеріс болады.

Аққаланың экран бетінде жүруі үшін Move процедурасы қолданылады. Аққаланы баратын жеріне жеткізу үшін программистке аққаланың баратын жерінің координаттары қажет емес (анығырақ айтсақ, әрбір баратын нүктесінің координаталарын). Сондықтан Move процедурасының параметрлері барлық мәні емес, тек жанама мәндерін ғана білуі тиіс. Демек, аққала өзінің бастапқы нүктесінен бастап жүруі тиіс.

Аққаланың барлық түстерін өзгерту үшін ChangeColor процедурасы қолданылады. Ол процедура орындалу үшін  екі параметр қажет. Өзгертілетін объектінің атымен өзгертілетін түсі. Егер екінші тәсілді пайдаланатын болсақ (ауыспалы мәні аз), онда біздің процедура аса қиынға түседі.  Мұның себебі, аққала құрылатын шеңберлердің (соңғысынан басқа) мәндерінің болмауы. Себебі, Draw процедурасы шақырылған сайын ол мәндерді білуі тиіс.  Демек, аққала салынған жағдайда ауыспалы R, X, Y, Color мәні тек соңғы объект үшін ғана берілген болады. Ал, қалғандарының мәні жоғалған болып табылады. Сондықтан шеңберлердің бәрінің орнын ауыстыру үшін, екінші тәсіл қолайсыз болып табылады.

Бірінші тәсілді қолдану барысында, өте үлкен тәсілді, барлық өзгертулерді енгізуге болады.  Бірақ жаңадан енгізілген мәндер үшін бір процедураны үш  реттен шақыруға тура келеді. Егер тағы да бірнеше аққала салатын болсақ, онда біздің программамыздың оқылуы өте баяу болғандығынан басқа, стандартты жолмен жазу өте қиын, әрі оқылмай қалу мүмкінділігі басым болады. Енді аталған мәселе объектті – ориентирлік программалау  жағынан қарастырайық. Мұнда бірнеше ауыспалылардың орнына, бірнеше объекті пайда болады.  Басында барлық шеңберлерді бірдей элементтер, бірақ әртүрлі құрылымнан тұрады деп есептейміз. Олар бір – бірінен тек параметрлерімен ерекшеленеді деп алайық. Draw, Move и ChangeColor процедураларының орындалуы бәрі үшін жалпы бірдей болады. Демек, әрбір шеңбер өзге шеңберге ұқсас тек өзінің параметрлерімен (радиусымен, орталық координатасымен, түсімен) ерекшеленетін, бірақ бірдей іс – қимыл әрекетін жасайтын деп есептеледі.  Ары қарай қарастырылып отырған мәселе ОБП терминдерін қолдану арқылы жазылады.

Әрбір элемент өз бетінше объект ретінде қарастырылып ортақ бір процедураны орындайды. Ол  әрбір элемент **(экземпляр)**белгілі бір **классқа** жатады. Объектінің параметрі (радиусы, түсі және т.б.) оның **қасиеті** деп аталады. Ал оның жасайтын іс - әрекеті (түсін өзгертуі, орын алмастыруы) **әдіс**деп аталады. Әдіс әрбір сұранысқа жауап беретін болса, онда оның класы анықталады, объектінің экземплярын анықтайды. Және керісінше.

Енді аталған мәселелерді мысалда қолданып көрейік. Барлық шеңберлер Round класына жатады деп алайық (5.2. сурет).

Round класының қасиеттері болып, төмендегілер есептеледі:

■   R – шеңбер радиусы;

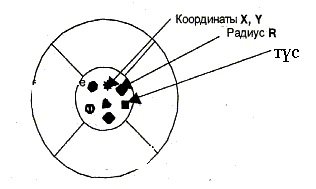
■   X, Y – шеңбердің орталық координатасы;

■   Color – шеңбер түсі. Round класының әдістері:

Draw – берілген параметрлер бойынша шеңбер салады;

Move – таңдалған бағытта шеңберлердің бәрін біркелкі қозғалтады;

ChangeColor – шеңбердің түсін өзгертеді.



1.2 сурет. Round класындағы бейнелер.

Аққаланы салу үшін, үш шеңбер қажет. Оның үстіңгісін Head, ортаңғысын Body,  ал астынғысын Foot  деп атауға болады. Демек, олардың барлығы бірдей қасиетке ие болады **(R, X, Y, Color)**және бірдей әдістерді шақырады **(Draw, Move,ChangeColor).**

ООП тәсілінде, қарапайым тәсілде қанша сақтау құрылғысы қолданылса, сонша сақтау құрылғысы қолданылады. Бірақ мұнда кішкене болса да есептеуге ыңғайлы болады. Тәуелсіз блоктардан құралып, құру және қазіргі заман талабына сай болады.  Программаны қолдану тәсілінің оңай болу себебі, ол әртүрлі блоктан құралып, басқа программаларда қолдануға мүмкіншілік береді. Бірнеше жаңа блоктарға қосуға мүмкіншілік береді.

**Мұраға алу.** Аталған мысалға қайта оралайық. Аққалаға көз салу керек делік, уақыт өткен сайын, ол көзін ашып жауып тұруы тиіс. **Round** объектілері секілді бұларда сол типке жатуы тиіс. Аққаланың көздері аққаламен бірге экран бетімен жылжып отыруы тиіс және түсін өзгерту мүмкіншілігін сақтауы керек.  Көздер үшін тағы бір класс құруға болады. Ол  **Eyes** (көз) деп аталады. Онда **Round**  әдісінің қасиеттері мен амалдары толық сақталуы және **Blink** (ашу - жұму) әдісі қосылуы тиіс.

Ол үшін ООП – да мұраға алу механизмі қарастырылған. Бұл механизм иерархиялық құрылымнан тұрады. Алдымен**Round** класы құрылады, сонан соң **Eyes** класы құрылады да **Round** класының қасиетіне ие болады. **Eyes** класына **Blink** әдісі  қоладанылады (1.3 сурет).

**Round**  класы ата – анасы, ал **Eyes** класы баласы, яғни мұраға алушы болып табылады.

Яғни, **мұраға алу** дегеніміз – мұрагерлік қасиетке ие болатын класстың ата – анасының қасиетіне және әдісіне ие болуы.

1.3 сурет. Eyes класының Round класының қасиеттері мен әдістеріне мұрасына ие болуы.

Демек, ООП- ны қолдану программаның көлемін қысқартып, оның тез істеуін қамтамасыз етеді.

**Инкапсуляция**— кластын берілгендері мен әдістерін біріктіріп, біртұтас түрінде қарастыру мүмкіндігі және оны басқа кластардың пайдалана алмауы. Мұндай класпен жүмыс істеу операторлары Visual Basic'тe *Private* кілттік сөзі арқылы жазылған процедураға енгізіледі *(инкапсуляция —*қорап ішінде).

**Полиморфизм.** Полиморфизм термині гректің екі сөзінен  шыққан: poly (көп) и morphos (форма). Демек полиморфизм  дегеніміз «көп форма» деген мағынаны білдіреді.  Бұл типтің ерекшелігі бұл тип кез келген  әртүрлі типтегі мәндерге ие  бола алады. Полиморфты параметрлері бар процедуралар (функциялар) **полиморфты** деп аталады. Ары қарай полиморфты ООП да қолданылуы қарастырылады.

Шеңберлермен мысалдарға қайта оралайық,  мысалы экранда тек шеңбер емес, басқа да геометриялық денелерді салу керек делік. Олармен де жоғарыда келтірілген параметрлермен қимылдар жасауы керек делік. Оны салу үшін біз оның биіктігі мен енін және ұзындығын білуіміз керек.

Аталған параметрлерді енгізу үшін біз класс құруымыз керек. Ол классты **Rect**(rectangle – тіктөртбұрыш), ол **Round**класының мұрагері болуы керек.

**Round** класын универсалды ету үшін, оның қасиеттерін өзгерту керек.оның радиусымен орталық координатасына орнына төмендегідей мәндерді енгізу керек:

**Тор**– фигураның жоғарғы жағының координатасы;

**Left**- фигураның сол жағының координатасы;

**Height**- фигураның биіктігінің координатасы;

**Width**- фигураның енінің координатасы.

Осы қасиеттер бойынша  фигураға жетпей координаттарды оңай  анықтауға болады. Мысалы, тіктөртбұрышты (квадратты) бейнелеу үшін сол жақ координатасына енін қосу керек, ал төбесі жағынан биіктігін алып тастау керек. Делік,

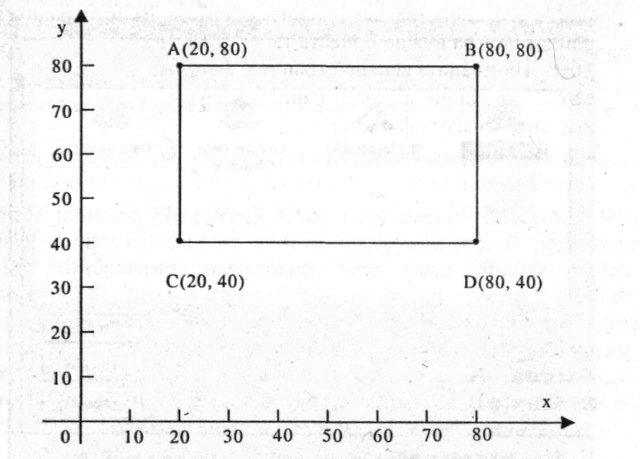
Тор =80, Left =20, Height =40, Width =60.

Онда оң  жағы  мына мәнге ие болады: Left + Width =80,

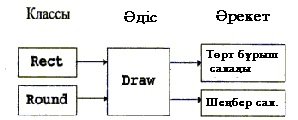
Ал төменгісі мынаған: Тор - Height =40.

Демек, аталған тіктөртбұрыш келесідей координаттарға ие болады (оң координата – горизонталь ось үшін, екінші мәні – вертикаль ось үшін):  А (20,80),  В (80,80), аталған тіктөртбұрыш 13.4 суретте қарастырылған.

Round тегі негізінен жоғарыда аталған қасиеттерге сүйене отырып, Rect  ұрпағын құруға болады. Яғни бұл ұрпақты тіктөртбұрыш салу барысында қолдануға болады. Демек, бұл сурет бейнелеу шеңберлерді бейнелеуден әлдеқайда бөлектеніп тұруы тиіс. Сонымен, Rect Draw әдісін өзіне тән етуі тиіс, ата – анасынан бөлектенуі тиіс. Өзге жағдайларда Rect әдісіне жататын Draw программасын шақыртқан уақытта экранда тіктөртбұрыш емес, шеңбер бейнеленіп тұрады.  Мұндай оқиға болмау үшін, Rect әдісіне Draw әдісінің қасиеттерін сипаттау қажет. Сонда Round әдісіне жататын объектілер үшін Draw  мәні шеңбер болады да, Rect әдісі үшін, Draw мәні тіктөртбұрышты бейнелейді. Draw атымен аталған әдістер әртүрлі қимылдар үшін жауап береді. Ол қай кластағы суретті бейнелеуге байланысты болады. Аталған жағдай ООП – да полиформизм деп аталады. Аталған полиформизмнің ООП – да графикалық интерпретациясы  13.5 суретте келтірілген.

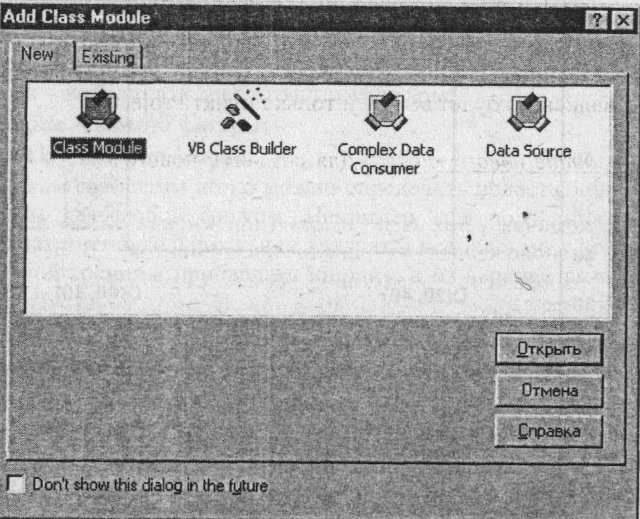


1.4 сурет. Берілген биіктік, ені мен ұзындығына байланысты тіктөрт бұрыш салу әдісі.

