Лабораториялық жұмыс № 1

Тақырыбы: Бірінші бағдарламамен танысу, оның құрылым, және сипаттау бөлімі және айнымалылар типі

Мақсаты: бағдарламалық құрылыммен танысу, ондағы бөлімдерге жеке-жеке тоқталу, жұмыс жасау қағидасына ерекше тоқталу, айнымалылар типімен тансыу оларды меңгеру

Теориялық бөлім

Турбо Паскаль тілінде бағдарлама құруда бірнеше кезеңдерді немесе құрылымдарды атауға болады. Ол құрылымға бағдарламаның атауы, бағдарламадағы айнымалына сипаттау бөлімі, бағдарламаның басы, бағдарлама денесі, бағдарламаның соңы жатады.Осылардың барлығы бағдарлама құрылымдарын береді.

Program pr1; /\*бағдарламаның атауы\*/

Var /\*айнымалыны сипаттау бөлімі\*/

Begin /\*бағдарлама басы\*/

End. /\*бағдарлама соңы\*/

Енді бірінші бағдарлама мысалын терейік.

program My\_First\_Program;

const

Text=’Мен Турбо Паскаль тілінде бағдарлама құрамын’;

Begin

WriteLn(Text);

End.

Бағдарламаны ашу үшін жолдарға көңіл бөлі қажет. Себебі осы жолдардағы мәтіндер жеке-жеке фрагментті береді.

Ескерту: Турбо Паскаль тіліндегі бағдарламада мәтін бас әріппен немес кіші әріппен теріле берсе болады. Олардың бір бірінен ешқандай айырмашылығы жоқ.

Турбо Паскаль тілінде бірнеше алдын ала белгіленіп айнымалы ретінде қолдануға болмайтын сөздер бар. Ол сөздер осы бағдарламада мысалы program, const, begin, end сияқты сөздері. Демек осы сөздерді айнымалы ретінде қолдануға болмайды деген сөз.

Екінші жолдағы Const сөзі осы оператордан кейін бірнеше немесе бір тұрақтының болтындығын көрсетеді. Тұрақты (Constants- тұрақты) дегеніміз бағдарламалау кезіндегі өзгеріске енбейтін сан.

Басқа бағдарламалау тілдеріндегідей Турбо Паскаль тілінде де бірнеше тұрақтыларды инженерлік есептеулер және ғылыми есептеулерде қолдану үшін алдын ала бір мәнге теңестіріп, тұрақты қылып алғанымыз жөн болады. Мысалы үшін π = 3.14159265 тұрақтысы бағдарлама басмында меншіктеп қойсақ бағдарлама аяғына дейін жеткілікті болады. Тұрақтының қандай болатындығы бағдарламаны құрушыға байланысты нәрсе болады. Ол мәтіндік ақпарат немесе сандық ақпараттан тұруы мүмкін. Біздің мысалымызда ол мәтіндік ақпараттан тұрады. Text деген қызметші сөздің арқасында сол сөзге меншіктелген тұрақты экранның бетіне шығуы керек. Турбо Паскаль тілінде тұрақты бірнеше түрлі нақты сан, белгілер, жолдық белгі ретінде, массив ретінде тағы басқалар ретінде ала беруге болады. Осы бөлімді сипаттау бөлімі деп атайды.

Келесі жол ол begin сөзімен байланысты жол. Осы жолдың немесе қызметші сөздің көмегімен біз бағдарламаның нағыз басын басауымызға болады. Бұл қызметші сөз барлық кез-келген бағдарламаның басын көрсеитеді. Осы қызметші сөзден кейін барып қана бағдарламаның операторлар бөлімін яғни нағыз жұмыс жасайтын бөлімін жазамыз. Біздің мысалда ол WriteLn(Text); операторы болып тұр. Осы оператордың көмегімен экранның бетіне бағдарламаның мәтінін және нәтижені шығаруға болады. Бағдарламаны іске қосқаннан кейін экранның бетіне нәтиже шығады. Осы бөлімдегі кез-келген операторлардың жиынтығын процедураға меншіктеп қоюға болады.

Процедура дегеніміз – ол бірнеше атаулары бойынша талап жасауға болатын операторлардың тізбектелген жиынтығы. Бірнеше операторларды қайцталап жаза бермеу үшін немесе бірнеше операторлардың жиынтығын орындау үшін осы процедуралар қолданылады.

WriteLn Турбо Паскаль тіліндегі стандартты процедураға жатады. Стандартты процедура алдын ала сипаттауды талап етпейді. Ол кез-келген бағдарлама үшін тең мүмкіндікті болады. Осы процедураның көмегімен бағдарламадағы кез-келген нәтижені экранның бетіне шығаруға болады.

WriteLn (‘Мен Турбо Паскаль тілінде бағдарлама құрамын’) нің орнына кез-келген операторды жазуға болады. Мысалы үшін біздің мысалда экранның бетіне мәтін шығарайын деп жатқандықтан WriteLn (Text) деп те алуға болады. Себебі Text алдын ала тұрақтылар бөлімінде сипатталып кеткен болатын.

Бағдарламаның соңында end қызметші алдын ала таңдалған яғни Турбо Паскаль тілі таңдағын сөзге жатады. Бұл сөз Турбо Паскаль тіліндегі бағдарламаның аяқталғанын көрсетеді. Осы сөз барлық бағдарламаны жабады. Артынан нүкте қойылады.

Ескерту: Турбо Паскаль тіліндегі бағдарламада барлық фргаменттен кейін демек жолдан кейін нүктелі үтір қойылып отырады. Бұны бағдарламалық тілдің синатксисі деп атайды. Егер бір нүктелі үтір жазылмай қалса, бұл синтаксистік қателікке жатады. Бағдарлама компиляция жасар кезінде бұны қате ретінде экранға шығарады.

Осы бағдарламадағы program, const, begin, end қызметші сөздері алдын ала бағдарламалық тілмен белгіленіп алған қызметші сөздерге жатады. Ал Text, My\_First\_program сөздері болса бұлар идентификаторлар болып табылады. Яғни идентификатор дегеніміз бағдарламалаушы қолданған сөздер, ярине ол сөздер ағылшын тілінде терілген болуы шарт.

Турбо Паскаль тілінде апострофтың ішіндегі ғана сөз орын немесе қазақ тілінде ғана жазылады. Қалған жағдайдың барлығында бағдарлама ағылшынша теріледі.

Бағдарламалаушы идентификаторды кез-келген ретпен қолдана беруге болады. Олар:

* идентификатор латын әріпінен, символдан тұруы мүмкін;
* идентификатор саннан басталуы мүмкін емес;
* идентификатор алдын ала бағдарлама белгілеп таңдап алған сөзбен сәйкес бомауы керек;
* идентификатордың ұзындығы кез-келген жағдайда болуы мүмкін, бірақ бағдарлама бастапқы 63 символды ғана оқиды.

Енді жазған бағдарламаны іске қосу қажет. Ол үшін Ctrl+F9 батырмасын басу керек. Осыдан кейін Турбо Паскаль тілінің экраны тазарып, экранның бетіне жұмыс жасаған бағдарлама нәтижесі шығады. Осы терезе бағдарлама терезесі деп аталады. Егер сіз ашылған бағдарлама бетін көріп үлгере алмай қалсмаңыз онда Alt+F5 батырмасын басуыңыз қажет болады. Нәтижені көріп болғаннан кейін бағдарлама бетіне қайтадан оралу үшін Esc батырманы басу арқылы бағдарлама бетіне келеміз. Жұмысты жалғастыру үшін F10 батырмасын таңдау арқылы жаңа бетті немес осы бетке жаңа бағдарлама терін жұмысты жалғастырамыз.

Жоғарыда айтақнда кез-келген бағдарламада мәліметтердің өздеріне тән типтері болады. Осы титерді бағдарламалау тіліндегі мәліметтер типі деп атайды.

Идентификаторды сипаттау – ол бағдарлама объектісімен байланысқан типтің көрсеткішін көрсету болып табылады. Жалпы тип түсіні Турбо Паскаль тіліндегі ең негізгі ұғымдардың біріне жатады.

Турбо Паскаль тілінде мәліметтер типінің тірнеше түрін қарастыруға болады. Олардың ішінде негізгісі:

* INTEGER – бүтін типі мәнол 2 байт орын алады, санның ұзындық мүмкіндігі ол –32768-ден +32767-ге дейінгі сандарды қабылдай алады; мәліметтер нақтылы түрде көрсетіледі;
* REAL – бүтін емес сандар жиыны, 6 байт орын алады, мүмкінігі 2,9E-39 дан 1,7Е+38-ге дейінгі сандар жиынтығы жатады;
* CHAR – символ, 1 байт орын алады;
* STRING – жол символы, MAX+1 байт оры алады, мұндағы MAX жолдағы символдардың максимальды ұзындығы;
* BOOLEAN – логикалық тип, 1 байт орын алады оның екі мән бар Олар False және True;

Тұрақтылар типіның мысалы ретінде мыналарды көрсетуге болады.

