ЛАБАРАТОРНАЯ РАБОТА № 8

Праграмаванне з выкарыстаннем працэдур і функцый

Мэта: навучыць праграмаваць з выкарыстаннем працэдур і функцый, вывучыць механізмы перадачы параметраў у працэдурах і функцыях, пазнаёміцца з рэкурсіяй.

Часта ў праграме знаходзяцца аднатыпныя часткі, якія праводзяць адныя і тыя ж вылічэнні, але з рознымі данымі. Такія часткі праграм мэтазгодна афарміць у выглядзе падпраграм. Іх выкарыстанне дазваляе:

1) зрабіць асноўную праграму больш нагляднай і кампактнай;

2) паменшыць аб’ём выкарыстоўваемай памяці ЭВМ;

3) скараціць час наладкі (таму што праграмаванне і наладку асноўнай праграмы і падпраграмы могуць здзяйсняць паралельна розныя праграмісты).

У Паскалі выдзяляюць два віды падпраграм: **працэдуры** і **функцыі**. Структура працэдур і функцый такая ж, як і структура асноўнай праграмы, г. зн. уключае загаловак і блок. У сваю чаргу блок складаецца з раздзела апісанняў і раздзела аператараў. Тэкст працэдуры ці функцыі можа быць размешчаны ў асноўнай праграме адным з наступных спосабаў:

1) размешчаны непасрэдна ў раздзеле апісаняў асноўнай праграмы;

2) запісаны ў асобны файл і ўстаўлены ў раздзел апісанняў асноўнай праграмы з дапамогай дырэктывы кампілятара (*$I імя-файла*);

3) аформлены ў выглядзе знешняга модуля.

Працэдуры

Апісанне працэдуры мае выгляд:

```
procedure імя_працэдуры (фармальныя параметры);
    раздзел апісанняў
begin
    раздзел аператараў
end;
```

і змяшчаецца ў асноўнай праграме ў раздзеле апісаняў. Раздзел апісанняў, як і ў асноўнай праграме, можа ўключаць раздзелы *label*, *const*, *type*, *var* і свой раздзел апісанняў працэдур і функцый. Фармальныя параметры прадстаўляюць сабой спіс пераменных з указаннем іх тыпу, якія паддзяляюцца адна ад адной кропкай з коскай. Гэтыя пераменныя не апісваюцца ў раздзеле апісанняў працэдур. Дапускаецца выкарыстанне працэдур без спісу параметраў.

Параметры працэдур могуць быць трох відаў:

1) параметры-значэнні (уваходныя параметры);

2) параметры-пераменныя (выхадныя параметры);

3) параметры працэдурнага тыпу.

Апісанне ўваходных параметраў працэдуры ў спісе фармальных параметраў мае такі выгляд:

спіс_пераменных1:тып1;спіс_пераменных2:тып2...

Адпаведна **апісанне выхадных параметраў** запісваецца так:

var спіс_пераменных1:тып1; var спіс_пераменных2:тып2;.

У Паскалі дапускаецца таксама выкарыстанне нетыпізаваных выхадных параметраў, якія маюць выгляд:

var спіс_пераменных;.

Выклік працэдуры ў асноўнай праграме ажыццяўляецца наступным чынам:

імя_працэдуры (фактычныя параметры);.

Тут фактычныя параметры прадстаўляюць сабой спіс параметраў, якія пералічаны праз коску (без указання іх тыпу). Паміж фармальнымі і фактычнымі параметрамі павінны быць суадносіны па колькасці параметраў, парадку іх запісу і тыпу даных. Імёны адпаведных параметраў могуць быць аднолькавымі ці рознымі.

Уваходнымі фактычнымі параметрамі, г. зн. тымі, якія перадаюцца ў працэдуру, могуць быць канстанты, пераменныя і выразы. Выхаднымі фактычнымі параметрамі (якія атрымліваюць значэнні з працэдуры) могуць быць толькі пераменныя. У якасці выхаднога параметра забараняецца выкарыстоўваць любую пераменную, якая мае спакаваны тып. Значэнне фармальнага ўваходнага параметра можа змяняцца ў самой працэдуры, аднак ніякага ўплыву на значэнне фактычнага ўваходнага параметра гэта не аказвае. У сілу гэтага ўваходны параметр (параметр-значэнне) не можа прадстаўляць сабой вынік работы працэдуры. Для гэтага неабходна выкарыстоўваць выхадныя параметры (параметры-пераменныя).

У якасці выхадных параметраў дапускаецца таксама выкарыстанне тыпізаваных канстант.