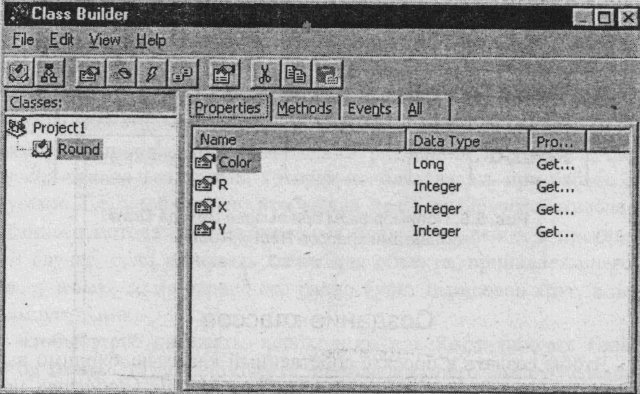


1.5 сурет. Round және Rect класс объектілері үшін Draw  әдісін шақыртқан кездегі полиформизм.

Проектте өзіндік класс құру үшін, менюде Project  - Add Class Module (Класс модульін қосу) командасын таңдау қажет. Нәтижесінде диалогтық терезе бейнеленеді. Бұл терезеде Class Module пунктін  таңдауға болады. Құрылып жатқан класс үшін  берілгендер мәнін осы терезеде енгізуге болады. Бірақ, Visual Basic класы арқылы VB Class Builder (VB класс құрғыш) пунктін таңдау арқылы салған жөн болады.



1.6 сурет. Класс модулін құру диалогтық терезесі.

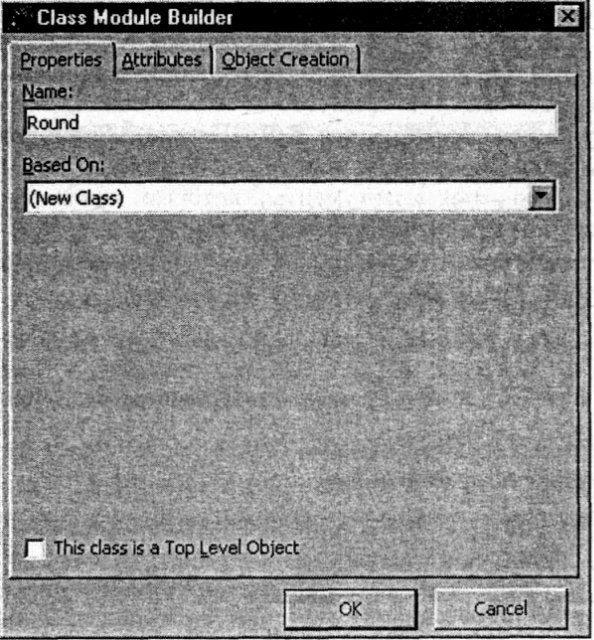


1.7 сурет. Класс құрғыш  терезесі.

Өзіндік класстар құру үшін Visual Basic- те, Class Builder (VB класс құрғыш) әдейі құру амалы қарастырылған. жоғарыда аталған Round  класын осы амал арқылы  құру әдісін қарастырайық.

VB Class Builder  пунктін таңдағаннан кейін, класс құрғыш  терезесі пайда болады. Бұл терезенің сол жақ шетінде аталған проектінің өзінің проектілері бейнеленеді. Егер класстар құрылмаған болса, онда Project 1 пункті ғана бейнеленеді.

**Жаңа класстар құру.** Жаңа классты құру үшін мына командалар қолданылады (13.8. сурет): File \ New | Class (құралдар панелінде орналасқан немесе Add New Class пернесі). Мұнда Name қатарында класстың аталуын көрсету керек,  аталған жағдайда  ол - **Round.** Егер де ұрпақ класын құру керек болса, онда  Based On таңдау қажет. Бар тізімдердің арасыннан қай классқа жататындығын нұсқау қажет.  Аталған жағдайда бұл  тізім бос болады (қолданылбалы тізім әлі құрылмаған).



1.8 сурет. Жаңа классты құруға арналған диалогтық терезе.

Жаңа класстың  құрылымын құру үшін, басқаша айтқанда оның қасиеттерін сипаттау үшін, терезенің сол жақ бетінен классқа атау таңдау қажет. Мәзір қатарынан File \ New, оның ішінде: Property (қасиеті), Method (әдісі), Event (құбылыс).  Бұдан бөлек құралдар панелінде  ұқсас әрекеттер жасайтын пернелерді пайдалануға болады.

Add New Property (Жаңа қасиет қосу).

 Add New Method (Жаңа әдіс қосу).

Add New Event (Жаңа құбылыс қосу).

**Класстың қасиетін қосу.**Жаңа классқа жаңа қасиет қосудан бұрын ол қасиетке жаңа ат табу қажет. **Round**класы үшін, келесідей қасиеттер қосу қажет:

**R**(**Integer** типі**);** **X, Y**(**Integer** типі**);** **Color** (**Long** типі**).**

Түс беру үшін, **Long** типті толық мағыналар қолданылады. Сол үшін **Color**қасиетін көрсеткенде осы тип қолданылады.

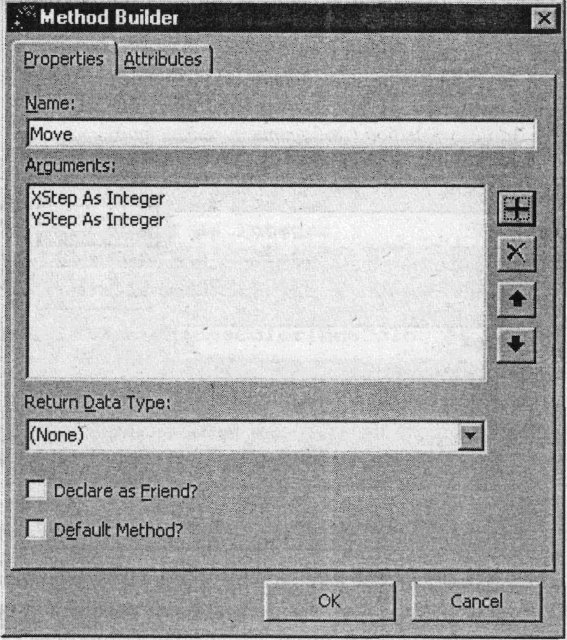
**Класс әдістерін қосу.**Классқа жаңа методтар қосу **Method** **Builder** (Әдіс құрғыш) (13.9 сурет) терезесінде іске асады.**Name** мәзірінде әдістің атауы беріледі. **Arguments**  қатарында, шақырған жағдайда орындалатын  аргументтер тізімі беріледі. Аргументтермен жұмыс терезенің оң жағындағы **Method** **Builder**  мәзіріндегі пернелер арқылы орындалады.

**Add (қосу)** – жаңа аргумент қосады.  **Delete** (өшіру) – белгіленген аргументті өшіреді.

**Move** **Up** (жоғарыға орын ауыстыру) – тізім бойынша бір қатар жоғары көтереді.

**Move** **Down** (төменге орын ауытыру) - тізім бойынша бір қатар төмен түсіреді.

**Add** пернесін басқан кезде, **Add** **Argument** (аргумент қосу) диалогтық терезесі пайда болады. Бұл мәзірде аргументтің аталуын (Name қатары), және де бірнеше қосымша параметрін көрсету қажет. Және де қосымша ашылатын **Data** **Type** тізімінің ішінен аргументтің типін нұсқау қажет (1.10 сурет).

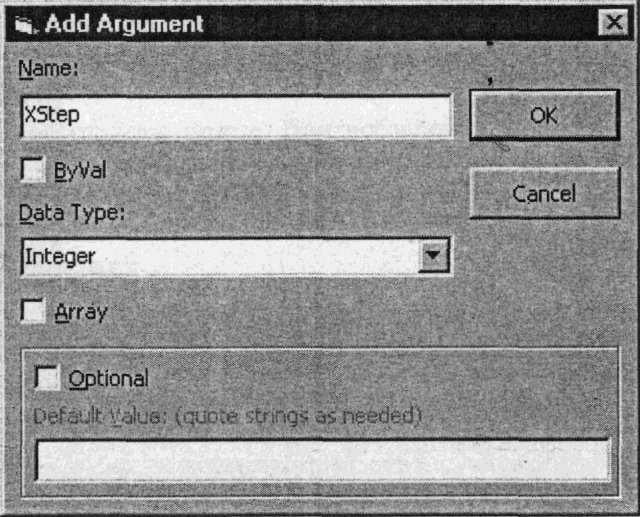


1.9 сурет. **Round**  класына **Move** әдісін қосу.

Классқа  осы әдіс арқылы **Draw, Move** және **ChangeColor** әдістерін қосу қажет. Осының алғашқысы үшін аргументтер тізімі  көрсетілмейді. Өз алдына **Move** екі аргументке ие, олар: **XStep** As **Integer** және **XStep** As **Integer.**Бұл аргументтер**Move** әдісін шақыру барысында іске асатын, шеңбердің вертикаль және горизонталь  орын ауыстыру мәндерін анықтайды. Басқаша айтқанда, бұл аргумент шеңбердің тұрған орнынан **Move**  әдісін шақырғаннан кейін, қай орынға ауысатын көрсетеді.

**ChangeColor** әдісі  **NewColor** As Integer атты жалғыз аргументке ие, бұл аргумент салынып жатқан шеңберге қандай түс беру болатындылығын анықтайды.

**Классты сақтау.**Керек компоненттердің бәрін енгізіп болғаннан кейін, проектіге енгізілген өзгертулерді сақтау үшін**File\UpdateProject** (проектті жаңарту) командасын орындау қажет.  Егер мұны орындамаса, онда жабу барысында Class Builder ескерту терезесі ашылады. Бұл терезеде өзгертілген немесе жаңадан құрылған классты сақтау ұсынылады. Жоғарыдағының бәрін орындау  барысында сізде жаңа Round класы қосылады. Бұл класстың кеңейтілуі CLS, бұл файлдың орналасу орны Round.cls файлының ішінде болады.



1.10 сурет. Move әдісіне **XStep**  аргументін қосу

**Кодтар класын өзгерту.** Қолданбалы класс құру барысында қорытынды этапы болып, автоматты түрде құрылған  кодты өзгерту болып табылады. Жоғарыдағының бәрін орындау  барысында сізде жаңа Round класы жаңа автоматты түрде енгізілген кодымен бірге беріледі. Бітіру барысында классты құру классты құрғыш мәзірінде  өзіне тән  кодтар терезесі ашық тұрады.  Мұнда бірнеше енгізулер енгізу қажет. Round  әдісінде көрсетілген командалар тізімін нұсқау қажет. Бұл өзгертулер тізімі 5.1. листингісінде көрсетілген.

Шеңберді көрсету барысында **Circle**әдісі қоладынлады. Оның синтаксисі " VB6 сурет салу" атты 12 тарауда толық  қарастырылады.

Мұнда **Prop­erty** **Let**  и **Property**   **Get** процедураларына аса назар аудару қажет. Бұл процедуралар автоматты түрде енгізіп қойылады. Бұл процедуралар **Let** қасиеттерін орнату барысында және қарапайым **Get**  мәнін  шақыру барысында орнатылады.

Аталған тарауда қозғалған сұрақтарға ActiveX басқару элементтері тақырыбына 17 тарауда көрсетілген.