Const

C1= 17; /\*Integer\*/

C2= 3.14; /\*real\*/

C3=’A’; /\*char\*/

C4=’3.14’; /\*string\*/

C5=False; /\*boolen\*/

Турбо Паскаль тілінде стандартты математикалық функциялар қолданылады:

|  |  |  |  |
| --- | --- | --- | --- |
| Талап беру | Параметр типі | Тип нәтижесі | Сиаптталуы |
| Abs(x) | Real, integer | Аргумент типі | Аргумент модулі |
| ArcTan(x) | Real | Real | Арктангенс |
| Cos(x) | Real | Real | Косинус |
| Exp(x) | Real | Real | Экспонента |
| Frac(x) | Real | Real | Санның бөлшектік түрі |
| Int(x) | Real | Real | Санның түтін түрі |
| Ln(x) | Real | Real | Натурал логарифм |
| Pi | Real | Real | π = 3.14159265 |
| Random | Real | Real | 0…1 арасындағы кез-келген сан |
| Random(x) | Integer | Integer | 0…(x-1) арасындағы кез-келген сан |
| Randomize | - | - | Кез-келген санды іске қосу |
| Sin(x) | Real | Real | Синус |
| Sqr(x) | Real | Real | Аргумент квадраты |
| Sqrt(x) | Real | Real | Квадраттық түбір |

Турбо Паскаль тіліндегі бағдарламамен алдын ала берілген таңдалған сөздер тізімі. Демек бұл сөздерді айнымалы ретінде немесе тип ретінде қолдануға болмайды. Олар:

|  |  |  |  |
| --- | --- | --- | --- |
| And | End | Nil | Shr |
| Asm | File | Not | String |
| Array | For | Object | Then |
| Begin | Function | Of | To |
| Case | Goto | Or | Type |
| Const | If | Packed | Unit |
| Constructor | Implementation | Procedure | Until |
| Destructor | In | Program | Uses |
| Div | Inline | Record | Var |
| Do | Interface | Repeat | While |
| Downto | Label | Set | With |
| else | mod | shl | xor |

Бақылау сұрақтары:

1. Бағдарлама құрылы деген не?
2. Оның қандай мәні бар, не үшін қажет?
3. Бағдарлама беті деген не?
4. Алдын ала бағдарлама таңдаған қызметші сөз деген не? Басқа оператордан қандай айырмашылығы бар?
5. Идентификатор деген не, ол не үшін қолданылады?
6. Процедура деген не?

*Негізгі әдебиеттер:*1. Йенсен К, Вирт Н. Паскаль. Руководство для пользователя и описаниеязыка. – М.: Финансы и статистика, 19822. Перминов О.Н. Программирование на языке Паскаль. – М.:Радио и связь, 1988

3. Фаронов В.В. Турбо Паскаль 7.0. Начальный курс. Учебное пособие. – «Нолидж», 1988.

4. Горбунов А.В., Антонова Н.А. Технология программирования на языке Паскаль с решением типовых задач.: Учебное пособие. – Караганда, 2003.

Тапсырмалар:

Турбо Паскаль тілінде төмендегі есептерге бағдарлама құру керек:

1. Жоғарыда айтылған бағдарламаға алгоритм құрастыру керек.
2. Герон форумалсына бағдарлама құрастыру керек.
3. Үш бұрыштың ауданын есептейтін формулаға бағдарлама құру.
4. Прифагор теоремасына бағдарлама құру.

Лабораториялық жұмыс №2

Тақырыбы**: Өрнектер және операциялар**

Мақсаты**: Турбо Паскаль тіліндегі өрнектермен танысу, олардың жұмыс жасауын меңгеру қолданылуын бақылау**

Теориялық бөлім

**Паскаль тілінде негізінен үш түрлі өрнектер қолданылады. Олар арифметикалық, логикалық және символдық өрнектер. Арифметикалық өрнек математикада кездесетін өрнек түсінігімен тығыз байланысты, бірақ кейбір айырмашылықтары бар.**

**Арифметикалық өрнектердің мәндері сан, логикалық өрнектердің мәндері – TRUE немесе FALSE символдық өрнектердің мәндері, символдар болады.**

**Турбо Паскаль тілінде келесі операциялар қарастырылады. Олардың түрлері:**

**Унарлық not, @;**

**Мультипликативтік \*, / , div, mod, and, shl , shr;**

**Аддитивтілік +, - , or, xor;**

**Қатынастар =, <>, <, >, <=, >=, in.**

|  |  |  |  |
| --- | --- | --- | --- |
| Операция | Әрекет | Операндтар типі | Тип нәтижесі |
| **Not** | **Теріс** | **Логикалық** | **Логикалық** |
| **Not** | **Ол да солай** | **Кез-келген бүтін** | **Операнд типі** |
| **@** | **Адрес** | **Кез келген** | **Көрсеткіш** |
| **\*** | **Көбейту** | **Кез-келген бүтін** | **Ең кіші бүтін** |
| **\*** | **Ол да солай** | **Кез-келген нақтылық** | **Exended** |
| **\*** | **Жинақтардың қиылысуы** | **Жинақтық** | **Жинақтық** |
| **/** | **Бөлу** | **Кез-келген нақтылық** | **Exended** |
| **Div** | **Бүтін бөлік** | **Кез-келген бүтін** | **Ең кіші бүтін** |
| **Mod** | **Бөлуден қалған қалдық** | **Ол да солай** | **Ол да солай** |
| **And** | **Логикалық және** | **Логикалық** | **Логикалық** |
| **Shl** | **Ол да солай** | **Кез-келген бүтін** | **Ең кіші бүтін** |
| **Shr** | **Сол жаққа жылжу** | **Ол да солай** | **Ол да солай** |
| **+** | **Оң жақ жылжу** | **Ол да солай** | **Ол да солай** |
| **+** | **Қосу** | **Ол да солай** | **Exended** |
| **+** | **Ол да солай** | **Кез-келген нақтылық** | **Жинақтық** |
| **+** | **Жиақтардың бірігуі** | **Жинақтық** | **Жолдық** |
| **-** | **Жолдардың бірігуі** | **Жолдық** | **Ең кіші бүтін** |
| **-** | **Алу** | **Кез-келген бүтін** | **Exended** |
| **-** | **Ол да солай** | **Кез-келген бүтін** | **Логикалық** |
| **Or** | **Логикалық немесе** | **Логикалық** | **Ең кіші бүтін** |
| **Or** | **Ол да солай** | **Кез-келген бүтін** | **Логикалық** |
| **=** | **Тең** | **Кез-келген жолдық және қарапайым** | **Логикалық** |
| **<>** | **Тең емес** | **Ол да солай** | **Ол да солай** |
| **<** | **Кіші** | **Логикалық** | **Ол да солай** |
| **<=** | **Кіші немесе тең** | **Ол да солай** | **Ол да солай** |
| **>** | **Үлкен** | **Ол да солай** | **Ол да солай** |
| **>=** | **Үлкен немесе тең** | **Ол да солай** | **Ол да солай** |

1. Арифметикалық өрнектер.

**Арифметикалық өрнектер - \*,/,+,-,MOD, DIV таңбалары және дөңгелек жақша арқылы белгілі бір ережеге сүйеніп құрастырылған айнымалылардан, константалардан және функциялардан тұрады. Арифметикалық өрнектерде айнымалыны, константаны, функцияны жеке-жеке алғанда қарапайым арифметикалық өрнек деп қарастыруға болады. Арифметикалық өрнектерді қолданғанда төменгі шарттарды естен шығармаған жөн;**

1. **Өрнек қарапайым өрнектерден құрастырылған болса, онда қарапайым өрнектердің арасын арифметикалық таңба қойылуы тиіс.**

**Мысалы, A-ны B-ге көбейтетін болсақ, онда A\*B деп жазу керек.**

1. **Екі жай арифметикалық өрнектің ортасында тек бір ғана таңба болуы тиіс. Мысалы A\*-B деп жазу дұрыс емес. A санын –B-ге көбейту үшін A\*(-B) деп жазу керек.**
2. **Арифметикалық өрнектердің есептеулері белгілі бір сатылық заңдылыққа бағынады. Бірінші кезекте өрнектің көбейту, бөлу, бөліндігнің қалдығын табу, бөліндінің бүтін бөлігі (\*,/,MOD, DIV)орындалады да, ал екінші кезекте қосу және азайту (+,-) амалдары орындалады.**
3. **Арифметикалық өрнек былайша есептеледі: өрнектегі бірінші екі таңба салыстырылады, егер бірінші таңбаның кезегі екінші таңбаның кезегінен үлкен болса, онда бірінші таңба орындалады, ал егер екінші таңбаның кезегі үлкен болса, онда үшінші таңбамен салыстырылады, тағы сол сияқты бұл процесс арифметикалық өрнектің таңбалары аяқталғанша солдан оңға қарай қайталана береді.**

**Мысалы: D+C\*D деген өрнекті қарастырайық.**

**Бұл өрнек алдымен C\*D орындалады да, одан шыққан мән D-ға қосылады.**

1. **Таңбалардың орындалу ретін дөңгелек жақша арқылы ғана өзгертуге болады.**

**Мысалы: A\*B/(C+D) өрнегінде алдымен бірінші кезекте A\*B екінші C+D соңында бөлу (\) амалы орындалады.**

**Бөлу амалы қолданылғанда бөліндінің мәні барлық жағдайда да нақты болады. Мысалы: 25/5=5,0 (5.0-нақты сан)**

1. **Паскаль тілінде дәрежеге көтеру амалы жоқ, сондықтан AB өрнегін былайша жазуға болады:**

**AB=EXP(B\*LN(A)), A>0;**

1. **DIV таңбасы тек бүтін сандар мен бүтін айнымалыларға ғана қолданылдады. Шыққан қорытынды әрқашанда бүтін сан болады.**
2. **Арифметикалық өрнектің ішінде ең болмағанда бір нақты сан болатын болса, онда өрнектің мәні нақты болады.**
3. Логикалық өрнектер.

