Дәріс 4

Тілдердің препроцессорлық құралдары, макроанықтамалар және макрокеңейтулер. Стандартты кітапханаларды қолдану. Объектіге бағытталған программалау кластар және әдістер түсінігі және сипаттамасы.

Лекция жоспары:

1. Стандартты кітапханаларды қолдану
2. Тілдердің препроцессорлық құралдары
3. Объектіге бағытталған программалау кластар және әдістер түсінігі

Лекция мазмұны

1. Стандартты кітапханаларды қолдану

Турбо Паскальда 8 стандартты модуль бар. Оларда көптеген әртүрлі типтер, тұрақтылар, процедура мен функциялар бар. Бұл модульдер: SYSTEM, DOS, CRT, PRINTER, GRAPH, OVERLAY, TURBOS , GRAPHS.

GRAPH, TURBOS және GRAPHS модульдері жеке TPU-файлдарға бөлінген, ал қалғандары TURBO.TPL библиотекалық файлына енгізілген. Тек бір модуль SYSTEM ғана кез-келген программаға автоматты түрде қосылады, ал қалғандарына USES-те жарияланғаннан кейін ғана қол жеткізуге болады.

Төменде стандатты модульдердің қысқаша сипаттамасы келтіріледі.

SYSTEM модулі. Оған стандартты Паскальдың барлық процедуралары мен функциялары енеді, сонымен бірге басқа стандартты модульдерге енбеген встроенный(қоса салынған) процедуралар мен функциялар да енеді(мысалы, INC, DEC, GETDIR т.с.с.). Бұл модуль USES-те жарияланбаса да кез-келген программаға қосылады. Сондықтан, оның глобал тұрақтылары, айнымалылары, көмекші программалары Турбо Паскалға қоса салынған (встроенный) деп есептеледі.

PRINTER модулі. Тексттерді принтерден шығаруға мүмкіндік береді. Бұл модульде TEXT типті LST файлдық айнымалысы анықталады. Ол логикалық PRN құрылысымен байланыстырылады. Модульді қосқаннан кейін мысалы, мынадай программа орындалу мүмкін:

Uses Printer; begin

writeln (LST, 'Турбо Паскаль') end.

CRT модулі. Тексттік режимдегі экранды басқаруға арналған процедуралар мен функциялар жинақталған. Модульге енгізілген көмекші программалар арқылы курсорды экранның қалаған позициясына жылжытуға, шығарылатын символдардың түсін олардың айналасындағы фон түсін өзгертуге, терезелер құруға болады. Бұлардан бөлек модульге клавиатураны «көзсіз» оқу және дыбысты басқару процедуралары енгізілген.

GRAPH модулі. Графикалық режимдегі экранды басқаруға арналған процедуралар мен функциялар, тұрақтылар мен типтерге ие. Модульге енгізілген көмекші

программалар арқылы әртүрлі графикалық бейнелер құруға және экранға текстік жазбаларды стандартты немесе программист дайындаған шрифтпен шығаруға болады. Көмекші программалар сәйкес баптаулар жасаған соң түрлі типтегі графикалық аппараттарды қолдай алады. Баптау жасау арнайы программа-драйверлермен жүзеге асырылады. Олар GRAPH. TPU файлына енбейді, бірақ солармен бірге қойылады.

DOS модулі. Модульде MS- DOS операциялық жүйесінің дисктік құралдарына қол жеткізуге мүмкіндік беретін процедуралар мен функциялар жинақталған.

OVERLAY модулі. Ол қиылыса жабуды қолданатын ірі программаларды дайындауда қажет болады. Турбо Паскаль ұзындығы ДК-ң негізгі оперативті жадысымен ғана шектелетін программаларды құруға мүмкіндік береді. MS-DOS операциялық жүйесі орындалатын программаға негізгі жадыдан шамамен 580 кбайт қалдырады(Турбо Паскаль жүйесі мен резидентті программаларды есепке алмағанда). Мұндай көлемдегі жады көбінесе жеткілікті, сонда да, қиылыса жабуды қолданатын программалар үшін мұндай шектеме жоқ.

TURBO3 и GRAPH3 модульдері Турбо Паскальдің алдыңғы 3.0 версиясымен үйлесімді болуы үшін енгізілген.

2. Тілдердің препроцессорлық құралдары

Объектіге бағытталған программалау тілдері тұрғысынан объектке келесі анықтаманы беруге болады.