Прыклад 1. Вылічыць значэнне x^n:

```
program ap;
    var n:integer; x,y:real;
procedure s1 (k:integer; e:real; var d:real);
    var i:integer;
begin
    d:=1; for i:=1 to k do d:=d*e
end;
BEGIN
    writeln('увядзіце паказчык ступені n'); readln(n);
    writeln('увядзіце аснову ступені x'); readln(x);
```

```
    s1(n,x,y);
    writeln('y=',y:6:1)
END.
```

Прыклад 2. Знайсці найбольшае са значэнняў сум усіх элементаў двух прамавугольных матрыц:

```
program smatr;
    type matr=array[1..10,1..10] of real;
    var i,j:integer; a,b:matr; v,s1,s2,h:real;

procedure vvod(m1,n1:integer;var k:matr);
begin
    for i:=1 to m1 do
        for j:=1 to n1 do
            k[i,j]:=random(10);
end; {канец vvod}

procedure vivod(m1,n1:integer;k:matr);
begin
    for i:=1 to m1 do
        begin
            for j:=1 to n1 do write(k[i,j]:4:0);
            writeln;
        end;
    writeln;
end; {канец vivod}

procedure suma(m1,n1:integer;k:matr;var s:real);
begin
    s:=0;
    for i:=1 to m1 do
        for j:=1 to n1 do s:=s+k[i,j];
end; {канец suma}

BEGIN {асноўная праграма}
    randomize;
    vvod(5,5,a); vvod(4,6,b);
    vivod(5,5,a); vivod(4,6,b);
    writeln;
    suma(5,5,a,s1);
```

```
    writeln(s1:7:0);
    suma(4,6,b,s2);
    writeln(s2:7:0);
    if s1>s2 then h:=s1 else h:=s2;
    writeln('h=',h:8:2);
END.
```

Функцыі

Другі від падпраграм у мове Паскаль – функцыя. Афармляецца аналагічна працэдуры і адрозніваецца ад яе па структуры толькі загалоўка, агульны выгляд якога такі:

function імя_функцыі (фармальныя параметры): тып;.

Іншыя адрозненні функцыі наступныя:

1) функцыя мае толькі адзін вынік выканання (але можа мець некалькі ўваходных параметраў);

2) вынік абазначаецца імем функцыі. Таму ў раздзеле аператараў абавязкова павінен прысутнічаць аператар прысвойвання, у левай частцы якога стаіць імя гэтай функцыі;

3) у загалоўку функцыі абавязкова павінен быць указаны тып функцыі;

4) выклік функцыі ў асноўнай праграме ажыццяўляецца непасрэдна ўнутры выразу по яе назве з указаннем фактычных параметраў.

Прыклад 3. Вылічыць значэнне выразу

$$Z = \frac{5a^7 + \sin(a)}{2a^m}:$$

```
program stp;
    var m:integer; a,z,r:real;
function st (n:integer; x:real):real;
    var i:integer; y:real;
begin
    y:=1; for i:=1 to n do y:=y*x;
    st:=y
end;
BEGIN
    a:=0;
    repeat
        writeln('увядзіце значэнне a<>0');
        readln (a);
    until a<>0;
    writeln('увядзіце цэлы лік m');
```

```
    readln (m);
    z:= 5*st(7,a) + sin(a);
    if m=0 then r:=1
            else if m>0 then r:=st(M,A) else r:=st(-M,1/A);
    z:=z/(2*r);
    writeln( 'z=',z:6:2)
END.
```

Нетыпізаваныя параметры-пераменныя

Калі фармальны параметр працэдуры ці функцыі з'яўляецца нетыпізаваным параметрам-пераменнай, г. зн. апісаным без указання тыпу, адпаведны фактычны параметр можа быць пераменнай любога тыпу. Аднак унутры працэдуры ці функцыі нетыпізаваны параметр-пераменная будзе несумяшчальным з пераменнымі ўсіх іншых тыпаў да таго часу, пакуль яму не будзе прысвоены вызначаны тып з дапамогай прывядзення тыпу пераменнай.

Прыклад 4. Вывесці суму двух лікаў ці двух масіваў, пры гэтым выкарыстаць адну функцыю:

```
program lpr;
    var i,ind:integer; m1,m2:array[1..3] of integer; x1,x2:real;
function sum(n:integer; var y1,y2):real;
    type mas=array[1..maxint] of integer;
    var i:integer; s:real;
begin
    if n=0 then sum:=real(y1)+real(y2)
           else begin
                s:=0; for i:=1 to n do s:=s+mas(y1)[i]+mas(y2)[i];
                sum:=s
                end;
end;{канец sum }
BEGIN writeln('увод масіваў');
    for i:=1 to 3 do
                begin writeln('ww m1[',i,'], m2[',i,']'); read(m1[i],m2[i]); end;
    writeln('увядзіце два лікі '); readln(x1,x2);
    writeln(' x1+x2=',sum(0,x1,x2):5:1);
    writeln(' m1+m2=',sum(3,m1,m2):5:1);
END.
```

У гэтай праграме функцыя *sum* вяртае пры $n=0$ сумму дзвюх скалярных пераменных, пры $n<>0$ – суму дзвюх масіваў.