**Логикалық өрнектер NOT (кері), AND (және, немесе логикалық көбейту), OR (немесе, логикалық қосу), логикалық қатынастар (<,<=,(<),<>,(=),>=(>),>,=) арқылы құрылады. Логикалық өрнектер екі логикалық таңба, тек олардың біреуі NOT болса ғана қатар тұра алады. Лоткалық өрнектер есептелгенде, амалдар белгілі бір кезекпен (арифметикалық өрнек сияқты) солдан оңға қарай орындалады.**

**Ол кезек мына ретке негізделген:**

1. **NOT**
2. **And**
3. **OR**
4. **Логикалық қатынастар**

**Мысалы:**

**NOT A=B орындалуы – (NOT A)=B**

**D>B AND С орындалуы – D>(B AND C)**

**C OR B AND A орындалуы – C OR (B AND A)**

Not **- логикалық жоқ**

And **- логикалық және**

Or **- логикалық немесе**

Xor **- логикалық немесе керісінше**

Логикалық өрнектерде жақшаларды пайдалану арқылы олардың орындалу кезеңін өзгертуге болады.

**Логикалық өрнектің мәтін TRUE немесе FALSE болады және FALSE TRUE ден кіші болып есептелінеді. Логикалық өрнектерді құрастырғанда мына жағдайды ескерген жөн:**

**A=B логикалық қатынасы дұрыс болуы үшін A мен B бүтін айнымалы болуы тиіс. Егер A Мен B нақты болатын болса, онда бұл сандар сандардың жазылуына байланысты немесе дөңгелектеу кезеңіндегі өзгерістерге байланысты теориялық тең сандар іс жүзінде бір-бірінен айырмашылығы болуы мүмкін болғандықтан, қорытынды дұрыс болмауы мүмкін.**

**Логикалық өрнектерде жоғары айтылған таңбалармен қоса +-,-,\*,MOD, DIV амалдары да пайдалануы мүмкін. Бұл жағдайда амалдардың орындалу кезегі мына ретке негізделген:**

1. **NOT**
2. **\*,\,DIV, MOD, AND**
3. **+,-,OR**
4. **Логикалық қатынастар**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| INTEGER типті мәліметтерге логикалық опеарцияларды қолдану кестесі (разряды бойынша) | | | | | |
| Операнд 1 | Операнд 2 | not | and | or | Xor |
| **1** | **-** | **0** | **-** | **-** | **-** |
| **0** | **-** | **1** | **-** | **-** | **-** |
| **0** | **0** | **-** | **0** | **0** | **0** |
| **0** | **1** | **-** | **0** | **1** | **1** |
| **1** | **0** | **-** | **0** | **1** | **1** |
| **1** | **1** | **-** | **0** | **1** | **0** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| BOOLEN типті мәліметтерге логикалық опеарцияларды қолдану кестесі (разряды бойынша) | | | | | |
| Операнд 1 | Операнд 2 | not | and | or | xor |
| **True** | **-** | **False** | **-** | **-** | **-** |
| **False** | **-** | **True** | **-** | **-** | **-** |
| **False** | **False** | **-** | **False** | **False** | **False** |
| **False** | **True** | **-** | **False** | **True** | **True** |
| **True** | **False** | **-** | **False** | **True** | **True** |
| **True** | **True** | **-** | **True** | **True** | **True** |

1. Символдық өрнектер.

**Символдық өрнектің мәні – символ болып табылады. Символдық айнымалылардың түрлері ЭЕМ-нің түрімен тығыз байланысты болғандықтан символдық өрнектердің де мәні “TVнің мүмкіндігіне тікелей байланысты. Бірақ Паскальда қолданылатын символдарға төмендегідей талаптар қойылады:**

1. **Әр символдың өзінің реттік нөмірі болады.**
2. **0,1,2,3,4,5,6,7,8,9 цифрларының реттік нөмірі өсу бағытында реттеледі де, бірінің соңынан бірі келіп отырады.**
3. **Әріптердің де реттік нөмірлері өсу бағытына қарай реттелген, бірақ бірінің артынан бірі келіп отырыуы шартты емес.**

**Программаның ЭЕМ-нің түріне тәуелсіз болуы үшін, тек стандартты символдар жиынын пайдаланған дұрыс. Қазіргі кезде көп тараған стандартты символдар жиынына ISO және ASCII системаларын жатқызуға болады.**

**Символдық өрнектер тек логикалық қатынастар арқылы және символдарды біріне бірін жалғау арқылы ғана құралады. Символдық өрнектерді бір-біріне жалғау үшін “+” амалын пайдаланады.**

**Қорытынды өрнектің ұзындығы 255 символдан аспауы керек.**

**Символдық өрнекте логикалық қатынастар (=,<>,<,>,>=,<=) таңбалары арқылы құрылады. Символдық өрнекте логикалық қатынастардың таңбаларының кезегі жалғау (+) амалының кезегінен кіші болады. Сондықтан символдық өрнекте жалғау және салыстыру таңбалары араласып келсе, онда алдымен барлық жалғау таңбалары орындалады да содан кейін солдан оңға қарай салыстыру амалдары орындалады. Логикалық символды өрнектің мәні true және false болады.**

**Егер екі символды салыстыратын болса, онда оңнан солға қарай біртіндеп кезегімен әр цифр, әр әріп салыстырып отырылады. Егер символдағы әріптер мен сандар бірдей болса, онда қайсысы ұзын, сол үлкен болады.**

**Бас әріп кіші әріптен кіші болып есептелінеді.**

*Негізгі әдебиеттер:*

1. Йенсен К, Вирт Н. Паскаль. Руководство для пользователя и описание языка. – М.: Финансы и статистика, 1982

2. Перминов О.Н. Программирование на языке Паскаль. – М.:Радио и связь, 1988

3. Фаронов В.В. Турбо Паскаль 7.0. Начальный курс. Учебное пособие. – «Нолидж», 1988.

4. Горбунов А.В., Антонова Н.А. Технология программирования на языке Паскаль с решением типовых задач.: Учебное пособие. – Караганда, 2003.

Бақылау сұрақтар:

1. **Өрнек дегеніміз не және ол қанша түрге бөқлінеді?**
2. **Арифметикалық өрнек қалай есептелінеді?**
3. **Қандай өрнекті логикалық өрнек дейміз және ол қалай есептелінеді?**
4. **Символдық өрнектер қалайша құрылады?**

Тапсырмалар:

1. **Мына формуланы Паскаль тілінде жазу керек:**
2. **Мына логикалық өрнектердің мәні неге тең болады:**
3. **Sqr (x), sqr(y)<4, егер x=0.3, y=-1.6**
4. **K mod 7 = k div 5-1, егер k=15;**
5. **T and (p mod 3=0), егер t=true, p=101010;**
6. **(x\*y<>0) or (y>x), егер x=2, y=1;**
7. **a or (not b), егер a=false, b=true;**

**Лабораториялық жұмыс**№3

**Тақырыбы**:Мәліметтер типі

**Мақсаты**:Турбо Паскаль тілінде мәліметтер типімен танысу, олардың қолданылу аймағын қарастыру

**Теориялық бөлім**

Кез-келген мәліметтер, тұрақтылар, айнамалылар, функция мәндері, өрнектер Турбо Паскаль тілінде өз типімен сипатталады. Тип – жіберілген мәндердің жиынтығын береді. Сонымен қатар тип компьютер жадысыныңішіндегі мәліметтің форматында көрсетеді. Турбо Паскаль тіліндегі мәліметтер типінің құрылымын төмендегідей беруге болады:

Қарапайым типтер

Қарапайым типтерге реттік және нақтылық типтер жатады. Реттік типтер мүмкін болатын мәндердің соңғы санын көрсетеді. Осы сандар анықталған, реттелген ретпен жүреді.