**' сақтау үшін локальды ауыспалылар**

'  Қасиет мәндері

**Private mvarR As  Integer**

**Private ravarX As  Integer**

**Private xnvarY As  Integer**

Private mvarColor As  Long

**Public  Sub ChangeColor(NewColor As Long)**

**Color  = NewColor**

**Draw**

**End Sub**

**Public Sub Move(XStep As Integer, \_**

**YStep As Integer)**

**Forml.Circle (X, Y), R, Forml.BackColor**

**X = X + XStep**

**Y = Y + YStep**

Draw

**End Sub**

**Public Sub Draw()**

**Forml.Circle (X, Y), R, Color**

**End Sub**

**Public Property Let Color(ByVal vData \_**

**As Long)**

**'Syntax: X.Color = 5**

**mvarColor = vData**

**End Property**

**Public Property Get Color() As Long**

**'Syntax: Debug.Print X.Color**

**Color = mvarColor**

**End Property**

**Public Property Let Y(ByVal vData \_**

**As Integer) 'Syntax: X.Y = 5**

**mvarY = vData**

**End Property**

**Public Property Get Y() As Integer**

**'Syntax: Debug.Print X.Y**

**Y = mvarY**

**End Property**

**Public Property Let X(ByVal vData\_**

**As Integer)**

**'Syntax: X.X =5**

**mvarX = vData**

**End Property**

**Public Property Get X() As Integer**

**'Syntax: Debug.Print X.X**

**X = mvarX**

**End Property**

**Public Property Let R(ByVal vData \_**

**As Integer)**

**'Syntax: X.R = 5**

**mvarR = vData**

**End Property**

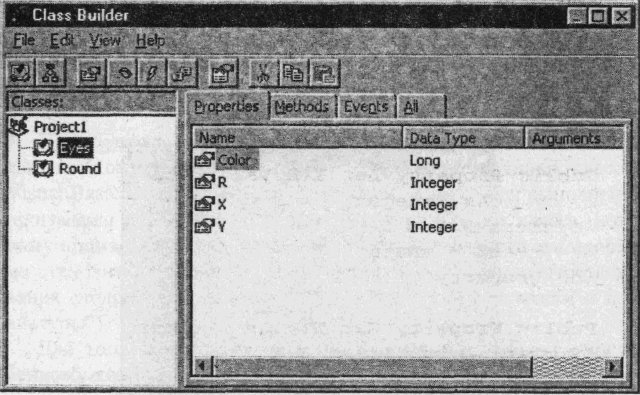
**Public Property Get R() As Integer**

**'Syntax: Debug.Print X.R**

**R = mvarR**

**End Property**

**Мұраға алу.** Алдыңғы тарауда **Round**және **Eyes**кластарының ұрпақтану принципетрі көрсетілген болатын. Visual Basic Класс Құрғышында өте қарапайым тәсілмен өз планыңызды іске асыруға мүмкіншілік бар. Ол үшін жоғарыда көрсетілгендей,  алғашқыда **Round**класын құру қажет. Онан соң экранның сол жақ бетінде Project 1 Class Builder белгілеу қажет. File | New \ Class немесе Add New Class пернесін қолдану керек. Жоғарыда көрсетілгендей Class   Module   Builder (13.8. сурет) **Eyes**класының  атын нұсқау керек. Based On тізімінде **Round**класын белгілеу қажет. Бұл 1.11 суретте көрсетілген.



1.11 сурет. **Eyes** класының **Round** қасиеттерімен әдістерін мұраға алуы

бұл жағдайда **Eyes** класының әдістері қайта анықталған болады. Немесе оларға қайта мән беру керек болады (**Round** – тың басқа әдіс кластарымен салыстырғанда, процедура терезесінде бұл мәндер “бос” болады).

**Объектілерді құру.**Кез келген класстың жаңа экземпляры жаңалықтар операторы арқылы құрылады. Мысалы, **Dim,**қызметтік сөз тіркесі арқылы As **New:**

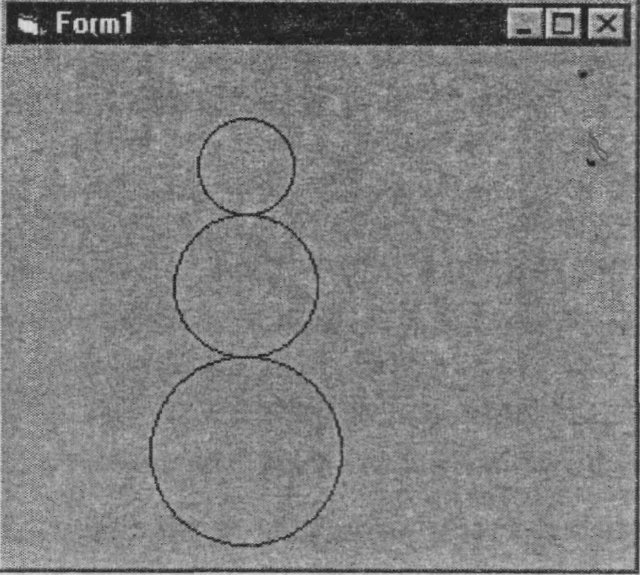
**Dim экземпляр As** **New класс**

Бұл жағдайда **класс**параметрі ретінде, бұл жағдайда қолданылатын Visual Basic классын  нұсқауға болады. Мысалы, класс формасы **(Forml)**немесе класс проектісі **(Project 1).**Басқаша жағынан мүмкін емес болып, басқа класс жаңа экземплярын құру, бөлектеп айтқанда, бар немесе басқаша да басқару элементтерін құру болып табылады. Мысалы,**CommandButton** (команда пернесі). Басқаша айтқанда программалық жағынан жаңа басқару элементін  қосу мүмкін емес болып табылады (мысалы, перне және т.б.).

Басқаша жағынан қарасақ, **класс**параметрі болып, құрушының көмегімен құрылған кез келген бір класс болуы мүмкін. мысалы,**Round:**

**Dim** **Head** **As**  **New** **Round**

13.2. листингісінде **Round** класы, объект кластарын қолдану арқылы аққала құру программалық кодтары келтірілген. 13.12. суретте  берілген кодтың жұмыс істеуі келтірілген.



1.12 сурет. Round объект класы арқылы аққала салу.

1.2 листингісі **Round** объект кластарын қолдану арқылы аққала құру программалық кодтары.

**Private  Sub Form\_Load()**

**Dim Head As  New Round**

**Dim Body As  New Round**

**Dim Foot  As  New Round**

**' Қасиеттерін** **орнату**

Head.X  =   100:   Head.Y  =   50

**Head.R**=   **20**

**Head.Color = vbBlue**

**Body.X = 100: Body.Y = 100**

**Body.R = 30**

**Body.Color = vbBlue**

**Foot.X = 100: Foot.Y = 170**

**Foot.R = 40**

**Foot.Color = vbBlue**

**1** **Аққаланы** **салу**

**Head.Draw**

**Body.Draw**

**Foot.Draw**

**End Sub**

Vis­ual Basic құрғыштары арқылы класстарды құру принциптері көрсетілді.

**Объект типі.** Жоғарыда көрсетілгендей, кез келген объект Visual Basic немесе қолданбалы класс болсын, әйтеуір белгілі бір классқа жатады. Кейде программаның жұмыс істей барысында классты анықтау қажеттілігі туады. Бұл қажеттілік орын алмастыру процесінде  өзекті мәселелердің біріне айналып келеді.

Қай классқа жататындығын анықтау үшін, келісімді құрушы оператордың **If ...End If:** көмегі керек болады:

**If  TypeOf   объект Is  класс Then**

**оператор**

**End** **If**

Егер аталған объект **класстың экземпляры**болып табылса, онда операторлар орындалған болады.

**1.6  Жиындар құру**

**Жиын - Visual Basic**  те өз бетімен объектілерді сақтау құрылымын білдіреді (немесе объектіге бару жолын). Жиынмен жұмыс массивтермен жұмыс секілді бірдей болып табылады. Бірақ жиынның жағымды жақтарының бірі болып, оларға алдын ала диапазонды нұсқау қажет емес. Мұнда объектіні қосу немесе өшіру барысында автоматты түрде диапазон беріле береді.

Visual Basic тілінде қолданушы стандартты панельді пайдалануына мүмкіншілігі бар. Мысалы, **Forms** (дайын форма) немесе**Controls** (формадағы басқару элементтері). **Collection** класы негізінде өзіне керек элементті өзі құрып ала алады.

Мысалы:

**Public  colPictures As New Collection**

Аталған жағдайда **Collection**класының жаңа түрі құрылады. Оның аты**colPictures**болады.

Жиынтыққа жаңа элементті қосу. Жиынтыққа жаңа элементті қосу **Add:**әдісінің көмегімен іске асады.

**жиынтық.Add  Item,    [Key],    [Before],    [After]**

мұнда

**жиынтық-** жиынтық  атауы;

**Item**– қосылатын элемент;

**[Key]**– кілт (тәртіб ретінде, қатар мәні) арқылы жиынтық арасынан іздеуге болады;

**[Before]**– жиынтық ішіндегі элемент (немесе оның қатар саны), алдына жаңа объект қосылатын қатар (мысалы, **Before  =  2**);

**[After]**- жиынтық ішіндегі элемент, қосылатын жаңа элемент сол қатардан кейін қосылады.

Бұл жайда  бір анығы **Add**әдісінен кейін **Before**немесе **After**бір – ақ элементі қосыла алады.

Айта кететін бір жай,**Collection**негізінде құрылған жиынтық элементтерінің саны өзге массивтермен немесе стандартты жиынтықтармен  (0 – ден басталатын жиынтықтармен)  салыстырғанда 1 – ден басталады.

Элементті жиынтықтан өшіру. Жиынтық ішінен өшіру **Remove:**әдісі арқылы іске асады.

**Жиынтық.Remove  Index**

**Index**параметрі ретінде өшірілетін элементтің позициясын немесе кілтті (егер ол **Add.**әдісі арқылы алдын ала құрылған болса) көрсетуге болады.

Жиынтық ішінен белгілі бір элементті таңдау. Жиынтық ішінен белгілі бір элементке ауыспалының мағынасын енгізу үшін **Item:**әдісін қолдануға болады.

ауыспалы= жиынтық.Item(Index)

**Index**әдісі дәл **Remove**әдісі секілді қолданылады.

**Item**әдісі стандартты түрде қолданылатын болғандықтан, командалар тізбегінде оны көрсету міндетті емес. Демек, төмендегі екі оператор бір нәрсені орындайтын болады:

**vntltem = colPictures.Item(l) немесе**

**vntltem = colPictures(1)**

**Жиынтық** **ішіндегі** **элементтер** **саны**. жиынтық ішіндегі элементтер санын анықтау үшін, **Count**қасиетін қолдану қажет. Бұл жағдайда ескере кететін бір жай, жиынтыққа әрбір өзгеріс енгізген сайын бұл элемент мәні өзгеріп отырады және программдық кодта да, даярлау барысында да өзгертілмейді.

Егер жолды басқа  жолмен салыстырмалы түрде тексеру қажеттігі туса, Is операторын қолдануға болады. Аталған команданың қорытындысы ретінде **True**мәні болады (егер жол бір объектіні көрсететін болса) немесе **False**(егер жол объектінің бірнеше объектіні көрсететін болса). Басқару конструкциясында Is операторында қолданылады.

**Жиынтық** **элементтерін** **өңдеу.** Жиынтық элементтерін өңдеу үшін, тәртіп ретінде арналған **For Each...Next**операторы қолданылады. Оның синтаксисі ретінде:

**For Each элемент In жиынтық**

**оператор Next**

**жиынтықтың** әрбір элементі цикл орындалу үшін (индекстердің өсуіне қарай) **блок** **операторы** қолданылады. Аталған циклды орындау үшін шектеулер берілген, атап айтсақ, әрбір элемент, жиынтыққа жатады. Міндетті түрде **Variant**типін қолдануы тиіс (мысалы, ауыспалы, элемент ретінде  процедураның басында тапсырыссыз сипатталады):

**Dim Birthday**

**For Each Birthday In colBirthday**

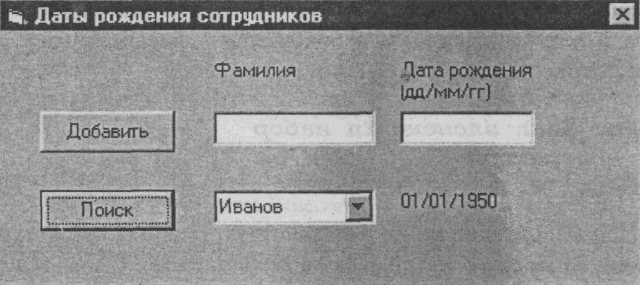
**If  Birthday =  #l/l/2000# Then Beep Next**

Жоғарыда келтірілген командалар даталар арасында **#1 /1/2000**#, **colBirthday**жиынтығында бар ма екендігін тексереді. Егер ол дата табылса, онда **Веер**операторы арқылы дауыс береді. Берілген дата **Birthday**  ауыспалысына алынады.