Объект - **бұл жеке локал жадыға ие және төмендегі әрекеттерді орындай алатын программалау тілінің абстракциясы(«бір нәрсе»):**

1. **Қайсыбір әрекеттерді орындайды.**
2. **Локал жадыда қайсыбір мәліметті сақтайды.**
3. **Өзге объекттермен өзара қатынас жасайды.**

Объекттер *хабарламалар* **арқылы өзара қатынас жасайды.**

(MS Windows операциялық жүйелерінде «хабарлама» ұғымы қолданылады. Бірақ бұл ұғымның мағынасы өзгешелеу).

Өзарақатынас жасауға екі объект қатысады: хабарлама *жіберуші* мен *қабылдаушы*. Хабарламада жіберуші қабылдаушыға қандай *әрекет* орындалу керек екеін білдіріп, бұл үшін керекті параметрлерлі жібереді. Қабылдаушы әрекетті орындап, жіберушіге *нәтижені* қайтарады. **Өзара қатынастың басқа түрі қарастырылмаған.**

ОБП-ң мықты тілі – Смолток тілінде (қазір қолданылмай кеткен) хабарламада *селектор* және **параметр** болып ажыратылады.

Селектор **– бұл** *қабылдаушы объекттен* **қандай әрекет (мағынасы бойынша) талап етілетіндігін білдіретін** *хабарламаның бөлігі***.**

Әрбір хабарламаға селектор қатысады.

Хабарлама параметрлері процедураның нақты параметрлері сияқты қажетті бастапқы берілгендерді беріп, әрекетті нақтылай түседі. Хабарламада параметр біреу не бірнеше не тіпті жоқ болуы да мүмкін.

Мысалы, әрекет «объектті экранда қайсыбір позицияда көрсету» дегенді білдіруі мүмкін. Онда хабарламаның селекторы **ShowAtPosition** идентификаторы болып, ал параметрлері – талап етілген экран позициясының X және Y координаталары болып табылуы мүмкін.

Бірқатар ОБП тілдерінде идентификатор ғана емес, әріпті-цифрлық тізбек және қызметші символдар да (мысалы, +, -, *, /, ->, =) селектор ретінде беріле алады. Бұл мысалыға, арифметикалық өрнекті хабарламалардың тізбегі ретінде жазуға мүмкіндік береді. Айталық, барлық сандар объект болып табылса, «+» селекторлы хабарлама «параметр мен өзін қосып нәтижені қайтару» дегенді білдіреді: өрнектің жазылуы былай болады: 2+3, ал нәтиже ретінде 5 қайтарылады. Мұндай жазылу адамға түсінікті. Егер қызметші символдарды селектор ретінде пайдалануға рұқсат етілмесе (Паскалда сияқты), онда конструкцияны келесі түрдегі сияқты жазуға тура келеді:

2.add(3) Алайда программаның оқып, түсінілуі қиындайтыны белгілі.

Объектті өңдеуге қабілетті *хабарламалардың* **жиыны объект** *интерфейсі* **деп аталады.**

Объектке интерфейсінде бар хабарламалар ғана жіберілуге тиіс.

Тәсілдер

Өзіне жіберілген хабарламаларды өңдеу үшін объект *тәсілдер*(кейде ережелер деп аталады) **деп аталатын ерекше процедураларды пайдаланады**.

Объекттің *мінез-құлығы* **деп оның тәсілдерінің бірлестігін айтады.**

Тәсіл хабарлама параметрлерін қабылдап, оларды және объекттің локал жадысындағы мәліметтерді басшылыққа ала отырып, қайсыбір әрекеттерді (есептеулерді) орындап, нәтижені хабарламаны жіберген объектке қайтарады.

Си++, Паскаль сияқты тілдердің жасақшылары Смолток тіліндегідей хабарламаны символдық идентификаторлармен бірге жіберуден бас тартты. Себебі, мұндай механизм өте үлкен және сәйкес тәсілді таңдау үшін кестелерден хабарлама идентификаторын іздеп отыру қажет. Мұның орнына хабарламаны жіберу бұл тілдерде тәсілді процедура ретінде шақырумен алмастырылған. Хабарламаның бастапқы идеясына сәйкестендіру үшін тәсілді шақыру қарапайым процедураны шақырудан өзгешелеу етілді.

Объекттің локал жадысы. Инкапсуляция принципі

Объекттің локал жадысы өз құрылымына ие. Програмалаудың әртүлі тілдерінде объекттің локал жадысының құрылымдық бірлігін белгілеу үшін түрлі термин қабылданған. Мысалы, **өріс** (Турбо Паскальда), объект **жағдайының айнымалысы**(Си++-те), немесе экземплярайнымалы(Смолток-та).