Нақтылық типтер, санның ең соңғы мәні болып табылады.

Реттік типтер.

Реттік типке бүтін, логикалық, символдық, тізбектелген және типтің диапазоны жатады. Осылардың кезкелгеніне ORD(X) функциясын қолдануға болады. Ол X санының реттік номерін қайтарады. Бүтін типтер үшін ORD(X) кезкелген Xтің мәнін бүтін тип ретінде қайтарады. ORD(X) логикалық, символддық, тізбектелген типтер үшін мен 0 мен 1 ге дейінгі (логикалық мән), 0 ден 255 ке дейінгі (символдық), 0 ден 65535 дейінгі (тізбектелген) мәндерді қайтарады. Типтің диапазоны барлық тізбектелген типтің базалық қасиетін қайтарады.

Реттелген типке сонымен қатар төмендегі функцияларды қолдануға болады:

PRED(X) – тізбектелген типтің алдындағы мәнін қайтарады. Ол ORD(X)-1 мәніне сәйкес келеді. Мысалы: ORD(PRED(X))=ORD(X)-1;

SUCC(X) – тізбектелген типтің келесі мәнін қайтарады. Ол ORD(X)+1 мәніне сәйкес келеді. Мысалы: ORD(SUCC(X))=ORD(X)+1;

Мысалы бағдарламада айнымалы былай берілген болса:

Var

C:char;

Begin

C:=’5’;

End;

Онда PRED(C) функциясы ‘4’ мәнін, ал SUCC(C) ‘6’ мәнін береді.

**Бүтін типтер.**

Мүмкін болатын бүтін типтер бір, екі, немесе төрт байтты алатын ішкі ұсыныстарға байланысты болады. Төмендегі кестеден бүтін типтердің атаулары, ішкі ұсыныстағы олардың байттық ұзындығы, әрі мүмкін болатын мәндер диапазоны берілген.

|  |  |  |
| --- | --- | --- |
| Бүтін типтер | | |
| Атаулары | Байттың ұзындығы | Мәндер диапазоны |
| Byte | 1 | 0…255 |
| ShortInt | 1 | -1280..+127 |
| Word | 2 | 0…65535 |
| Integer | 2 | -32768…+32767 |
| LongInt | 4 | -2147483648…+2147483647 |

Бүтін типтерге қолданылатын процедуралар мен функциялар төмендегі кестеде көрсетілген. b,s,w,i,l мәндері Byte, Shortint, Word, Integer, Longint бүтін мәндеріне сәйкес келеді. X- кез-келген типтегі өрнек. Vb, vs,vw,vi, vl,vx –типтерге сәйкес келетін айнымалылар. Квадраттық жақшаларда міндетті емес параметрлер көрсетілген.

|  |  |  |
| --- | --- | --- |
| Бүтін типтерге қолданылатын стандартты процедуралар және функциялар | | |
| Талап жасау | Типтің нәтижесі | Әрекет |
| Abs(x) | X | X модульін қайтарады |
| Chr(b) | Char | Коды бойынша символды қайтарады |
| Dec(vx,[,i]) | - | Vx мәнін i кішірейтеді, ал егер i жоқ болса , онда 1-ге өзгертеді |
| Inc(vx[,i]) | - | Vx мәнін i үлкейтеді, ал егер i жоқ болса , онда 1-ге өзгертеді |
| Hi(i) | Byte | Аргуметнттің үлкен байтын қайтарады |
| Hi(w) | Ол да солай | Аргументтің үлкен байтын қайтарады |
| Lo(i) | ‘’ | Аргументтің кіші байтын қайтарады |
| Lo(w) | ‘’ | Аргументтің кіші байтын қайтарады |
| Odd(l) | Boolean | Егер аргумент тақ болса, онда True мәнін қайтарады |
| Random(w) | Параметрдікі сияқты | 0…(w-1) диапазонындағы бірқалыпты кез-келген санды қайтарады |
| Sqr(x) | X | Аргументтің квадратын қайтарады |
| Swap(i) | Integer | Сөзде байттардың орнын ауыстырады |
| Swap(w) | word |  |

Әрекет кезінде бүтін сандар типінің нәтижесі типтің операндысына сәйкес келеді, ал егер операндылар әртүрлі бүтін типтерге сәкес келсе, онда бір үлкен типті операндының үлкендігіне байланысты соның операндысының типі алынады.

Мысалы:

Var

A:integer;

X,y:real;

Begin

A:=32767; {Integer типті максимальды мүмкін мән}

X:=a+2; {Осы өрнекті шығару кезіндегі толықтырылу}

Y:=LongInt(a)+2; {Өте үлкен типтегі анйлымалыны келтіргеннен кейін толтырылу жоқ}

Writeln(x:10:0; y:10:0);

End.

Нәтижесінде алатынымыз:

-32767 32769

**Логикалық тип.**

Логикалық типтің мәні False (жалған) және True (ақиқат) сәйкес келеді.

Ол үшін келесі ережелер сай келеді:

Ord(false)=0;

Ord(true)=1;

False<true;

Succ(false)=true;

Pred(true)=false;

Лоикалық тип реттік типке жататын болғандықтан оны санама операторына жатқызуға болады. Оны төмендегі мысалдан көруге болады:

Var

L:boolean;

Begin

For l:=false to true do …

**Символдық тип.**

Символдық типтің мәні болып дербес компьютердегі барлық символдардың жиынтығы жатады. Әр символға бүтін сан 0…255 ке дейінгі сандар аралағындағы сандар жата алады. Оларды кодтау үшін ASCII (American Standard Code for Information Interchange – американский стандартный код для обмена информацией) жүйесі қолданылады. Ол 7 биттық код. Демек, оның көмегімен тек қана 128 ғана символды ғана 0 ден 127 ге дейінгі символды ғана кодтауға болады.

Осы кесте төменде берілген:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ASCII стандартына сәйкес символдардың кодталуы | | | | | | | | | |
| 0 | NUL | 31 | US | 62 | > | 93 | } | 124 | | |
| 1 | SOH | 32 | BL | 63 | ? | 94 | ^ | 125 | } |
| 2 | STX | 33 | ! | 64 | @ | 95 | \_ | 126 | ~ |
| 3 | ETX | 34 | “ | 65 | A | 96 | ‘ | 127 | || |
| 4 | EOT | 35 | # | 66 | B | 97 | a |  |  |
| 5 | ENQ | 36 | $ | 67 | C | 98 | b |  |  |
| 6 | ACK | 37 | % | 68 | D | 99 | c |  |  |
| 7 | BEL | 38 | & | 69 | E | 100 | d |  |  |
| 8 | BS | 39 | ‘ | 70 | F | 101 | e |  |  |
| 9 | HT | 40 | ( | 71 | G | 102 | f |  |  |
| 10 | LF | 41 | ) | 72 | H | 103 | g |  |  |
| 11 | VT | 42 | \* | 73 | I | 104 | h |  |  |
| 12 | FF | 43 | + | 74 | J | 105 | i |  |  |
| 13 | CR | 44 | , | 75 | K | 106 | j |  |  |
| 14 | SO | 45 | - | 76 | L | 107 | k |  |  |
| 15 | SI | 46 | . | 77 | M | 108 | l |  |  |
| 16 | DEL | 47 | / | 78 | N | 109 | m |  |  |
| 17 | DC1 | 48 | 0 | 79 | O | 110 | n |  |  |
| 18 | DC2 | 49 | 1 | 80 | P | 111 | o |  |  |
| 19 | Dc3 | 50 | 2 | 81 | Q | 112 | p |  |  |
| 20 | Dc4 | 51 | 3 | 82 | R | 113 | q |  |  |
| 21 | NAK | 52 | 4 | 83 | S | 114 | r |  |  |
| 22 | SYN | 53 | 5 | 84 | R | 115 | s |  |  |
| 23 | ETB | 54 | 6 | 85 | U | 116 | r |  |  |
| 24 | CAN | 55 | 7 | 86 | V | 117 | u |  |  |
| 25 | EM | 56 | 8 | 87 | W | 118 | v |  |  |
| 26 | SUB | 57 | 9 | 88 | X | 119 | w |  |  |
| 27 | ESC | 58 | : | 89 | Y | 120 | x |  |  |
| 28 | FS | 59 | ; | 90 | Z | 121 | y |  |  |
| 29 | GS | 60 | < | 91 | { | 122 | z |  |  |
| 30 | RS | 61 | = | 92 | \ | 123 | { |  |  |

0…31 берілген кодтардағы символдар қызметтік кодтар деп аталады.

CHAR типіне қатынастар операциясы сонымен қатар құрылымдық функциялар қызмет ете алады:

CHR(B) – CHAR типінің функциясы, BYTE Типінің символының мәнін қайтарады.

UPCASE(CH) – CHAR Типінің функциясы, кіші әріппен жазылған болса, оны бас әріппен жазуға көмектеседі.