Мысалы 2.1. Жиынтық арқылы жұмысшылардың туылған күні, айы және олардың аты – жөнін енгізу. Берілген аты – жөні бойынша жұмысшылардың туылған күнін анықтау.

Шешуі. Жиынтық ретінде **colBirthday** қолданылып, қолданушының көмегімен жұмысшылардың аты – жөні мен туылған даталары енгізіледі. Мұндағы туылған датасы жиынтықтың міндетті элементтерінің бірі болып табылады. Ал, аты – жөні – кілттік мәнге ие болады және сол бойынша іздеу іске асады.

Мәндерді енгізу  қосу пернесі  арқылы іске асады. Демек тізімге енгізген сайын бір енгізілген аты – жөні қосылып отырады. Бірнеше мән енгізіліп болғаннан кейін іздеуге де болады. Бір аты – жөнді белгілеу арқылы ашылатын тізімде іздеу  пернесін басу қажет.  Нәтижесінде, оң жақ шетінде белгіленген жұмысшының туылған датасы көрсетіледі (14.1. сурет).



2.1 сурет. Иванов деген жұмысшының  туылған датасын анықтау.

2.1. листинг жұмысшылардың туылған даталарын өңдеу барысында жиынтықты қолдану

Dim colBirthday As New Collection

Dim Surname As String

Dim Birthday As String

Option Explicit

Private Sub cmdAdd\_Click()

Surname = txtSurname.Text

Birthday = txtBirthday.Text

' Жиынтыққа аты – жөн қосу

cboSurname.Addltem Surname

' Туылған датасын енгізу

' жиынтыққа кілттік мәнімен (аты – жөнімен)

colBirthday.Add Birthday, Surname

' тексттік жолдарды тазарту

txt Surname. Text = ""

txtBirthday.Text = ""

txtSurname.SetFocus

End Sub

Private Sub cmdExit\_Click()

End

End Sub

Private Sub cmdSearch\_Click()

' Туылған датаны іздеу

' белгіленген аты – жөн бойынша

lblSearch. Caption = \_

colBirthday(cboSurname.Text)

End Sub

**IІ** **ТАРАУ. WWW және HTML**

**2.1 HTML**

 Қазіргі заманғы Интернеттің дамуы берілгендерді алмастырудың жаңа протоколының пайда болуымен 90 – шы жылдардың басында пайда болды. Бл протокол HTTP (Huper Text Transmission Protocol – гипертекстті жіберу протоколы) деп аталады. Осы протоколмен бірге Интернет арқылы файлдарды жіберіп, HTTP серверлерінің желісін біріктіретін World Wide Web (кейде WWW немесе Web деп атайды) протоколы пайда болады.

Бұл файлдардың негізгі бөлігін Web – беттер деп аталатын, HTML (Hyper Text Markup Language – гипертексті белгілеу типі) тілінде жазылған, арнаулы файлдар құрайды. Web – беттер не Интернетте жарияланатын файлдар HTTP серверлеріне орнату арқылы жарияланады. Web – беттердің мазмұны кезкелген тақырыптарда болуы мүмкін, бірақ олардың барлығында түпкі тамыры – HTML тілі. HTML құжатының кеңейтілуі көбінесе htm немесе html болып келеді.

HTML тілі WWW қызметімен бірге пайда болып, дамып келген. Ол WWW – тің негізгі және сонымен бірге оның өте әйгілі болғандығының себебі болып табылады. HTML тілінің мәнін және қажеттілігін оның атынан түсінуге болады.

Гипертекст бұл – иллюстрация, мультимедиялық қоюлар, басқа документтерге гиперсілтемелер т.с.с. қосымша элементтерді басқаратын арнаулы кодтары бар текст. Ал белгілеу осы тексттің кодтарын қою, демек осы документтің Web – браузерлер арқылы қалай көрінетінін анықтайды. Белгілеу қарапайым немесе күрделі болуы мүмкін, бірақ бастапқы текст бәрібір құжатта өзгертілмеген түрде сақталады.

Мұндағы ең негізгі атау бұл – тіл. HTML басқа программалау тілдері сияқты компьтерлік тіл болып табылады. Оның бойында әжептеуір қатал ережелер бар және дұрыс алу үшін осы ережелер сақтау қажет.

Web – беттер компьютер экранында форматталған түрде көрінгенімен, HTML тілі құжатты форматтауға арналмаған, өйткені текст элементтерін дәл позициялау Web – беттік шектеулерге алып келеді, ал бұл Интернетте мүмкін емес.

Мысалы, Word тексттік процессорында текстті форматтаған кеде, біз бұл тексттің нақты бір принтерде және нақты форматтағы қағазда қалай көрінетіндігін дәлме – дәл нақты анықтаймыз. Ал, бұл құжат Интернетте орналастырылған кезде, бұны көру үшін қандай компьютердің қолданылатынын білу мүмкін емес. Мүмкін бұл текст қазіргі заманға сай Windows жүйесінде істейтін браузер бар компьютер экранында көрінетін шыға. Мүмкін ол графиканы көрсете алмайтын, MS DOS жүйесіндегі браузер шығар, ал одан басқа,мүмкін бұл текст мүлдем экранда көрінбейтін қолданушы үшін құжат арнаулы құрылғыға Брайль шрифті шығару мүмкін.

Осы атылған құрылғылардың барлығы үшін құжаттар құру қажет болғандықтан HTML тілі құжат форматын анықтайды. Ол құжатты функциональды белгілеуге қызмет етеді, демек текст фрагменттеріні қызметін анықтауға мүмкіндік береді.

Мысалы, текст атауы кездессін, олай болса, HTML коды осы фрагменттің атау екенін көрсетіп қояды. Мұндай кодты алған соң, браузер бұл атауды үлкен шрифтпен көрсетер, мүмкін экран текст бойынша тегістер, мүмкін құжаттың ойнап жатқан дауысын көбейтер. Дегенмен, HTML тілінде бірнеше форматтау кндалары бар. Бірақ оларды қолдану ұсынылмайды.

**2. 1. 1 Тегтер**

Бастапқы текстті белгілеуді орындайтын HTML тілінің кодтары – тегтер деп аталады. Тегсимволдар жиынтығы. Барлық тегтер “<” (кіші) символынан басталады, “>” (үлкен) символымен аяқталады. Кейде бұл символдар жұбын бұрыштық жақша деп те атайды. Ашу бұрыштық жақшасының (<) кейін тегті анықтайтын кілттік сөз жазылады.

HTML тіліндегі әрбір тегтің арнаулы қызметі бар. Тег атауы үшін кіші әріптерді де, үлкен әріптерді де қолдана беруге болады, алайда көбінесе негізгі тексттен айырмашылығы болу үшін үлкен әріптер қолданылады.

HTML – дің бір тегі құжаттың бір ғана бөлігіне әсер етеді, мысалы абзацқа. Бұл жағдайда тегтер жұбы қолданылады: ашу (<) және жабу (>). Ашу тегі бір эффектті бастайды, ал жабу тегі – оны тоқтатады. Жабу тгтері қисық сызық (/) символымен басталады.

Кейбір тегтер өзінің тұрған жеріне ғана эффект береді. Бұл жағдайда жабу тегі қолданылмайды. Егер тег ішінде HTML тілінде жоқ сөз кездессе, онда ол тег оқылмай кетеді.

Құжат браузер арқылы ашылғанда тегтер көрінбейді олар тек құжаттың қалай көрінетіндігіне әсер етеді.

Тег атрибуттары.

Көбінесе тегтер ішінде өзінің бастапқы эффектісіне әсер ететін атрибуттар болуы мүмкін. Атрибуттар бұл бір-біріне пробел арқылы бөлінген және кілттік сөзден бөлек қосымша кілттік сөздер. Кейбір атрибуттар олардың мәнін қажет етеді. Мәндер кілттік сөздер теңдік белгісімен (=) ажыратылады. Атрибут мәні тырнақшаға алынуы керек. Бірақ көбінесе оларды тастап кетуге де болады. Жабу тегтерінде ешқандай атрибут болмайды.

Программалау тілдерінде **түсініктеме** (программаның бөлігі емес, нұсқау үшін қолданылатын тексттік жолдар) қолдану мүмкіндігі мәлім. HTML тілінде де мүмкіндік бар. Мұнда түсініктемелер арнаулы <!- тегі арқылы басталады. Жабу бұрыштық жақшасының жоқ екендігіне көңіл бөлу керек.осы тегтен кейін кез келген текс түсініктеме деп қарастырылып, экранда көрінбейді. Түсініктемелер --> символымен аяқталады. Түсініктеме ішінде кез келген символдар болуы мүмкін, “>” (үлкен) символынан басқа, бұл түсініктеме ішінде тегтер болмайды деген сөз.

**2.1.2  HTML тілінің құрылымы**

HTML құжаты жоғарыда айтылғандай белгілеу тегтерінің (қарапайым символдар) және негізгі тексттен тұрады.демек, HTML құжаты бұл қарапайым тексттік файл оны құру үшін кез келген тексттік редакторды қолдануға болады, мысалы, Windows 9х құрамындағы Блокнот.

1. HTML құжатының барлығының нақты белгіленген құрылымы болады. Құжат <HTML> тегінен басталып, тиісті </HTML> жабу тегімен аяқталу керек. Бұл тегтер жұбы браузерге алдында тұрған нағыз HTML құжаты екенін көрсетеді.
2. HTML құжаты атау бөлімінен және ол дәл осы ретпен орналасады. Атау бөлімі <HEAD> және </HEAD> тегтерінің арасында жазылады және ол құжат туралы информацияны сақтайды. Сондай – ақ, бұл бөлімде <TITLE> және </TITLE> деген құжаттың “нағыз” атауы орналасатын тегтер болуы керек. Windows жүйесіндегі істейтін браузерлердің көпшілігі бұл тегті браузер терезесінің атауын толтыру үшін қолданылады.
3. Құжат мәтінінің өзі құжат денесінде орналасады. Ал, құжат денесі <BODY> және </BODY> тегтері арасында орналасады.

Осы аталған 4 жұп тег HTML құжатының негізгі құрылымын анықтайды. Бұл кез келген HTML құжатында кездеседі.

Практикада осы негізгі құрылымды тегтерінің орнын олар жоқ болса да білуге болады. Сондықтан, егер <HTML>, <HEAD> және <BODY> тегтері жоқ болса, онда браузер программасы олардың орнын анықтай алады. <TITLE> тегі міндетті болып табылады, алайда онын жоқ болуы қазіргі заманғы браузерлер үшн қате деп есептелмейді. Бірақ Web – беттерді құрған кезде бұл тегтерді тастап кетуді ұсынылмайды, өйткені кейбір ескі браузерлер оны қате деп есептеуі мүмкін. Қарапайым дұрыс құрылған HTML құжаты:

<HTML>

          <HEAD>

                   <TITLE>Құжат аты</TITLE>

          </HEAD>

          <BODY>

              Бұл текстті экранда оқуға болады.

          </BODY>

</HTML>

 Құжаттың функционалды бөлімдерін анықтау

HTML тілі құжаттың функционалды бөлімдерін суреттеуге негізделген. Қарапайым құжаттардың көбіне негізгі функционалды бөлімдер атаулар мен абзацтар болып табылады.

1. HTML тілінде құжаттың ішкі атауларының 6 деңгейін қлдануға болады. Олар <H1> және </H1> ден бастап <H6> және </H6> болып белгіленеді.
2. Қарапйым абзацты анықтау үшін <P> және </P> тегтері қолданылады. Бұл тегтер міндетті емес. Бірақ бұлар бар кезде кез келген браузер абзацтың шекараларын дәл анықтайды.
3. HTML тілінде “қызыл жол” тастайтын тег жоқ, сондықтан текст оңай оқылу үшін браузер абзацтар арасындағы бір бос жол қалдырады. Құжаттарды текстті бөліп тұратын керекті құралы болып көлденең сызықтар саналады, олар құжаттың әрбір бөлімін бөліп тұрады. Горизонтальды сызық <HR> тегі арқылы құрылады. Бұл жалғыз тег және жабу тегін қажет етпейді.
4. Осы тегтердің бәрін қолданған кезде есте сақтайтын нәрсе:

HTML құжаты экранда көрінген кезде барлық сөздер арасындағы бос орындар мен бос жолдар жойылады.