ОБП идеясы нақты объекттің тәсілдері ғана оның локал жадысымен жұмыс істей алады дейді, және локал жадының құрылымы тәсілдің ішінде ғана белгілі деп есептейді. Осы көзқарас тұрғысынан «Бұл тәсіл объекттің мынадай айнымалысының (өрісінің) мәнін қайтару үшін қызмет етеді» деу дұрыс емес. Өйткені сырт көзге объекттің локал жадысында осындай айнымалының бар екендігі белгісіз.

Мысалы, жолаушы лифтінің мінез-құлқын моделдейтін объектті сипаттайтын болсақ, және локал жадының ***passengerCount*** деп аталатын бір ұяшығында кабинадағы жолаушылардың ағымдық саны сақталса, онда «кабинадағы жолаушылар санын қайтаратын» тәсіл құруға болады. Бірақ, ол туралы «***passengerCount*** өрісінің мәнін қайтаратын» тәсіл деп айту дұрыс емес (ол осылай жұмыс істеуі де мүмкін).

Жоғарыда айтылған мәліметтерді қорытындылай келе, объекттерді «қара жәшік» ретінде елестету мүмкін, оның кірісіне қайсыбір әсер(хабарлама) беріледі, ал

шығысынан жауап сигналы(нәтиже) алынады. Қара жәшіктің ішінде дәл не орындалатындығы, оның құрылымы– сырткөзге көрінбейді.

Инкапсуляция принципі объектпен тек бір бүтін ретінде ғана жұмыс істеуді талап етеді. Объектті бір бүтін ретінде құруға, көшіруге, жоюға немесе «қолынан келетін нәрсені өтініш етуге»(хабарлама жіберуге) болады. Бірақ, сырттан тұрып объекттің бөлігіне(өрісіне) қол жеткізуге болмайды.

Паскальда Си++, Смолток және ОБП-ң басқа тілдерінде объект өрісі ретінде басқа бір объектті алуға болады. Бұл жағдайда инкапсуляция принципі бойынша тәсілдің ішінде мұндай объект-өріспен жұмыс тек хабарлама жіберу жолымен жүргізілуі керек. **Объект-өрістердің өрісіне қолжеткізу тек олардың өз тәсілдері арқылы мүмкін болады.**

3. Объектіге бағытталған программалау кластар және әдістер

түсінігі

Егер жеке-жеке әрбір объект үшін интерфейсі, локал жадысының құрылымы, тәсілдері сипатталатын болса, онда программа өте үлкен болып кетеді. Сондықтан ОБП-ң барлық тілдерінде класстар механизмі енгізілген.

Класс **деп бірдей объекттердің, яғни,** *бірдей интерфейске, локал жадының құрылымына***(өрістердің бірдей жиыны)***, **және** *мінез-құлыққа* **(жіберілген хабарламаларды өңдеу үшін бірдей тәсілдер қолданатын) ие объекттердің жиынын атайды.**

Программа жазу кезінде программист алдымен классты сипаттайды, яғни, ішіне енетін барлық объекттердің интерфейсін, ішкі құрылымын және тәсілдерін (мінез-құлқын) сипаттайды. Бұдан соң, нақты объектті сипаттаған уақытта оның берілген классқа тиістілігін білдіреді. Және мұндай объектті берілген класстың **экземпляры** деп айтады. Әрбір объект қайсыбір класстың экземпляры болып табылады.

Объектілі типті жариялау

Паскалда «объект» терминінен шатасуымыз мүмкін, өйткені «объект» термині классты да, осы класстың экземплярын да білдіре береді. Сондықтан, анық болуы үшін бұдан бұлай әңгіме класс туралы болатын болса, «класс» не «объектілі тип» терминдерін қолданамыз, әңгіме экземпляр туралы болатын болса онда «экземпляр (объекттің)» терминін қолданамыз. Барлық объекттер қайсыбір объектілі типтің экземпляры ретінде туындайды. Демек, тип экземплярды жариялаудан (сипаттаудан) бұрын жариялануы (сипатталуы) керек.

Объектілі типті жариялау (сипаттау) 2 бөлімнен тұрады: 1.Класс атауын жариялау. Класс атауы интерфейсті (тәсілдер атауының тізімі) және берілгендер өрісі құрылымын өз ішіне алады. 2.Тәсілдер орындалуларын (реализациясын) жариялау. Класс атауын жариялау **type** секциясында жүргізіледі. Тип атауынан соң **object** кілттік сөзі жазылады, бұдан соң өрістер тізімі жазылады, одан кейін тәсіл атауларының тізімі жазылады(нүктелі үтір арқылы).

Тәсіл **процедура немесе функция** болу мүмкін. Әдетте, Паскалда объект міндетті түрде **Init** атаулы инициализации тәсіліне және **Done** атаулы деинициализации (жойып жіберуге дайындық) тәсіліне ие болу керек.