Мысалы:

Var

C1,c2:char;

Begin

C1:=UpCase(‘s’);

C2:=UpCase(‘ф’);

Writeln(c1,’ ‘,c2);

End.

Экранға S ф деген мәлімет шығады.

**Тізбектелген тип.**

Тізбектелген типпен тек қана тізбектеу арқылы ғана, соған арналған мәндер мен ғана жұмыс жасауға алдын ала берлген мәндер жиынтығы. Әр мән идентификатор бойынша емес, тізбекте қалай жақшаға алынып тұр сол бойынша сипатталады.

Мысалы:

Type

Colors=(red, white, blue);

Мысалы үшін тізбектелген типті қолдану кезінде ол бағдарламаны нақтылы немесе түсініктілеу қылып жібереді. Ол көп жағдайда күндерді, айларды тізу кезінде қолданылады. Бадарлама мысалы төмендегідей болуы мүмкін:

Type

АйТипі=(янв,фев,мар,апр,май,июн,июл,авг,сен,окт,ноя,дек);

Var

Ай:АйТипі;

Begin

…….

End.

*Негізгі әдебиеттер:*

1. Йенсен К, Вирт Н. Паскаль. Руководство для пользователя и описание языка. – М.: Финансы и статистика, 1982

2. Перминов О.Н. Программирование на языке Паскаль. – М.:Радио и связь, 1988

3. Фаронов В.В. Турбо Паскаль 7.0. Начальный курс. Учебное пособие. – «Нолидж», 1988.

4. Горбунов А.В., Антонова Н.А. Технология программирования на языке Паскаль с решением типовых задач.: Учебное пособие. – Караганда, 2003.

Тексеру сұрақтары:

1. Мәліметтер типі дегеніміз не?
2. Мәліметтер типі не үшін қолданылады.
3. Типтердің түрлері
4. Типтердің бір бірінен айырмашылығы.

Тапсырмалар:

1. 64,65,78,90,92,96,98 мәндерінің алдындағы мәнді қайтаратын бағдарлама жазу керек.
2. 23,56,89,45,21,23,36,69,78,45,11,2,3 мәндерінің артындағы мәндерді қайтаратын бағдарлама жазу керек.

**Лабораториялық жұмыс** №**4**

**Тақырыбы**: Турбо Паскаль тіліндегі операторлар

**Мақсаты:** Турбо Паскаль тіліндегі операторлармен жұмысты ұйымдастыру, олармен жұмыс жасау принциптерін меңгеру.

**Теориялық бөлім**

**Оператор** – брілген бағдарламалау тілінде, ЭЕМ-ақпаратты өңдеу процесіндегі кейбір аяқталған әрекеттерді көрсетуге арналған жазба. Оператор – тілдің негізгі түсініктерінің бірі. Әрбір оператор тілдің аяқталған фразасын білдіреді және мәліметтерді өңдеудің кейібр сатыларын анықтайды.

Меншіктеу операторлар

Бұл кез-келген бағдарламалау тіліндегі негізгі операторлар. Меншіктеу операторы теңдік белгісінің сол жағында орналасқан айнымалыға қандай да бір өрнектеп берілген жаңа мәнді меншіктейді.

A:=3 - бұл жазба a айнымалысына 3 мәні меншіктелетінін білдіреді.

A:=a+1, ал бұл жазба a айнымалысының бұрынғы мәніне бір саны қосылып және осы айнымалыға жауабы қосылады, ал егере a –ның мәні 3-ке тең болса, онда екінші меншіктеу операторы орындалғаннан кейін a айнымалысының мәні 4-ке тең болады. Бұрынғы мәні сақталмайды.

Мысалы:

Меншіктеу операторының көмегімен a және b айнымалыларының мәнін береміз, әрі қарай өрнектің мәнін есептейміз және шыққан нәтижені c айнымалысына меншіктейміз.

Бағдарлама фрагменті төмендегідей болады:

A:=2.3;

B:=-0.7

C:=(a\*a+b)/(sin(a+b)+2.5);

**Операторлық жақшалар:**

Паскаль тілінде операторлық жақша ретінде түйінді сөз алынады:

Begin (ашылған) және end (жабылған жақша).

Құрама операторға кіретін жай операторлар түрлі сан алуан. Олардың ішінде тілдің басқа да құрама операторлары болуы мүмкін. Құрама операторларды сатылап қабаттаса қолданылатын бірнешелеу бола береді. Бүкіл операторлар бөлімі бір құрама операторларды құрайды.

Begin (операторлар білімінің басы)

Begin

Begin

Begin

End

End

End

End (бағдарламаның соңы)

“begin” сөзінің саны “end” сөзінің санына сәйкес келуі керек.

Шартты оператор

Шартты оператор кейбір шарттарды тексеретін олардың тексерілуі нәтижесіне сәйкес келетін оператор болып табылады. Сондықтан шартты оператор - есептеу үрдісіндегі бағдарламалық құрал.

Шартты оператордың құрылымы келесідегідей болады:

IF <шарт> THEN <операторлар 1> ELSE <оператор 2>

Мұндағы IF, THEN, ELSE – резервті сөздер. (егер, онда, әйтпесе);

<шарт> - логикалық типтің туынды өрнегі;

<операторлар 1>, <оператор 2> - Турбо Паскаль тіліндегі кез-келген операторлар

Шартты оператор келесі алгоритм бойынша жұмыс жасайды. Алдынды <шарт> есептеледі. Егер нәтиже True (ақиқат) болса, онда <оператор 1> орындалады, ал <оператор 2 > бос жіребіледі; егер нәтиже FALSE (жалған) болса, онда керісінше <оператор 1 > жіберіледі, ал <оператор 2 > орындалады. Мысалы:

Var

X,y,max:Integer;

Begin

…..

if x> max then

y:=ma[

else

y:=x;

Осы фрагменттің орындалуы кезінде Y айнымалысының мәні X айнымалысының нәнін алады, егер оның мәні Max болмаса кері жағдайда Y Max тең болады.

ELSE бөлімі <оператор 2> шарты үшін бос жіберіледі. Онда True мәні үшін <оператор 1> өрнегі орындалады, кері жағдайда оператор бос жіберіледі.

Мысалы:

Var

X,y,max:Integer;

Begin

…..

if x>max then

max:=x;

y:=x;

Осы мысалда Y айнымалысы үнемі X айнымалысының мәніне тең болады, ал Max айнымалысы X мәнінің максимальды мін болып сақталады.

Осы екі оператордың кез-келгені әртүрлі типті болуы мүмкін, ал әр құрылымдық шартты оператор ELSE-нің шарты болуы мүмкін. Егер бірінші және екінші оператор орындалмаса онда ӘЙТПЕСЕ бөлігі орындалады. Мысалы:

Var

A,b,c,d:Integer;

Begin

A:=1;

B:=2;

C:=3;

D:=4;

If a>b then

If c<d then

If c<0 then

C:=0

Else

A:=b; {a 1-ге тең}

If a>b then

If c then

If c then

C:=0

Else

Else

Else

A:=b; {a 2 – ге тең}

Келесі бағдарлама 0..15 арасындағы ондық бүтін санды он алтылық санға түрлендіре отырып шығарады артынан алынған нәтижені экранға шығарады:

Program Hex; {Бағдарлама 0 мен 15 арасындағы санды ан алтылық жүйеге ауыстырады және нәтижені экранға шығарады}

Var

N:Integer; {енгізетін сан}

Ch:char; {нәтиже}

Begin

Write(‘n=’);

Readln(n); {саны енгіземіз}

{0..15 арасында санның жататындығын тексереміз}

If (n>=0) and (n<=15) then

Begin {ия, осы аралыққа сан жатады}

If n<10 then

Ch:= chr(ord(‘0’)+n)

Else

Ch:=chr(ord(‘A’)+n-10);

Writeln(‘n=’,ch)

End

Else {осы аралқыққа жатпайды}

Writeln(‘Қате’)

End.

Он алтылық санау жүйесінде 16 цифр қолданылады әр разряды 0..9 дейінгі санды қолданады.

Қайталау операторы

Турбо Паскаль тіліндегі қайталанатын операторларды үш түрлі оператормен есептеуге болады.

Санау операторы FOR циклі.

Оның құрылымы:

FOR <цикл параметрі> := <бастапқы мән> TO <соңғы мән> DO <оператор>.

Мұндағы:

FOR ,TO ,DO – резервті сөздер. (ол үшін, оған дейін, орындау);

<цикл параметрі> - цикл параметрі Integer типі кез-келген реттік айнымалы;

<бастапқы мән> - бастапқы мәні олс типтің өрнегі;

<соңғы мән> - соңғы мәні – сол типтің өрнегі;

<оператор> - Турбо Паскальдің тіынды операторы.

FOR операторын орындау кезінде <бастапқы мән> орындалады және <цикл параметр>:=<бастапқы мәнге> меншіктеледі. Осыдан кейін цикл қайталанады:

* <цикл параметрі> <= <соңғы мән> шартын тексеру; егер шарт орындалмаса, онда FOR операторы өз жұмысын аяқтайды;
* <оператор> операторы орындалады;
* <цикл параметрі> айнымалысы бірге дейін айналады.