1. Егер жаңа абзац құрмастан келесі жолға өту керек болса, <BR> тегін қолданса болады.

<HTML>

          <HEAD>

                   <TITLE>Құжаттың функционалды бөлімдері</TITLE>

          </HEAD>

          <BODY>

                        <H1>негізгі атау</H1>

                        <H2>келесі атау</H2>

                        <P> Мына жолдар бірге көрінеді, бірақ олар бөлек орналасады

                        <P>Абзацты жабу тегі міндетті емес.  <P> Абзацтың басталу тегі келесі

                             жолға өтуден керектірек

                        <HR>

                        Горизонталь сызықтан кейінгі текст <BR> екі жолға бөлінеді

          </BODY>

</HTML>

**2.1.3 Сілтемелер мен якорьлар**

Web – беттердің басқа бір Web – беттерге сілтемелерінің болуы WWW – тің ең бір тартымды ерекшеліктерінің бірі. HTML құжаттарында гипертексттік сілтеме құру өте оңай. Ол үшін <A> және </A> тегтері қолданылады.

1. Гиперсілтемені құрған кезде HREF = атрибуты болуы міндетті. Ол сілтеменің нұсқайтын URL адресін көрсетеді. Сілтеменің мәтінін <А> және </А> тегтері арасына орналастырады. Көбінесе браузерлерге сілтемелер көк түсті және асты сызылған болып көрінеді. Сілтемені шерткен кезде көрсетілген URL адреске көшу орындалады. Гипертексттік сілтемелер кез келген сол – бетке немесе файлға сілтеуі мүмкін. Сол сілтемені шерткен кезде ол Web – бетке көшу немесе сол файлды көшіру мүмкіндігі ашылады.
2. Егер сілтеме сілтеп тұрған Web – бет басқа Web – түйінде орналасқан болса, онда HREF = атрибутының мәні URL атрибуттының мәні құжаттың толық аты (протокол аты, Web – түйін адресі) болуы керек. Мұндай сілтемелерді сыртқы сілтемелер деп атайды.
3. Егер сілтеме өзі орналасқан Web – түйіндегі Web – бетке сілтеп тұрса, онда ізделінді құжаттың қысқаша орны көрсетілсе жеткілікті. Бұл жағдайда ішкі сілтеме құрылады. Ішкі сілтемелерді қолдану тиімдірек, өйткені бұл жағдай Web – түйінді басқа серверге жылжытқан кезде әрбір құжатқа дұрыстаулар енгізудің қажеті жоқ.
4. Гипертексттік сілтемелер беттің нақты бір аймағына сілтеуі мүмкін, ол үшін сол керекті аймаққа якорь орнату қажет. Якорь да <А> және </А> тегтері қолданады, бірақ HREF = атрибутының орнына NAME = атрибуты болу керек. Бұл атрибуттың мәні – якорь аты. Ол тек латын әріптері және цифрлардан тұруы және пробел болмауы керек.
5. Орнатылған якорьға сілтеу үшін құжат атынан кейін URL адресінің соңында, “#” символымен ажыратып, якорь атын көрсету керек.

<HTML>

          <HEAD>

                   <TITLE>Сілтемелер және якорьлар</TITLE>

          </HEAD>

          <BODY>

                        Операциялық жүйенің стандартты программаларының жаңа версияларын және ең жаңа драйверлерді мына компанияның

                            Web – түйінінде табуға болады.

                        <А HREF = “http://www.microsoft.com/” ></A>

                        <P> Ал мына сілтемеден менің Web – түйініме өтуге болады.

                        <А HREF = “my.htm” ></A>

                        <P> Авторды қайдан табуға болатындығы туралы мына

                        <А HREF = “#address”> жерде айтылған</А>

                        Бұл жерде беттің мазмұны бар

                        <А NAME = “address” > Электрондық почта адресі</А>

                        Горизонталь сызықтан кейінгі текст <BR> екі жолға бөлінеді

          </BODY>

</HTML>

**2.1.4  Бетке суреттер енгізу**

Иллюстрациялар Web – беттердегі безендіруде өте үлкен роль атқарады. Суреттер HTML құжатына енбей, өз алдына бір файлда сақталады., ал браузер арқылы экран бетіне Web – бетпен бірге көрінеді.

1. Құжатқа сурет ендіру үшін <IMG> тегі қолданылады.
2. бұл тегтің құрамында міндетті түрде SRC = атрибуты адресі. Құжат жүктеліп жатқан кезде суретте жүктеліп құжаттың <IMG> тегі орналасқан жерде көрінеді.
3. Кескін Web – бетке өлшемі сақталып көшіріледі. Егер суретті орналастырғанкезде оны қайта масштабтау керек болса, онда суреттің керекті өлшемдерін WIDTH = (ені) және HEIGHT = (ұзындығы) атрибуттары арқылы беруге болады. Бұл атрибуттардың мәні сол – беттегі суреттің ені мен ұзындығын пиксель өлшем бірлігінде анықтайды. Иллюстрацияларды қолданған кезде бірнеше қиындықтарға ұшыраймыз. Біріншіден, Web – беттерді көріп отырған браузер суреттерді кескіндей алмайтын болуы мүмкін, екіншіден, қолданушылардың көбісі жүктелу тезірек орындалу үшін суреттердің көрінуін алып тастайды. Осы жағдайлар туындаған кезде суретте не кескінделген не екенін білуге мүмкіндік беру керек. Бұл үшін альтернативті текст қолданылады.
4. Альтернативті текст бұл шын мәнісінде кескінді суреттеу ғана. Егер браузер белгілі себеппен суретті көрсете алмаса, орнына осы тегті шығарады. Альтернативті текст <IMG> тегінде, арнаулы ALT = атрибутының мәні арқылы беріледі. Кескінді де текст сияқты сілтеме етуге болады. Ол үшін <IMG> тегі, сілтемені анықтайтын <А> және </А> тегтерінің арасында болуы керек. Кескін – сілтеме көк рамканың ішінде түрі текст-сілтеме сияқты түрге өзгереді. Осы жолмен Web – беттерде графикалық өту пернелерін құрады.

**Шрифтің стилін ұйымдастыру**

Тек HTML – дің 4.0 версиясынан бастап құжаттың мазмұны оның безендіруінен толық бөліп қарастыру мүмкіндігі пайда болды, бірақ бұл енді пайда болған концепция, сондықтан бұны көптеген браузерлер түсінбейді. Осы себепті HTML тілі құжатты безендіруге ғана арналған тегтерді қолдануды жалғастырады. Мүмкін келесі версияларда HTML ол тегтерді керек етпейді, бірақ қазір оларды қолдануға болады.

1. Шрифтің өлшемін, түсін және сызығын беру үшін <FONT> тегі қолданылады. Бұл жұп тег ашу және жабу тегтерінің арасындағы барлық текстке әсері бар. <FONT> тегінде ең болмағанда бір атрибут болуы керек. Олар SIZE =; COLOR =; FACE =.
2. SIZE = атрибуты шрифт өлшемін анықтайды. Өлшемінің мәні тек 1 мен 7 аралығында болады. Бұлардың ешқандай санау жүйесінің қатысы жоқ, бар болғаны мән үлкен болған сайын шрифт үлкенірек әріппен жазылады. Жалпы жағдайда ол 3 – ке тең.
3. COLOR = атрибуты шрифттің түсін береді. Ол кілттік сөз арқылы (мысалы, Red - қызыл) немесе RGB жүйесіндегі оналтылық санақ жүйесінде  (мысалы, #FF0000 – бұл да қызыл) берілуі мүмкін.
4. FACE = атрибуты шрифттің сыртқы бейнесін анықтайды. Бұл атрибуттың мәні компьютерде орнатылған шрифттердің бірінің атауы қолданушының компьютерінде қандай шрифт бар екенін болжау мүмкін емес, сондықтан бұлатрибутты тастап кеткен абзал.
5. осы айтылған атрибуттардың мәндерін бүкіл құжатқа қолдану үшін <BASEFONT> тегі қолданылады. Ол жалпы жағдайда қолданылатын шрифттің өлшемін, түсін және бейнесін анықтайды.
6. Шрифттің сызылуы үшін арнаулы тегтер қолданылады.

<В> және </В> тегтері арасында тұрған текстті қалың әріппен шығарады, ал <І> және </І> тегтері курсивті етеді. <U> және </U> тегтері астын сызу, <S> және <S> тегтері текстті сызып тастау.

<HTML>

         <HEAD>

                   <TITLE> Шрифт стилін ұйымдастыру </TITLE>

         </HEAD>

         <BODY>

                   <BASEFONT SIZE=4 FACE= “Arial”>

                   Мына текст жалпы түрде қолданылатын стандартты емес стильді шрифтті қолданады.

                   <P><FONT SIZE=2 FACE=“Times New Roman” COLOR=“Green”>

                   Мына текст кішілеу және басқа шрифт пен түсті қолданады </FONT>

                   <P> <B> Қалың шрифт </B> және <І> курсивті </І> текст фрагменттерін бөліп көрсету үшін қолданылады.

                   <Р> <U> Асты сызылған </U> текстті онша қолданбау керек, өйткені оны сілтемемен шатастырып алуға болады.

                   <P> <S> Сызылып сақталған текстті кейде </S> қажет емес фрагменттер үшін қолданылады

</BODY>

</HTML>

Фраза элементтерін тағайындау.

Бұл бөлімде қарастырылатын тегтер текстті безендіру тегтеріне ұқсас. Айырмашылығы, бұл тегтер тексттің шын қасиетін анықтайды. Бұл мына мағынаны білдіреді: кейбір  браузерлер кейбір шрифттерді әр түрлі көрсетуі мүмкін, ал бұл жағдай егер компьютерде шрифттік эффектілер жоқ болса да олардың кейбір эффектілері дәл көрсетіледі.

Бұлардың барлығы жұп тегтер және олар тек арасында тұрған текстке әсер етеді. Жабу тегі бұл группаның тегтері үшін міндетті.

1. <CITE> тегі тексттің басқа информация көзінен алынған цитата екенін білдіреді және ол курсивпен көрінеді.
2. HTML тілі компьютерге қатысы бар адамдармен ойлап табылған болғандықтан, кейбір тегтер компьютерлік программалардың текстін беруге және олармен қарым-қатынасының нәтижесін айқындау үшін жасалған. <CODE> тегі компьютерлік программалардың бастапқы текстіне нұсқайды. <KBD> тегі клавиатура арқылы енгізілген текстті безендіреді. Бұл тегтерді қолданушының командаларын белгілеуге немесе пернелердің атауын беруге қолданған ыңғайлы. <SAMP> тегі компьютерлік программаның шығысының мысалы болып табылатын текстті қоршайды. <VAR> тегі болса, программалық айнымалылардың атауын безендіру үшін қолданылады. Осы тегтермен белгіленген тексттер әдетте экранда бір енді болып көрінеді. Сонымен қатар, кейбір браузерлер <KBD> және </KBD> арасындағы текстті қалың шрифтпен көрсетеді.
3. <EM> тегі текстті бөліп көрсету үшін қолданылады, ол текст көбіне курсивпен көрінеді. Онда да қатты бөліп көрсету <STRONG> тегі арқылы жүзеге асырылады. Бұндай текст қалың шрифтпен жазылады. Мұндағы бөліп алу безендіруге өте ұқсас, сондықтан <EM> және <STRONG> тегтерін сәйкесінше <I> және <B> орнына қолдану ұсынылады.

<HTML>

<HEAD>

         <TITLE> Фраза элементтері</TITLE>

</HEAD>

 <BODY>

         <P><CITE> АҚШ конгресінің сайтына Стар атаулы прокурорының докладының барлық 445 беті орналастырылған </CITE>  -бұл “Егемен Қазақстан” газетінен цитата.

<P><CODE> Программа тексттері </CODE>, <KBD> клавиатурадан енгізілген командалар <KBD> <SAMP> компьютерлік программалардың шығуларының <SAMP> мысалдары және программалық <VAR> айнымалылар </VAR>  әдетте экранда бірдей болып көрінеді.