Вобласць дзеяння пераменных

Пераменныя, якія прадстаўлены ў раздзеле апісання асноўнай праграмы, дзейнічаюць у раздзеле аператараў асноўнай праграмы і ў любой яе падпраграме (працэдуры ці функцыі). Гэтыя пераменныя называюць **глабальнымі**. Пераменныя, якія апісаны ў падпраграме, дзейнічаюць толькі ў гэтай падпраграме і ў любой, аб'яўленай у ёй працэдуры і функцыі. Такія пераменныя называюцца **лакальнымі**, і яны недаступны для аператараў асноўнай праграмы і іншых.

У працэсе трансляцыі праграмы ўсе глабальныя пераменныя (у тым ліку глабальныя пераменныя знешніх модуляў, якія выкарыстоўваюцца праграмай) размяшчаюцца у сегменце даных. Яго максімальны памер можа быць недастатковым, тады трэба паменшыць колькасць глабальных пераменных (зрабіць іх лакальнымі) ці выкарыстоўваць дынамічныя пераменныя.

Пры выкананні праграмы пры кожнай актывізацыі (выкліку) працэдуры ці функцыі ўсё мноства іх лакальных пераменных змяшчаецца ў сегмент стэку. Пры завяршэнні работы працэдуры ці функцыі памяць, якая займаецца лакальнымі пераменнымі, выслабаняецца. У любы момант выканання праграмы агульны памер лакальных пераменных у актыўных працэдурах і функцыях не павінен перавышаць памер сегмента стэку. Гэты памер устанаўліваецца з дапамогай спецыяльных дырэктыў кампілятара.

Пры ініцыялізаванні ў падпраграмах тыпізаваных канстант яны ініцыялізуюцца толькі адзін раз – пры першым выкліку працэдуры ці функцыі. Пры кожным новым выкліку гэтай працэдуры ці функцыі лакальна апісаныя тыпізаваныя канстанты зноў не ініцыялізуюцца.

Напомнім, што кожны модуль (працэдура, функцыя, праграма) складаецца з загалоўка, які змяшчае адпаведна службовае слова *procedure*, *function*, *program* і блок. Пры напісанні праграм, якія маюць уложаныя модулі, неабходна ўтрымліваць наступныя правілы:

1) апісваць імёны пераменных у тым блоку, дзе яны выкарыстоўваюцца, калі гэта магчыма;

2) калі адна і тая ж пераменная выкарыстоўваецца ў двух і болей блоках, то апісваць яе неабходна ў самым знешнім з іх, які змяшчае ўсе астатнія блокі, што выкарыстоўваюць дадзеную пераменную;

3) калі пераменная, якая выкарыстоўваецца ў працэдуры, павінна захаваць сваё значэнне да наступнага выкліку гэтай працэдуры, то такую пераменную трэба апісаць толькі ў знешнім блоку, які змяшчае дадзеную працэдуру;

4) у Паскалі кожны выклікаемы модуль павінен быць апісаны да яго выкліку;

5) у Паскалі ў адным модулі можа быць апісана не больш 512 працэдур ці функцый.

Рэкурсіі

Працэдуры і функцыі могуць выклікаць самі сябе, гэта значыць валодаюць уласцівасцю **рэкурсіўнасці**. Умова поўнага заканчэння рэкурсіўнай працэдуры павінна знаходзіцца ў самой працэдуры.

Рэкурсіўныя модулі (працэдуры і функцыі) маюць адну з двух формаў: прамую ці ўскосную рэкурсію. У першым выпадку ў модулі змяшчаецца аператар выкліку гэтага ж модуля. У другім выпадку адзін модуль выклікае які-небудзь іншы, які альбо сам, альбо праз другія модулі выклікае зыходны модуль.

Напрыклад, аб фактарыяле ліку N можно сказаць, што

$$N!=1*2*3*\mathrm{K}*N \ (0!=1);$$

$$N!=(N-1)!*N \ (f(n)=f(n-1)\cdot n).$$

Другі запіс – рэкурсіўны.

Прыклад 5. Скласці табліцу значэнняў фактарыялаў лікаў ад 1 до 10 з выкарыстаннем рэкурсіі:

```
program lpr;
    { ¤ A-} {для сістэмы Экспрэс-Паскаль}
    var z:real; i:integer;
function fact(n:real):real;
begin
    if n=0 then fact:=1 else fact:=fact(n-1)*n;
end;
BEGIN
    for i:=1 to 10 do
                begin
                    z:=fact(i);
                    writeln('i=',i:3,' fact=', z:6:2);
                end;
END.
```