Төмендегі мысал пернелер тақтасының көмегімен бүтін N санын енгізетін және 1 ден N-ге дейінгі барлық бүтін сандардың қосындысын шығаратын бағдарламаны қарастырайық.

Program Summ\_of\_Integer;

{N оң санын енгізеді және 1-ден N-ге дейінгі бүтін саедардың қосындысын есептейтін бағдарлама}

var

i,n,s:Integer;

Begin

Write( ‘N=’);

Readln(n); {N енгіземіз}

S:=0; {Қосындының бастапқы мәні}

For i:=1 to n do{қосындының есептейтін цикл}

S:=s+1;

Writeln(‘Қосынды =’,s); {Нәтижені шығару}

End.

Екі жағдайды басып айтуға болады. FOR опеараторымен жұмысты басқаратын шарт, ол <операторының> алдында тексеріледі. Егер шарт орындалмаса онда FOR операторы бірден орындалмайды. Екінші жағдай ол айналу қадамы үнемі +1 ге тең болады.

Сонымен қатар барқа да циклдік операторды қолдануға болады. Оның құрылымы төмендегідей болады:

FOR <цикл параметрі> := <бастапқы мән> DOWNTO <соңғы мән> DO <оператор>

TO қызметші сөзін DOWNTO сөзіне алмастырғанымыз оның айналу қадамы (-1) – ге тең болады.

Мысалы кез-келген санның қосындысын оң немесе теріс қылу үшін төмендегідей мысалды қолдануға болады:

……..

s:=0;

if n>=0 then

for i:=1 to n do

s:=s+1;

else

for i:=-1 downto n do

s:=s+1;

………

**WHILE цикл операторы. (тексеру алдындағы)**

Оның құрылымы:

WHILE <шарт> DO <оператор>.

Мұндағы:

WHILE , DO – резервті сөздер (осы уақытқа дейін , шарт орындалады, орындау)

<шарт> - логикалық типтің өрнегі;

<оператор> - Турбо Паскль тілінің туынды операторы.

Егер <шарт> өрнегі TRUE мәніне ие болса, онда <оператор> орындалады, осыдан кейін <шарт> -өрнегі оның тексерісі қайталанады. Егер <шарт> False мәніне тең болса, онда WHILE операторы өзінің жұмысын тоқтатады.

Келесі мысалды қарастырайық. “Машиналық эпсилонды” табатын бағдарламаны құрайық. Осы санға 1,0 қоссақ ол тағы да 1,0 мәнін беретін болсын.

Program EpsilonDetect;

Var

Epsilon:real;

Begin

Epsilon:=1;

While epsilon/2 +1>1 do

Epsilon:=epsilon/2

Writeln(‘Машиналық эпсилон=’,epsilon)

End.

**REPEAT…UNTIL цикл операторы. (тексеруден кейінгі)**

Оның құрылымы:

REPEAT <цикл денесі> UNTIL <шарт>

Мұндағы:

REPEAT, UNTIL – қызметші сөздер.

<цикл денесі> - Турбо Паскальдағы операторлардың туынды тізбегі;

<шарт> - логикалық типтің өрнегі;

<циклдің денесі> операторы <шарт> өрнегі орындалғаннан кейін бір рет орындалады. Егер оның мәні FALSE тең боласа; онда <цикл денесі> қайталанады, кері жағдайда REPEAT…UNTIL циклі жұмысын аяқтайды.

Ол үшін төмендегі мысалды қарастырайық. Бағдарлама экранға символды енгізеді және экранға оның кодын щшығарады. Ол жұмысын ақтау үшін екі рет Enter батырмасын басу керек болсын.

Program Codes\_of\_Chars;

Var ch:Char;

Const

CR=13; {}

Begin

Repeat

Readln(ch);

Writeln(ch,’=’, ord(ch))

Until ord(ch) = cr

End.

**Таңдау операторы.**

Таңдау операторы бір немесе бірнеше мүмкіндігі бар бағдарламаның ерекшелігін қарастыратын таңдау операторы болып табылады. Таңдау кілті деген параметр кез-келген реті үшін қолданылады.

Таңдау операторының құрылымы болып төмендегі табылады:

CASE <таңдау операторының кілті> OF <таңдау тізімі> [ELSE <оператор>] END

Мұндағы

CASE, OF, ELSE, END – резервті сөздер.

<таңдау операторының кілті> - таңдау кілті;

<таңдау тізімі> - бір немесе одан да көп конструкция;

<оператор> Турбо Паскаль тіліндегі туынды операторлар.

Таңдау операторы төмендегідей жұмыс жасайды. Алдында <таңдау кілті> өрнегінің мәні есептеледі, артынан тізбектелген ретпен <таңдау тізімі> есептеленетін мәнді теңестіру керек. Табылған оператор барлық таңдау операторының жұмысын аяқтайды. Нәтиже табылғанға дейін таңдау операторы жұмысты жалғастыра береді.

Төменде бағдарлама мысалы келтірілген.

Program Calc;

{Бағдарлама бірінші жолға екі санды енгізеді ал екінші жолға +,-,\*,/ шығарады артынан экранға сәйкес арифметикалық әрекетті шығарады.}

var

operation:char; {}

x,y,z:real;

stop:boolean;

begin

stop:=false;

repeat

Writeln;

Write(‘x,y=’);

Readln(x,y);

Write(‘операция:’);

Readln(operation);

Case operation of

‘+’: z:= x+y;

‘-’: z:= x-y;

‘\*’: z:= x\*y;

‘/’: z:= x/y;

else

stop:=true;

end;

if not stop then

writeln(‘нәтиже = ’,z)

until stop

end.

Белгілер және келесіге өту операторы

Кез-келген оператормен жұмыс жасау кезінде бір беттен екінші ретке қарай өту үшін операторға белгі беруге немесе басқа операторға өту үшін Турбо Паскль тілінде оператор қоладанылады.

Оператордың құрылымы мынадай:

GOTO <белгі>

Мұндағы:

GOTO –резервтік сөз;

<белгі> - белгі.

Белгі Турбо Паскаль туынды идентификатор.

Негізгі әдебиеттер

1. Фаронов В.В. Турбо Паскаль. Версия 7.0. Начальный курс. Учебное пособие. М.: Издательство Нолидж, 1997.
2. Епанишников А.М., Епанишников В.А. Программирование в среде Турбо Паскаль 7.0. М.: Диалог-МИФИ, 1995.
3. Йенсен К., Вирт Н. Паскаль. Руководство для пользователя. М.: Финансы и статистика, 1989.
4. Керниган Б., Плоджер Ф. Инструментальные средства программирования на языке ПАСКАЛЬ. М.: Радио и связь, 1985.
5. Абрамов С.А., Зима Е.В. Начала программирования на языке Паскаль.- М.:Наука,1987.-112 с.
6. Абрамов В.Г., Трифонов Н.П., Трифонова Г.Н. Введение в язык Паскаль. - М.: Наука, 1988.
7. Задачи по программированию. /С.А. Абрамов, Г.Г. Гнездилова, Е.Н. Капустина, М.И. Селюн. - М.:Наука, 1988.-224 с.

**Тексеру сұрақтары:**

1. Оператор деген не? Ол не үшін қолданылады?
2. Шартты оператор не үшін қолданылады?
3. Цикл қайталанатын оператор не үшін қолданылады?
4. Таңдау оператор не үшін қолданылады?
5. Операторларға жалпы сипаттама беру керек.

**Тапсырмалар:**

1. a,b,c сандарының ең үлкенін табу керек. Егер олардың арасында бірдей екі сан кездессе, үшіншіге сол мәнді меншіктеу керек.
2. A,b,c сандарын өсуі бойынша орналастыру керек.
3. A,b,c сандарын кемуі бойынша орналастыру керек.
4. A,b нақты сандары берілген. AX+B=0 теңдігіне бағдарлама құру керек.

**Лабораториялық жұмыс №5**

**Тақырыбы**: Құрылымдық типтер

**Мақсаты:** Турбо Паскаль тіліндегі кез-келген құрылымдық типпен танысу, оның басқа типтерге қарағандағы айырмашылығын ашу. Массивтер, жинақтар ға тоқталу. Олардың жұмыс жасалуын меңгеру.

Теориялық мәлімет

Турбо Паскаль тілінде құрылымдық типке төрт типті жатқызуға болады. Олар: массивтер, жазбалар, жинақтар және файлдар. Осы компоненттердің кез-келгенімен Паскаль тілінде жұмыс жасауға болады. Ол айнымалы немесе тұрақты болмасын құрылымдық тике жатқыза аламыз. Турбо Паскальдағы типтердің кез-келгенінің ұзындығы 65520 байәттан аспауы керек.