<P><EM> Бөліп көрсету </EM>  және <STRONG> қатты бөліп көрсету

         <STRONG> қалың және курсив тексттің мағыналы эквиваленті ретінде қолданылады.

</BODY>

</HTML>

**2.1.5  Тізімдермен жұмыс**

Тексттік процессорлармен жұмыс кезінде екі түрлі тізімдер қолданылады: реттелген (номерленген) және реттелмеген (белгіленген). HTML тілі бес түрлі тізім құруға мүмкіндік береді. Алайда бұлардың екеуі (мәзір тізімі және каталогтар тізімі) ескірген деп саналады және онша көп қолданылмайды. Осылайша қазір тізімнің ү түрі кеңінен қолданылады: реттелген, реттелмеген және анықтамалар тізімі.

1. Реттелген және реттелмеген тізімдер бірдей құрылады. Тізім әрқашан тізім ашу және жабу тегтерінің арасында орналасады. <OL> және </OL> - реттелген, <UL> және </UL> - реттелмеген тізім. Тізім ішінде тізім элементтері орналасады. Оларды <LI> және </LI> тегтерінің арасына алады, алайда бұл жағдайда жабу тегін тастап кетуге болады, өйткені оның жабылатын орны белгілі.

<OL>, <UL> және <LI> тегтерінде TYPE= атрибуты болуы мүмкін. Оның мәні тізім элементтерін белгілеу әдісі. Реттелген тізімдерді арабю цифрларымен, рим цифрларымен, латын әріптерімен (кіші және үлкен) номірлеуге болады. Реттелмеген тізімдер арнаулы маркерлер арқылы белгіленеді.

Тізімдер ішкі де болуы мүмкін. Сонымен қатар, тізім элементтерінде гиперсілтемелер, фраза элементтерін бөліп көрсетуге немесе форматтауға арналған тегтер болуы мүмкін.

1. Анықтамалар тізімі <DL> тегімен басталып </DL> тегімен аяқталады. Бұл тізімдерде маркерлер және нумерация қойылмайды. Ол анықталатын терминдерден және анықтамалардан тұрады. Біріншісін - <DT>, екіншісін - <DD> тегтері арқылы белгілейді. Жабу тегтерін тастап кетуге болады. Анықтамалар экранда сол жақ шеттен бір бос орын кейін шығады.

<UL>

<LI> Сыртқа

<OL>

<LI> Принтер

<LI> Сканер

<LI> Монитор

</OL>

<LI> Ішкі

</UL>

<DL>

<DT> <STRONG> MIDI </STRONG>

<DD> Компьютерге музыкалық инструменттерді қосу стандарты

<DT> <STRONG> MPC </STRONG>

<DD> Мультимедиялық компьютер қажеттігін білдіретін стандарт

<DT> <STRONG> OLE </STRONG>

<DD> Объектілерді безендіру және байланыстыру технологиясы

<DT> <STRONG> PLUG AND PLAY </STRONG>

<DD> Жасаушы компания және модель туралы информациясы бар, сонымен қатар программалық жабдықтар арқылы икемденуге болатындығын білдіретін стандарт

</DL>

**2.1. 6  HTML-дің  мультимедиялық объектілері**

Мультимедияның кеңінен таралуы WWW қызметінің бар кезіндегі уақытқа сәйкес келеді, сондықтан мультимедиялық объектілерді Web-беттерде орналастыру бірден қалыптаса қойған жоқ. Аудио және видео файлдар әлі күнге дейін “Сыртқы” объектілер ретінде қарастырылады, демек оларды көріп, тыңдау үшін көмекші программалар қажет.

Мультимедиялық файлдар бірнеше форматта болуы мүмкін. Қарапайым форматтағы файлдарды тыңдау үшін, оларды толығымен жүктеу қажет. Ал, қазіргі заманғы үздіксіз форматтар аудио мен видео кескінді жүктеліп жатқан кезінде – ақ көрсете беруге мүмкіндік береді.

Интернетте үздіксіз форматта радиолар жұмыс істейді. Мысалы, осы форматта “Серебрянная дождь” радиостанциясы жұмыс істейді ([www.silver.ru](http://www.silver.ru/)).

Web-бетке мультимедиялық объектті кірістірудің ең оңай жолы – гиперсілтеменің <A> тегін қолдану. Бұндай сілтемені құру оп – оңай және де бұл жағдайда мультимедиялық файл оқушының көңілін бөлмейді.

Кейбір жағдайларда дыбыстық файлды Web – беттің өзіне кірістіруге тура келеді, мысалы, музыканы автоматты түрде ойнатқан кезде. Бұл үшін ең қолайлысы <EMBED> тегін қолдану, алайда бұл тег HTML стандартына кірмейді және бұл тегті тек екі ең танымал браузерлер Internet Explorer және Netscape Navigator түсінеді.

<EMBED> тегі жалғыз тег болып есептеледі, сондықтан жабу тегі қажет емес. Оның міндетті атрибуты SRC= , бұл атрибуттың мәні – қажетті мультимедиялық файлдың орналасқан адресі. Сонымен қатар WIDTH= және HEIGHT= атрибуттарын қолданунға да болады. Бұлар ойналынатын файлдың ені мен ұзындығын анықтайды. Мысалы,

<HTML>

<HEAD>

         <TITLE> Документтегі мультимедиа</TITLE>

</HEAD>

<BODY>

         <P><A HREF= “music.wav”> Музыка тыңдау үшін осы жерді шертіңіз </A>

         <P> Төртбұрыш рамканың ішінде көрінетін видео

<EMBED SRC= “File.avi” WIDTH= “200” HEIGHT= “200”>

</BODY>

</HTML>

**2.1.7  Кесте құру**

Кесте бұл үлкен көлемді берілгендердің көрінісінің ең қолайлы әдістерінің бірі. HTML тілінде кестелердің түрлі типтерін қруға мүмкіндік бар.

1. HTML тілінде кесте <TABLE> тегімен басталып </TABLE> тегімен аяқталады. Кесте элементтерін анықтайтын текст кесте ішінде арнаулы тегтер арқылы бекітіледі (тақырыптар, қатарлар және ұяшықтар).
2. Кесте тақырыбын анықтауышы <CAPTION> және </CAPTION> тегтері <TABLE> және </TABLE> тегтерінің аралығында кездесуі мүмкін. Тақырып кестенің үстіңгі не астыңғы жағында жазылуы мүмкін.
3. Мұнан әрі  <TR> және </TR> тегтері арқылы жалғасады, бұлар кесте қатарын анықтайтын тегтер. Жабушы тегті қатардың соңында, яғни келесі қатар басталар кезде немесе кестенің жабушы тегімен бірге жазуға болады.
4. Кестенің әрбір қатары ұяшықтардан құралады. Ұяшықтарды келесі тегтермен орналастыруға болады:  <TH> - баған және қатар тақырыптарынан құралған, <TD>  - берілгендерден құралған. Бұл тегтерде жұптасып кездеседі, бірақ жабушы тегтерді дәл осында да жазылуы мүмкін, бұл ешнәрсені өзгертпейді.
5. Ұяшық ішінде документ денесінде кездесетін кез келген берілгендер мен HTML – дің тегтері болуы мүмкін. Мысалы, Т ұяшығына ішкі кесте салуға болады. Браузерлер автоматты түрде ұяшықтар мен кестелердің өлшемдерін есептеп қояды, сондай ақ атрибуттар көмегімен бұл қасиеттерін бөлек - бөлек тапсыруға да болады.
6. Кестенің арнаулы атрибуттарының ішінен, мысал үшін  BORDER= атрибутын алайық, бұл атрибут жеке ұяшық пен кестенің айналасындағы рамкаларды құру мүмкіндігін береді. Бұл атрибуттың мәні ішкі рамканың пиксельдегі қалыңдығы болып табылады.
7. Төменде көрсетілген мысалда, басқа да атрибуттар пайдаланылған, осы атрибуттар көмегімен ұяшықтар арасындағы аралығын кеңейтіп және ұяшықтардың биіктіктері құрылған.

Мысалы,

<HTML>

         <HEAD>

                   <TITLE> Кесте </TITLE>

         </HEAD>

         <BODY>

         <TABLE WIDTH= “90%”  BORDER= 10  CELLPADDING=2 CELLSPACING=10>

<CAPTION> Кестенің негізгі тегтері </CAPTION>

<TR> <TD> <B> Ашушы тег </B> <TD> <B> Жабушы тег</B>

<TD> <B> Сипаттама </B>

<TR> <TD> TABLE <TD ROWSPAN=2> Міндетті түрде <TD>

Кестенің басы мен соңы

<TR> <TD> <CAPTION> <TD> Кесте тақырыбының басы мен соңы

<TR> <TD> TR <TD ROWSPAN=2> Міндетті емес <TD>  Кесте қатарының басы

<TR> <TD> TD <TD> Кесте ұяшығының басы

</TABLE>

</BODY>

</HTML>

2.1. 8 Фреймдер

HTML тілі браузер программасының терезесін бірнеше бөліккке бөлуге және оның әрқайсысында жеке документті бейнелеуге көмектеседі. Мұндай аймақтар фрейм деп аталады.

1. Фреймдерді құру үшін құрылымы кәдімгіден ерекше HTML – дің ерекше документін қолданады. Мұндай құжатта документ “денесі” болады, мұнда ешқандай текст болмайды. Мұның орнында мұнда <Frameset> және </Frameset> тегтері арасында қорытылған фрейм мазмұны болады.
2. <Frameset> тегінде COLS = немесе ROWS= атрибуты болады, бұл терезені бөлу тәсілін анықтайды. COLS = атрибутын қолданған уақытта терезе аймақта вертикальды сызықтармен бөлінеді, ал ROWS= атрибутын қолданғанда горизонтальды сызықтармен бөлеміз. Егер екі атрибуты берілсе терезеде аймақ асты торы құрылады.

        Бұл атрибуттың мәні терезе аймағының ұзындығын анықтайды. Әр бағанға арналған параметрлер пикселге ендірілген күйде немесе пайызда (% символы) береді. Соңғы параметр ретінде “жұлдызша” (\*) символын қолдануға болады. Мұндай фреймнен барлық бос кеңістіктер бөлінеді.

1. <Frameset>, </Frameset> тегтері арасында қосымша тегтерді қосады, бұлар құрылған аймақтар мәнін көрсетеді. Бұл мақсат үшін <Frameset> қосымша терезелердің бөлінуін беретін тегті қолдануға немесе жеке аймақтарға енетін құжаттарды анықтайтын тегтерді қолданады. <Frameset> және </Frameset> тегтері арасындағы элемент саны құрылған аймақ мөлшеріне сәйкес болу керек.
2. <Frame> тегінде берілген аймаққа енетін құжатты анықтайтын SRC= міндетті атрибуты болуы керек. Қосымша атрибуттар жеке фреймдер арасындағы рамкаларды басқаруға және кейбір басқа қасиеттерді басқаруға пайдаланылады.

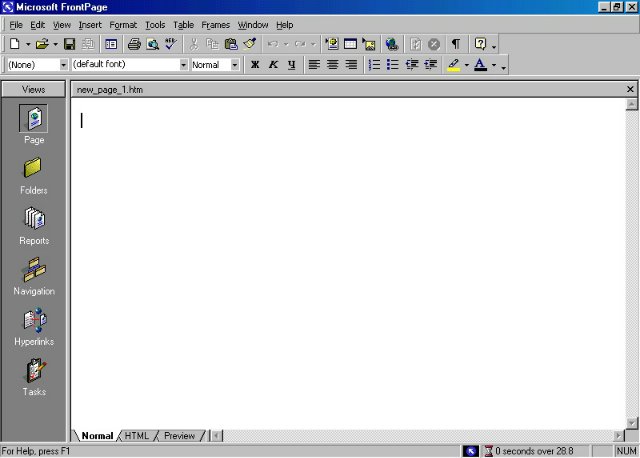
2. 2  Front  Page программасы

Электронды оқулық дайындаудың қолайлы да белгілі әдістерінің бірі Front – Page программасы болып табылады.  Front Page Web-беттер  мен Web-т‰йіндер құруға арналған  Mіcrosoft Offіce 2000 құрамына кіретін программа. аспаптар тақтасы мен мәзірдегі командаларды пайдаланып бетке мәтіндік және графикалық информациялар, навигциялар тақтасы мен бір беттен екіншісіне өту үшін қолданылатын сілтемелер орналастыруға болады. Front Page-дің басқа да кестелерді, фреймдерді, жүгіртпе жолдарын, түрлі анимцияларды және т. б. пайдалану мүмкіндіді бар. Web-беттер мен Web-түйіндерді құруды жылдамдату  үшін шаблондар мен шеберді пайдалануға болады. Front Page программасы Web-тұйінге Mіcrosft Offіce-тің құрамына кіретін Word, Excel, Access  және Power Poіnt сияқты программаларда дайындалған құжаттарды енгізуге мүмкіндік береді.