Объект және тәсілді жариялау мысалы

Сөздік сипаттама. Объект – бұл жерде жүгіріп тұрған қатар(бегущая строка). Экранның белгілі бір жерінде төртбұрыш орналастырылуы мүмкін, оның ішіндегі текст қатарлары горизонталь бойынша жылжып отырады. Бұл әрекетті тоқтатуға және іске түсіруге болады.

Әрекеттер(Действия): инициализация (қатарды және оның экрандағы орнын беру), іске түсіру, тоқтату. Деинициализация әрекеті талап етілмейді, бірақ ол ереже бойынша қатысуы тиіс. Сондықтан оның тәсілін объект интерфейсіне тіркейміз, ал орындалуын бос қалдырамыз.

Локал жадының құрылымы: X,Y координаталары, S қатары, әрекеттің іске түскендігінің белгісі - **running**, жүгірмелі қатардың экранда көрсетілетін ағымдағы бөлігі (текущее смещение от начала строки для показа на экране) - **offset**.

```
type TRunnungString = object
        x,y: integer;
s:string;       running:boolean;
         offset:integer;

        procedure Init(xx,yy:integer;ss:string);
procedure Run;      procedure Stop;
function IsRunning:boolean;          procedure
Done;
  end;
```

Тәсілдің формальді параметрлерінің атауы объект өрістерінің атауларымен бірдей болмауы қажет. (**Init** тәсіліне қараңыз).

Тәсіл реализациясының жариялануы және тәсілді шақыру

Класс атауын жариялаудан бөлек, одан әрі программаның процедураларды сипаттау бөлімінде класс атауында келтірілген барлық тәсілдердің орындалуы (реализациясы) жарияланады. Тәсіл орыдалуы процедура не функция сипаттамасына ұқсас сипатталады. Бірақ:

- тәсіл атауының алдынан міндетті түрде классының атауы нүкте арқылы жазылады;
- тәсіл ішінде класс атауында келтірілген объекттің барлық өрістері **көрінетін** (видимыми) деп есептеледі.

Мәселен, **Init** тәсілінің орындалуы (реализациясы) мынадай болады:

```
procedure TRunningString.Init(xx,yy: integer; ss:string); begin
 x:=xx; y:=yy; s:=ss; end;
```

Паскаль тілі синтаксисінің ережесі тәсілдің реализациясын (орындалуын) сипаттау кезінде формальді параметрлер тізімін жазбауға рұқсат береді, себебі олар класс атауында келтірілген. Бірақ бұлай **істеу керек емес**! Программа оқылғанда түсінікті болуы үшін тәсіл реализациясында да формальді параметрлер тізімін келтірген дұрыс. Жазылған **Init** тәсілі нені білдіреді? Бұл мынаны білдіреді: **TMyObject** классының әрбір экземпляры инициализация бойынша әрекет ете алады, және

формальді параметр ретінде берілген 2 бүтін сан және қатар инициализация жасалуын қалап отырған экземплярдың өрістеріне орналастырылады.

Алайда, біздің класстың әлі бірде-бір экземпляры жоқ, сондықтан біз сипатталған тәсілдерді шақыра алмаймыз. Программаға жүгірмелі қатардың экземплярын құру керектігін білдіру үшін айнымалыларды сипаттау бөлімінде былай жазамыз: **var** RunningString:TRunningString;

Бұл экземплярды (айнымалыны) пайдаланып программа денесінде мысалы былай жазуға болады:

```
RunningString.Init(10,10,'Hello, world');
RunningString.Run;
.....
RunningString.Done;
```

Яғни, тәсілді шақыру үшін объект экземплярының атауынан кейін нүкте қойылып, тәсіл атауы нақты (фактический) параметрлер тізімімен бірге жазылады.

Бақылау сұрақтары

1. Тілдердің препроцессорлық құралдары?
2. макроанықтамалар және макрокеңейтулер?
3. Стандартты кітапханаларды қолдану?
4. Объектіге бағытталған программалау кластар және әдістер түсінігі және сипаттамасы?

Ұсынылатын әдебиеттер

1. Бадд Т. Объектно-ориентированное программирование в действии. Питер. 1997.
2. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на С++, 2-е изд./Пер. сс англ. –М.: «Издательство Бином», Спб.: «Невский диалект», 2001.
3. Бьярн Страуструп. Язык программирование С++. Киев: Диасофт, 1993. 1,2 часть.

Гамма Э. Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного проектирование. Паттерны проектирования. – СПб: Питер, 2001. 5. Ишкова Э.А. С++ начала программирования. – М.:Бином, 2001