**Массивтер.**

Турбо Паскаль тіліндегі массив ұғымы басқа бағдарламалаудағы массивтерге ұқсас. Массивтің ерекшелігі ол басқа компоненттерге қарағанда оны реттеуге, реттік нөмірі бойынша қарапайым көрсеткішке келтіре алатын құрылымдық типтің компоненті болып табылады. Мысалы:

Type

Digit = array [0..9] of char;

Matrix =array [byte] of single;

Var

M: matrix;

D:digit;

I:integer;

Begin

……

m[17] :=ord(d[i-1])/10;

…….

End.

Массивтің типі келесі түрде беріледі:

<типтің атауы> =array [<бір немесе бірнеше индексті типтер>] of <тип>

Мұндағы

<типтің атауы> - дұрыс идентификатор;

array, of – резервті алдын ала белгілінеіп алынған сөздер; (массив, одан)

[<бір немесе бірнеше индексті типтер>] - бір немесе бірнеше индексті типтердің тізімі, олар үтір арқылы бөлініп жазылады әрі квадраттық жақшаға алынады;

<тип> - Паскаль тілінің кез-келген типі.

Массив екі түрлі болады. Ол бір өлшеді массив және екі өлшемді массив. Екі өлшемді массив ол анықтауыш. (матрица). Яғни оның өзіне тән синтаксисі болады. Бір өлшемді массивпен қалай жұмыс жасасақ екі өлшемді массивпен де солай жұмыс жасауға болады.

Индексті типтер негізінде Турбо Паскальда бірнеше реттік типтерді қолдануға болады. Тек қана LONGINT- типін ғана қолдануға болмайды. Массивті айнымалы ретінде сипаттау үшін айнымалылар бөлімінде алдын ала сипаттау қажет болады. Мысалы:

Var

A,b:array[1..10] of real;

Әдетте индексті типтің негізінде массивтің қолданылу диапазон-типі беріледі.

Құрылымдық типтің негізінде шексіздікпен жұмыс жасауға болады. Сондықтан да массивтермен жұмыс жасау, немесе массив элементтерімен жұмыс жасау да шексіздікке дейін бара алады. Дербес компьютер жадысында массив элементтері бірінің бірі артынан орналасады. Әдетте кіші адрестен үлкен адреске өту кезінде массивтің алғашқы индексі тез ауысады. Мысалы:

Var

A:array[1..2,1..2] of byte;

Begin

A[1,1]:=1;

A[2,1]:=2;

A[1,2]:=3;

A[2,2]:=4;

End.

Турбо Паскль тілінде бір операторға меншіктеу арқылы массивтің элементтерін келесі массивке беруге болады. Мысалы:

Var

A,b:array [1..5] of single;

Begin

…..

a:=b;

…..

end.

Осы меншіктеуден кейін А массивінің барлық эелемнті В массивніңі элементтерімен бірдей болады. Бірақ массвитерге қолданылатын қатынастар операциясы жоқ.

Мысалы үшін төмендегідей жазуға болмайды:

If a=b then …….

Екі массивті элементтері бойынша салыстыруға болады:

Var

A,b :array [1..5] of single;

Eq :boolean;

I :byte;

Begin

……

eq:=true;

for i:=1 to 5 do

if a[i] <> b[i] then

eq : =false;

if eq then

……

end.

**Жазбалар.**

Жазба – жазба өрісі деп аталатын белгіленген компоненттер санынан тұратын мәліметтер құрылымы. Массивтен айырмашылығы жазбаның компоненттері (өрістері) әр түрлі типті болуы мүмкін. Жазбаның сол немесе басқа компонентіне сілтеме жасалуы үшін өріс нөмірленеді.

Жазбаның хабарлау құрылымы төмендегідей болады:

<тип атауы> = record <өріс тізімі> end

Мұндағы

<тип атауы> - дұрыс идентификатор;

record, end – резервті сөздер (жазба, соңы);

<өріс тізімі> - жазба бөілктерінің тізбегін беретін өрістер тізімі. Олардың

арасында нүктелі үтір қойлады.

Жазбаның әр бөілігі бір немесе бірнеше өрістердің идентификаторларынан тұрады. Олардың арасын үтірмен бөлінеді. Идентификатордың артынан екі нүкте қойлады және өріс типтінің сипатталуы келтіріледі. Мысалы:

Type

BirthDay = record

Day,month :byte;

Year :word;

End;

Var

A,b:BirthDay;….

Осы мысалдағы BIRTHDAY (туылған күн) типіндегі DAY, MONTH, YEAR сияқты өрістер болады. Ал А және В BIRTHDAY типінің жазбаларын сақтап тұрады.

Массивтегі сияқты жазба типінің айнымалыларын басқа айнымалыға меншіктеуге болады. Мысалы:

A:=b;

Жазбаның әр компонентіне мүмкіндік алуы үшін айнымалы үшін құрылымдық атауды беруге боладыү Ол нүкте арқылы жазылады. Мысалы:

a.day:=27;

b.yaer:=1939;

Ішкі өрістер үшін кейбір нәрселерді нақтылауға тура келеді:

Type

Birthday = reord

Day,month:byte;

Year:word;

End;

Var

C:record

Name:string;

Bd:birthday;

End;

Begin

….

If c.bd.year=1939 then ……

End.

Жазбаның өрісіне мүмкіндікті жеңілдету үшін WITH қосу операторын қолдануға болады:

WITH <айнымалы> DO <оператор> Мұндағы:

WITH, DO – кілттік сөздер;

<айнымалы> - ішкі өрістердің тізімін беретін жазба типтіндегі айнымалы атын беру;

<оператор> - Турбо Паскальдағы кез-келген оператор.

Мысалы:

With c.bd do month : =9;

Бұл эквивалентті

With c do with bd do month : = 9;

немесе

with c,bd do month:=9;

немесе

c.bd.month:=9;

Турбо Паскаль тілінде нұсқаулы өрістер деп аталатын жазбаларды құолдануға да рұқсат береді. Мысалы:

Type

Forma = record

Name:string;

Case byte of

0: (birthPlace:string [40]);

1: (country :string [20]);

EntryPort : string[20];

EntryDate: 1..31;

ExitDate :1..31)

End;

CASE…OF сөзі нұсқаулы бөлікті көрсететін таңдау операторы болып табылады. Сондықтар бұл сөзінше бір құрылым болғандықтан артында END қызметші сөзі қолданылады.

**Жинақтар.**

Жинақтар – объектілер арасындағы бір бірімен байланысқан логикалық бір типті жиынтық болып табылады. Екі объектінің арасындағы байланыс ешқандай бағдарлама құрушымен анықталады, ол Турбо Паскальда ешқандай компиляциядан өтпейді. Жианққа кіретін элементтердің саны 0 ден 256 ға дейін болуы мүмкін. Жинақ нақтылы өзінің элементтері болмағандықтан массивтен және жазбадан ерекшеленеді. Екі жинақ тек қана оның элементтері бірдей болған жағдайда ғана эквивалентті болып есептеледі. Сонымен қатар жинақтағы элементтердің орналасу реті ең басты мүмкіндік емес.Жинақты анықтау және тапсырмалау мысалы төменде көрсетілген:

Type

DiditChar = set of ‘0’ …’9’;

Digit = set of 0..9;

Var

S1,s2,s3 : digitChar;

S4,s5,s6:digit;

Begin

…..

s1:=[‘1’, ‘2’, ‘3’];

s2:=[‘3’, ‘2’, ‘1’];

s3:=[‘2’, ‘3’];

s4:=[0..3, 6];

s5:=[4, 5];

s6:=[3…9];

…..

end.

Осы мысалдағы S1 және S2 эквивалентті, ал S3 және S2 – ке енгізілген, бірақ бір біріне эквивалентті емес.

Жинақтың сипатталу типі төмендегідей болады:

<типтін атауы> = SET OF <базалық тип>

Мұндағы:

<типтін атауы> - дұрыс идентификатор;

SET, OF - резервтелген сөз;

<базалық тип> - жинақ элементіндегі базалық тип. Оған кез-келген тізбекті типті қолдануға болады. Тек қана WORD, INTEGER, LONGINT типтерін ғана қолдануға болмайды.

Жинақтарды беру үшін жинақтың конструкторы қолданылады: жинақ элементінің арнайы тізімі. Олар нүктелі үтір арқылы бөлініп тұрады. Элементтің арнайылығы болып тұрақты немесе өрнек табылады, сонымен қатар диапазон-типі қолданылады.

Жианқтарға қолданылатын бірнеше операциялар бар. Олар:

\* жинақтарды тізу. Нәтижесі барлық жинақ элементіне тән болады, мсылаы: S4\*S6 [3] ке тең болады, S5\*S4 - бoc жинақ.

+ жинақтарды біріктіру. Нәтиже бірінші жинаққа сақталады, ал жетіспейтін элементтер екінші жнақтан алынады:

S4+S6 [0,1,2,3,4,5,6]; элементінен тұрады

S5+S6 [3,4,5,6,7,8,9]; элементінен тұрады

* жианқтардың айырмашылығы. Нәтиже екінші жинаққа тән болатын элементке тән болады.