Front Page  программасын іске қосу үшін орындалатын әрекеттер: Пуск ® Программы  ®  Mіcrosoft Front Page

Негізгі терезе

Front Page программасы іске қосылғанда экранда оның негізгі терезесі ашылады (1-сурет).



1-сурет. Front Page программасының негізгі терезесі.

**2.2.1** **Стандартты аспаптар тақтасы**

Front Page программасында қолданылуына байланысты әртүрлі түймелер жиынтығынан тұратын кұптеген аспаптар панелі бар. Қандайда бір жұмыс режиміне көшетін болсақ, программаның негізгі терезесінде қажетті аспаптар тақтасы пайда болады. Мысалы, Web-түйінді кұру режимі мен есеп құруға өткенде Reportіng тақтасы іске қосылады, ал графикамен жұмыс істегенде Pіctures тақтасы қосылады.  Аспаптар тақтасын экранға шығару үшін Vіew(Тұр) мәзіріндегі Toolbars (аспаптар тақтасы) командасын орындау керек. Бұл команда орындалғанда  Front Page программасының барлық тақталар тізімі  шығады (1-кесте).

1-кесте. Front Page программасының аспаптар тақтасы.

|  |  |
| --- | --- |
| **Аспаптар тақтасы** | қызметі |
| Standard  Formattіng  **DHTML Effects**    Navіgatіon      Pіctures    Posіtіonіng      Reportіng      Style    Table | Стандартты аспаптар тақтасы.  Форматтау тақтасы  Web-бетте динамикалық HTML-эффектісін қрады.  НавигЦия тақтасы, Navіgatіon   режимінде жұмыс істеуге  арналған.  Суреттермен жұмыс істеуге арналған аспаптар тақтасы.  Объектілерді позициялауға мүмкіндік беретін түймелерден тұрады.  Web-түйіндегі қрастырылатын есеп типтерін жасауға  мүмкіндік беретін аспаптар тақтасы.  Безендіру стилін жасауға арналған аспаптар тақтасы.  Кестелерді құру мен модификциялауға арналған түймелерден тұрады. |

Мәзірдегі керекті тақтаны таңдағанда оның сол жағында белгі (V) пайда болып, кажетті тақта негізгі терезеде көрінеді. Экрандағы керек емес тақтаны алып тастау үшін мәзірдегі сәйкес келетін команданы қайта таңдау қажет, сонда ол  Front Page программасының негізгі терезесінде  көрінбей қалады (2-сурет).

|  |
| --- |
|  |
|  | http://s020.radikal.ru/i714/1312/80/bf046fd32aba.jpg |

2-сурет. Front Page программасының аспаптар тақтасы.

Аспаптар тақтасының тізімін экранда кұру үшін курсорды Front Page программасының негізгі терезесінде орналасқан кез-келген аспаптар панеліне қойып, тышқанның оң жақ түймесін басу жеткілікті.

http://s020.radikal.ru/i716/1312/55/fd856b2a1624.jpg  
 3-cурет. Front Page программасының барлық режимінде қолданылатын аспаптар тақтасы.

**2.2.2. Жұмыс режимі тақтасы**

Front Page  программасында Web-түйінді кұру мен қрудың 6 тұрлі режимі бар. Осы режимдердің біріне өту үшін негізгі терезенің сол жағында орналасқан  6  батырмасы бар Vіews  жұмыс режимі тақтасы қолданылады (2-кесте).

2-кесте. Жұмыс режимі тақтасының түймелері.

|  |  |  |
| --- | --- | --- |
| Түйме | аталуы | Қызметі |
| http://s019.radikal.ru/i617/1312/ea/558e588019ee.jpg | Page          Folders    Reports    Navіgatіon    Hyperlіnks      Tasks | Беттерді  кұру және редакциялау. Берілген режимде Front Page программасының жұмыс облысында бұрыннан бар Web-беттерді редакциялауға немесе жаңасын құруға болады.  Web-түйінде қрылған бумалардың  құрылымын кұру.  Есептерді кұру. Бұл режим Web-түйіндегі есептерді құру мен кұруға арналған.  Web-түйін құрылымын жобалау.    Web-түйін гиперсілтемелерін кұру.    Web-түйін жасау кезінде тапсырмаларды құру және кұру режимі. |

Front Page программасы іске қосылғанда бірден редакциялау режимін пайда болады. Программаның негізгі терезесінің сол жағында жұмыс режимі панелі орналасқан. Үндемеген жағдайда Page(Бет) орналасады. Аспаптар панелі мен мәзір командаларын қолданып жаңа Web-беттер құруға, бұрынғыларын кұруға және оларды редакциялауға болады.

Page режимінде жұмыс облысының төменгі жағында Web-беттерді түрлі режимде кұруға мүмкіндік беретін 3 қосымша   орналасқан (3-кесте).

3-кесте. Web-беттерді кұру режимі.

|  |  |
| --- | --- |
| **Қосымша** | Қызметі |
| Normal            HTML            Prevіew | қарапайым жұмыс режимі, беттерді қруды WUSІWSІG  (“не кұріп тұрсаң, соны аласың”) режимі кұмегімен қаматамасыз етеді. Бет мәтіндер, сызықтар, тұймелер, сілтемелер және т. б. мәзір мен аспаптар тақтасы пайдаланылып қрылады, бұл жағдайда  HTML-кодын білу міндетті емес.  Беттерді HTML-кодында кұру және құру. Бұл режимде қарапайым жұмыс режимінде қрылған беттің HTML-кодын кұруға және де кодтар көмегімен бет құруға болады. Бұл жағдайда беттің бір бұлігі қарапайым режимде, ал қалғаны кодталып жазылуы мүмкін.  Беттердің Интернет серверінде қалай көрінетінін алдын ала кұру. |

**Web – бетке графикалық объектілерді орнату**

|  |
| --- |
|  |
|  | http://i069.radikal.ru/1312/15/71b1fad2d26c.jpg |

Сондай –ақ Web – бетке графикалық объектілерді орнатып оларды өңдеуге болады. Web-бетіне графикалық объектіні қою үшін курсор графикалық объект қоятын жерде  қойылады. Саймандар тақтасынан  Іnsert Pіcture(Суретті қою) пернесі шертіледі немесе ІnsertäPіctureäFrom Fіle(ҚоюäСуретäФайлдан) командасы беріледі, нәтижесінде  суреттеді диалог терезесі ашылады. Қойылатын графикалық файл ағымдағы Web-түйінің тізімінен таңдалынады. JPEG немесе GІF  файлдарын көру үшін файл ерекшеленеді, FrontPage алдын-ала көру аймағында сурет көрінеді.

4 -Сурет. Pіcture(Суретті қою) диалог терезесінен графикалық файлды таңдау.

http://s020.radikal.ru/i721/1312/34/ab3e185d9561.jpgЕгер сурет ағымдағы Web-түйіннің сыртында болса, онда Use Yore Browser to Select a Page (Бетті таңдау ұшін браузерді қолдану) немесе Select a Fіle on YourComputer (Бұл компьютерде файлды таңдау) пернелерін  шерту арқылы WWW-дан немесе  дербес және желілік дискілерден графикалық  файлдарды іздеу процедурасы іске қосылады. Файл таңдалынғаннан кейін OK пернесі басылады.

2.2.3 Графикалық форматтардың түрлері

Web-беттерінде графикалық объектілер ақпараттық мәлімет немесе дипперсілтеме түрінде қолданылады. Сондай-ақ гиперсілтемемен мақсатты ресурсы болып келетін графикалық файлға нұсқауға болады, оны шерту арқылы көрсеткенде осы файл браузерде ашылады.

Графикалық оъектілер әшекелеудің бағалы элементі болып табылады және біркелкі мәтіндік бетке өң береді.

Web-бетінің бөліді болып келетін драфикалық объектілер  орнатылған драфикалық объектілер деп аталады. Соған байланысты HTML мәтіндік ақпаратты көрсету  тілі болғандықтан, орнатылған графикалық объектілер бөлек файлдарда сақталынады және Web-бетімен бірде браузерде ашылып  көрсетіледі.

Компьютерлік графикалық файлдар  әртұрлі форматта берілген. Файл форматы файлда ақпаратты сақтаудың әдісін анықтайды. Драфикалық файлдың екі форматы Word Wіde Web -те  стандарт болып келеді, бұл GІF  және JPEF  форматтағы файлдар.

Бұл форматтар графиканы сығуды қолданады, сондықтан да олардың  сығуды қамтамасыз етпейтін  басқа форматтағы файлдарға қарағанда  өлшемі аз. Бұл мінездеме Интернет сияқты  желіде  файлдары жіберу   кезінде  қатты ескеріледі, өйткені файл өлшемі неғұрлым кіші болса соғырлым тез жіберіледі және жұктелінеді.

Web - бетіне FrontPage  редакторымен драфикалық объектіні орналастырғанда басқада драфикалық форматтағы файлды қолдануға болады, олардың ішіне: BMP, TІFF, TGA, RAS, EPS, PCX, PCD   және WMF кіреді. Web- бетін сақтаған кезде FrontPage драфикалық объектіні JPEG немесе GІF файлдарына өздертеді.

**Web – беттерін безендіру.**

Web – беттерін безендіруде тұрлі анимациялар мен кұрделі элементтерді қолдануға болады.

Кұны кеше Web-беттерде динамикалық объектілерді, курсорды орнатқанда өзінің түрін өзгертетін түймелерді, мәтіндік жүгіртпе жолдарын, анимацияларды орнату туралы тек армандауға ғана болатын. Қазір бұл Web-түйіндерді құруда қолданылатын, жаңа технологияның арқасында мүмкін болды.

Web-беттерде объектілерді орнату арқылы беттерді қолданушылар үшін тартымды, көрікті және тірірек етуде мүмкіндік бар. Жарнамалық баннерлер менеджері арқылы Web-беттерде әшекейлер, және ең бастысы қандайда  бір нәрсені (мысалы, тауарды) жарнамалауға қолданылатын – слайдтарды орналастыруға болады. Web-түйінде Ұсынылған информацияны түрлендіру үшін фреймдер көмегін қолдануға және бетке орнатылған сілтемені таңдау арқылы оны жаңартуға болады.

**2.2.4 Динамикалық  эффектілер**

Front Рage продраммасымен жұмыс істейтін Web-түйін құрушыға беттерді қызықты етіп жасау мүмкіндідін беретін әртүрлі әдістер ұсынылады. Мұндай әдістерде ең бірінші Dynamіc HTML (Динамикалық HTML-DHTML) жатады. Динамикалық эффектілер құру ұшін Java, Java Serіpt немесе VBScrіpt тілдерін білу міндетті емес. Web-түйіндер құрғанда DHTML-ді кеңінен қолдануға шек келтіретін жағдай - қолданушылар Web-беттерді қарағанда пайдаланылатын шолушы продраммалардың барлығы бірдей берілден эффектілерді қабылдамайды. Іnternet Explorer-де құрылған барлық эффектілерді, Netscupe Navіgator қолданушылары көре алмайды, және керісінше. Біз уақыт кетіріп қажетті эффектілерді алғанымызбен оны Web-түйінді қараушылар көрмеуі де мүмкін.

2.2.5 Жүгіртпе жол

|  |
| --- |
|  |
|  | http://s019.radikal.ru/i616/1312/9b/0e3b60784562.jpg |

Front Рage программасы арқылы құрылатын HTML динамикалық элементтерінің бірі-тікбұрыш облыста орналасатын мәтіндік жол түрінде безендірілден, "жүгіртпе жол".  Жүгіртпе жол орналастырғымыз келген тексті белгілеп алу. Іnsert (Кірістіру) мәзірінен Component (Компонент) командасын таңдап, ашылған мәзірден Marquee (Жүгіртпе жол) опциясын таңдау. Marquee Propertіes сұхбат терезесі (5-сурет) ашылады, онда Web-бетке орналастырылатын объекті параметрлерін орнатуға болады.  Сұхбат терезенің жоғарғы жағында экранда жүгіртпе жол түрінде ұсынылатын мәтінді енгізуде арналған Text жолы орналасқан. Осы жолға қажетті сөйлемді енгізу.

Осы ашылған сұхбат терезесінде өзімізге қажетті қасиеттерін орнатып OK түймесін шертеміз.

**2.2.4 Бетке анимация ұйымдастыру**

Front Page продраммасы Web-бетті безендіргенде, Mіcrosoft Offіce пакеті құрамындағы презентацияларды дайындауға арналған Power Poіnt программасы сияқты түрлі анимациялық эффектілерді қолдану мүмкіндігін береді. Анимация көмегі арқылы бүкіл абзацты тірілтіп, олардың түрлі жылжу бағыттарын беруге болады. Анимациялар эффектілерін құрғанда DHTML Effects аспаптар панелі (6-сурет)  қолданылады.  Оны программаның негізгі терезесінде орнату үшін келесі әрекеттердің бірін орындаймыз:

* Format (Формат) мәзірінен Dynamіc HTML Effects (Динамикалық HTML-эффект) командасын таңдау.

|  |
| --- |
|  |
|  | http://s020.radikal.ru/i712/1312/8c/b5b4217f524a.jpg |

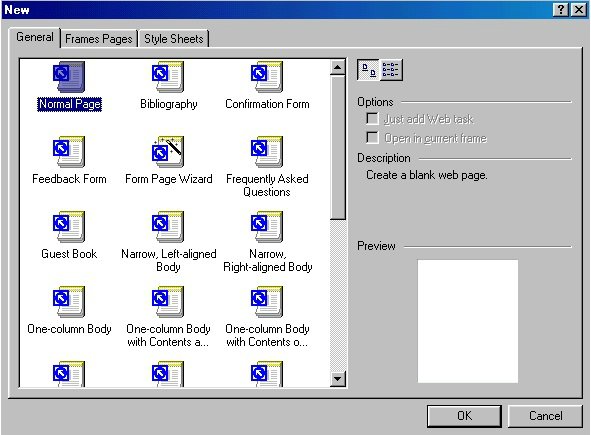
Vіew (Тұр) мәзірінен Toolbar (аспаптар панелі) командасын орнатып, ашылған мәзірден - DHTML Effects опциясын таңдау.

6-сурет**.** DHTML Effects аспаптар панелі.

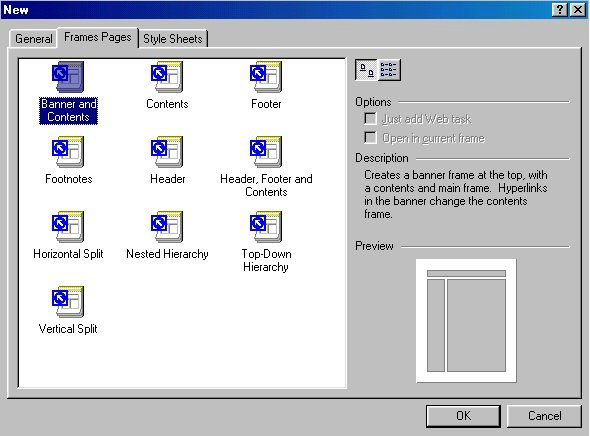
Аспаптар панелінде үш тізім орналасқан. On (Жағдай) тізімі таңдалғанда динамикалық HTML-эффект орындалатын оқиғалардан тұрады.

**2.3 ОБП технологиясы жайлы Web түйіннің дайындау**

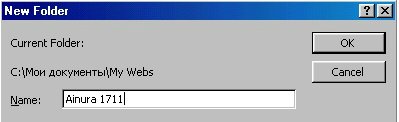
Front Page бағдарламасын іске  қосу үшін Пуск ® Программы  ®  Mіcrosoft Front Page командасын орындап, терезе ашылған соң File-> New -> Page  командасын орындаймыз. Сол кезде төмендегідей терезе пайда болады:



Frame Pages жақтаушасын ашып, фреймнің Banner and Contents силін таңдаймыз:



Жұмыс аймағындағы әрбір New Pages түймелерін шертіп, олырды сақтап аламыз. Сақтау барысында қай фреймді сақтап жатқандығы көрініп тұрады. Бірінші бума атын берілуі тиіс:



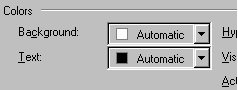
Осылайша сол жақтағы, оң жақтағы сосын бүкіл құжат сақталады.

         Adobe PhotoShop бағдарламасын іске қосып, фон түсін таңдап, фон түсіндегі жаңа құжат құрамыз. Оның өлшемі 600х150. Бұл біздің мазмұнымыздың фоны болып табылады. Суретті әсемдеу мақсатында мұнда басқа құжаттың бөліктерін қырқып алып қоюға болады:

Ал фондағы жұлдызшалар болса қыл қаламның көмегімен салынған. Енді осы суретті құжатқа фон ретінде кірістіреміз. Ол үшін Format -> BackGround командасын орындап, суреттің адресін көрсетіп С/Мои документы/My Webs/Ainura 1711/F1.jpg ОК түймесін шертсек сурет фон ретінде түседі.

         Келесі фреймдерге дайын фондарды да қолдануға болады. Ол үшін өзімізге ұнаған фондарды бумаға сақтаймыз да жоғарыдағыдай іс әрекеттер орындаймыз. Мысалы, интернеттен алынған фон орнататын болсақ та осындай іс әрекет орындаймыз:

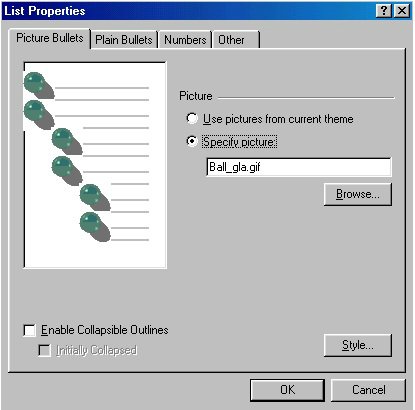
Front Page бағдарламасының тек бір түсті фон ретінде орнату мүмкіндігі де бар. Format -> BackGround командасын орындап пайда болған терезеден



BackGround тұсын бассақ түстер палитрасы ашылады:

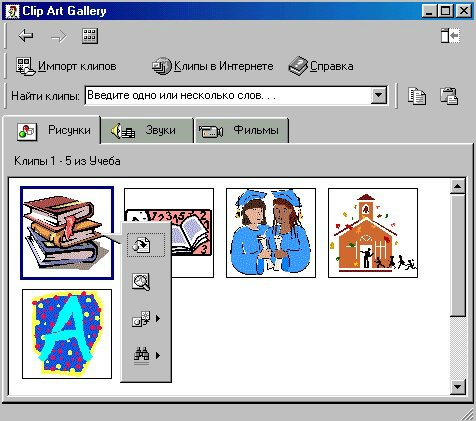
Егер мұнда ұсынылған түстерден басқасын таңдағыңыз келсе, More Colors түймесін басыңыз, сонда төмендегідей қосымша палтра ашылады:

Беттті дайындау барысында тізімдермен жұмыс істеуге тура келеді. Ол үшін дайын маркерлер  қолдана аламыз немесе арнайы дайындалға маркерлер қоюға болады. Екінші жағдай үшін маркер суретін тауып, оны бумаға сақтаймыз. Format -> Bullets and Numbering командасын таңдап, пайда болған терезеден Specify Picture белгісін шертіп,  Browse… түймесін басамыз, буманың адресін беріп суретті белгілейміз де ОК түймесін басамыз.





Бетке сурет енгізуге болады, яғни Clip Art – тан немесе файлдан (From File). Ол үшін Insert - Picture - Clip Art немесе Insert - Picture - Clip Art From File командаларын орындау керек.



Файлдан қоятын болсақ:

Сонымен қатар бетке анимациялық суреттерді де қоюға болады. Оны құжатқа кірістіру файлдан сурет кірістіру тәсілі сияқты. Мысалы:

Енгізілген тексттердің түстерін өзгерткіңіз келсе мәтінді белгілеп саймандар қатарындағы  белгісін басамыз.

Бетке  жүгіртпе жол енгізгіміз келсе жүгіртпе жол орналастырғымыз келген тексті белгілеп алу. Іnsert (Кірістіру) мәзірінен Component (Компонент) командасын таңдап, ашылған мәзірден Marquee (Жүгіртпе жол) опциясын таңдау. Marquee Propertіes сұхбат терезесі ашылады, онда Web-бетке орналастырылатын объекті параметрлерін орнатуға болады.  Сұхбат терезенің жоғарғы жағында экранда жүгіртпе жол түрінде ұсынылатын мәтінді енгізуде арналған Text жолы орналасқан аймаққа қажетті сөзді орналастыру керек:

Осылайша әртүрлі безендірілген бірнеше беттер дайындап аламыз. Негізгі фреймге білім  қорының мазмұнын жазамыз да әрбір тақырыпқа гиперсілтемелер жасаймыз. Ол үшін жазуды белгілеп Ctrl+K немесе Insert -> Hiperlink командасын орындаймыз да сілтеме жасалатын бетті көрсетеміз.

Қорытынды

**Объект ұғымы -**Объектілі бағдарлы программалаудың ең үздік технологиясы  және модификациясы жеңіл программа жасауға мүмкіндік береді. Оның мүмкіндіктерін ОБП тілінде қолдануға болады. Ондай тілдердің бірі Simula 67 , ол 60 – шы жылдардың екінші жартысында пайда болды. Қазіргі программалау тілдері ОБП-ны қолдайды. Оларға С++ , Java, Delphi т.б жатады.

ОБП- ның келесі негізгі түсінігі – **операция**. Операция дегеніміз берілген кластың объектісіне қолданатын функцияны айтамыз. Мысалы , үшбұрыштың геометриялық өзгерістері бір операцияны әртүрлі кластың объектісіне қолданады, оны полиморфтық операция деп атайды.

Объектiге енетiн программалар :

Процедуралар;

Функциялар;

Конструкторлар;

Деструкторлар түрiнде iске асады.

Объектке енгiзiлген программалар әдiстер деп аталынады және сонымен қатар объект берілгендер мен әдістерден тұрады. Объект Turbo Pascal-да берiлгендердi сипаттау бөлiмiнде жазылады. Ал айнымалыларды сипаттау бөлiмiнде объект экземплярлары берiледi.

Дипломдық жұмыстың кіріспе бөлімінде көрсетілген мақсат пен иіндеттерге сай келесі жұмыстар орындалды:

1. Объектілі-бағдарлы программалауға қажетті мәліметтер жиналды;
2. Инкапсуляция, мұрагерлік, полиморфизмнің мағыналарына түсініктеме берілді;

Пайдаланылған әдебиеттер

1. А.С. Немнюгин. Л. Перколаб “Изучаем Turbo Pascal”.  Питер. 2002.
2. А. Файсман. Профессиональное программырование на языке Турбо Паскаль.

3.Нұрмұханбетова Г.К, А. Әмiрбекұлы, М. Сүлейменова

Паскаль тiлiнен лабораториялық практикум. Шымкен. 2002

4. В.В. Фаронов.  “Turbo Pascal 7.0”.  Издательства МГТУ. 1992.

5. А.И. Марченко, Л.А. Марченко. Программирование в среде Turbo Pascal 7.0. Москва. 1999

6.О. Камардинов. Паскаль тiлiндегi программалау. Шымкент. 2001

8.

1. Молли Э. Хольцаг. Использование HTML 4. Москва. 2001.
2. С. Симонович., Г. Евсеев., Специальная информатика. М. 2002.
3. Р. І. Қадірбаева, Л. Ә. Сүлейменова. Microsoft Front Page. Шымкент. 2000.
4. В. А. Острейский. Лабораторный практикум по информатике. М. 2003.
5. Глушаков В. Программирование в Visual Basic. М. 2000