S6-S5 [3,6,7,8,9] элементінен тұрады;

S4-S5 [0,1,2,3,6] элементінен тұрады;

= эквивалентті тексеру. Егер екі жинақ эквивалентті болса, онда True мәнін қайтарады.

<> эквиваленттілік еместі тексеру. Егер екі жианқ та эквивалентті болмаса онда True мәнін қайтарады.

<= жүрісті тексеру.

>= жүрісті тексеру.

IN жатуды тексеру. Бұндағы бірінші элемент - өрнек, екінші – жинақ.

Осыларға қоса екі операцияны қолдануға болады:

INCLUDE – жинаққа жаңа элемент қосады. Процедураға талап етілу төмендегідей болады:

INCLUDE (S,I)

Мұндағы

S – TsetBase базалық типіндегі элементтерден тұратын жинақ

I - TsetBase элементінің типі, осының көмегімен жинаққа элемент қосуға болады

EXCLUDE – жинақтан элементті алып тасау үшін қолданылады. Талап етілу

EXCLUDE ( S,I)

Негізгі әдебиеттер

1. Фаронов В.В. Турбо Паскаль. Версия 7.0. Начальный курс. Учебное пособие. М.: Издательство Нолидж, 1997.
2. Епанишников А.М., Епанишников В.А. Программирование в среде Турбо Паскаль 7.0. М.: Диалог-МИФИ, 1995.
3. Йенсен К., Вирт Н. Паскаль. Руководство для пользователя. М.: Финансы и статистика, 1989.
4. Керниган Б., Плоджер Ф. Инструментальные средства программирования на языке ПАСКАЛЬ. М.: Радио и связь, 1985.
5. Абрамов С.А., Зима Е.В. Начала программирования на языке Паскаль.- М.:Наука,1987.-112 с.
6. Абрамов В.Г., Трифонов Н.П., Трифонова Г.Н. Введение в язык Паскаль. - М.: Наука, 1988.

7. Задачи по программированию. /С.А. Абрамов, Г.Г. Гнездилова, Е.Н. Капустина, М.И. Селюн. - М.:Наука, 1988.-224 с.

Тексеру сұрақтар:

1. Массив дегеніміз не? Бағдарламалаудағы алатын қызметі.
2. Жинақ деген не? Бағдарламалаудағы алатын қызметі.
3. Жазбалар деген не? Бағдарламалаудағы алатын қызметі.
4. Жазбаның массивтен айырмашылығы неде?
5. Массивтің жинақтан айырмашылығы неде?
6. Жинақтың массивтен, жазбадан айырмашылығы неде?

Тапсырмалар.

1. 1..10 массиві берілген. Массив элементтерінің ішінен ең үлкен элементті табу керек.
2. 1..10 массиві берілген. Массив элементтерінің ішінен ең кіші элементті табу керек.
3. Массив элементтерінің бірінші элементін табу керек.
4. Массив элементтерінің соңғы элементін табу керек.
5. Массив элементтерінің бірінші элементін табу керек.
6. Массивтің екінші элементін 5 санына өзгерту керек.

Лабораториялық жұмыс №6

Тақырыбы: Процедуралар мен функциялар

Мақсаты: Турбо Паскаль тіліндегі процедура және функция ұғымына түсінік беру, оларды ұйымдастыруды меңгеру.

Процедура мен функцияның не үшін қажеттігін ашу.

Теориялық бөлім

Турбо Паскаль тілінде ерекше рәсімделуіне байланысты бағдарламаның өзіндік жеке фрагменті болып табылатын бөлікті процедура (функция) деп атайды.

Функцияның процедурадан айырмашылығы – функцияның денесін беретін бірнеше операторлардың орындалу нәтижесінде тек бір ғана мән немесе көрсеткіш болып табылатын бөліктің іске асыуы. Сондықтан функцияға талап жасалғанда айнымалылар мен тұрақтыларды қолдана аламыз. Турбо Паскаль тілінде функция мен процедураны бағдарлама ішіндегі бағдарлама деп атау қалыптастырылған. Бағдарлама ішіндегі бағдарлама негізгі бағдарламаны бірнеше бөліктерге бөледі. Осындай бөлу екі себеп бойынша орындалады.

Бірінші – бұл жады орын үнемдеу, әр бағдарлама ішіндегі бағдарлама негізгі бағдарламада бір данамен сақталады, ал осы уақытта оған жадыдан бірнеше рет талап жасай беруге болады. Бағдарлама ішіндегі бағдарламаны іске қосу үшін сондағы операторлардың орындалу тізбегі белсенді болады. Екіншісі – бағдарламаны жобалау тәсілін қолданудан тұрады. Осы жағдай бойынша бағдарлама үлкен негізгі бағдарлама ішіндегі бағдарламадан тұрады.

Атауды локализациялау

Процедураны шақыру үшін оның міндетті түрде атауы беріледі. Бағдарламаның құрылымы төмендегідей болады:

Бағдарлама ішіндегі бағдарламаны сипаттау – ол оның атауын және денесін беру деген сөз. Процедураның атауы берілгеннен кейін сипаттау бөлімі мен денесі басталады. Процедураны сипаттау иерархиясы төмендегідей болады.

Program …;

Procedure A;

Procedure A1;

…..

begin

….

End {A1};

Procedure A2;

…..

begin

….

End {A2};

Begin {A}

….

End {A};

Procedure B;

Procedure B1;

….

Begin

…

end {B1};

Procedure B2;

Procedure B21;

….

Бағдарлама ішіндегі бағдарламаны сипаттау

Атауы

Түрі төмендегідей болады:

PROCEDURE <атауы>;

FUNCTION <атауы>:<типі>;

Мұндағы

<атау> - бағдарламананың ішіндегі бағдарлама атауы

<типі> - функцияның нәтижесін қайтаратын тип

Мысал 1:

Турбо Пасвкаль тіліндегі дәрежеге шығару LN(X) және EXP(X) функциясының көмегімен шығаруға болатындығы белгілі. Төмендегі мысалда X,Y сандары беріледі және +Y дәрежесіндегі X саны шығарылады, артынан –Y дәрежесі шығарылады. Осы бағдарламадан шығу үшін Ctrl-Z және Enter батырмасын басу керек.

Var

X,y:real;

Function Power(a,b:real):real;

Begin {Power}

If a>0 then

Power:=exp(b\*ln(a))

Else if a<0 then

Power:=exp(b\*ln(abs(a)))

Else if b=0 then

Power :=1

Else

Power:=0

End {Power};

Begin

Repeat

Readln(x,y);

Writeln(Power(x,y):12:10,Power(x,-y):15:10);

Until EOF

End.

Мысал 2:

Бағдарламада екі 5 және 7 сандары беріледі. Біреуі айнымалы параметрі сияқты, екіншісі параметр мәні сияқты беріледі. Параметрлер мәні процедураны шақырғаннан кейін экранның берітіне шығарылады.

Const

A:integer = 5;

B:integer = 7;

Procedure Inc2 (var c:integer; b:integer);

Begin {Inc2}

C:=c+c;

B:=b+b;

Writeln(‘екі еселеу:’, c:5, b:5)

End {Inc2};

Begin

Writeln(‘Бастапқы мән:’, a:5, b:5);

Inc2(a,b);

Writeln(‘нәтиже:’, a:5, b:5)

End.

Нәтижесінде экранға мынандай мәлімет шығады:

Бастапқы мән: 5 7

Екі еселенген мән: 10 14

Нәтиже: 10 7

Негізгі әдебиеттер

1. Фаронов В.В. Турбо Паскаль. Версия 7.0. Начальный курс. Учебное пособие. М.: Издательство Нолидж, 1997.
2. Епанишников А.М., Епанишников В.А. Программирование в среде Турбо Паскаль 7.0. М.: Диалог-МИФИ, 1995.
3. Йенсен К., Вирт Н. Паскаль. Руководство для пользователя. М.: Финансы и статистика, 1989.
4. Керниган Б., Плоджер Ф. Инструментальные средства программирования на языке ПАСКАЛЬ. М.: Радио и связь, 1985.
5. Абрамов С.А., Зима Е.В. Начала программирования на языке Паскаль.- М.:Наука,1987.-112 с.
6. Абрамов В.Г., Трифонов Н.П., Трифонова Г.Н. Введение в язык Паскаль. - М.: Наука, 1988.

7. Задачи по программированию. /С.А. Абрамов, Г.Г. Гнездилова, Е.Н. Капустина, М.И. Селюн. - М.:Наука, 1988.-224 с.

Бақылау сұрақтары

1. Процедура дегеніміз не?
2. Функция дегеніміз не?
3. Процедура мен функция не үшін қолданылады?

Тапсырмалар:

1. Қисық сызықты трапецияның ауданын процедура арқылы емептеу керек.
2. Тригонометриялық функциялардың графигін процедура арқылы есептеп шығару керек.