Тема: Цикли.

Мета: Познайомитися із видами операторів циклу для організації багатократних обрахунків та як за їх допомогою знайти суму та добуток послідовності чисел, значення максимуму чи мінімуму.

1. Алгоритмічна конструкція повторення та її різновиди: визначені та невизначені цикли, цикли із передумовою та післяумовою. Оператори циклів у мові програмування.

Запис послідовності команд не завжди може бути ефективним способом розв'язання поставленої задачі. Дуже часто в програмування потрібно виконати декілька разів одну й ту ж саму послідовність команд. Наприклад, потрібно на формі вивести числа від 1 до 10. Один із способів – написати десять рядків програмного коду. Проте такий спосіб є найгіршим. Для виведення, звичайно ж, потрібно використати цикл.

Цикл – це спеціальна конструкція мови, яка дозволяє запрограмувати багатократне виконання певного блоку команд.

В Delphi існує три типи циклів:

- Цикл із параметром (цикл зі змінною, значення якої збільшується (зменшується) задану кількість разів).
- Цикл із передумовою (умова перевіряється перед операторами циклу; якщо умова не виконується, то цикл завершується).
- Цикл із післяумовою (цикл виконується мінімум один раз, але можливо й більшу кількість; після кожного виконання операторів циклу перевіряється умова виходу із циклу).

Цикл із параметром.

Цикл із параметром дозволяє виконати набір команд фіксовану кількість разів, тобто, число ітерацій повинно бути відомим до початку виконання циклу. Особливістю даного циклу є та, що вводиться спеціальна змінна – лічильник, яка послідовно проходить вказаний діапазон значень. Значення цієї змінної може бути використане в блоці коду, який міститься в циклі. Ця змінна повинна бути локальною, тобто, бути описаною всередині модуля процедури або функції, в якій написано цикл, та не може змінюватись за допомогою операторів присвоювання всередині циклу.

Загальний вид конструкції циклу із параметром:

For лічильник:=початкове_значення [To / DownTo] кінцеве_значення Do
{дії}

Лічильник – оголошена раніше змінна перераховуваного типу (у більшості випадків – ціле число).

Початкове значення та кінцеве значення – межі діапазону, який послідовно проходить змінна – лічильник. Значення відповідно того ж типу даних, що й змінна – лічильник.

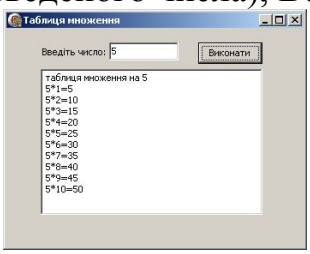
В залежності від співвідношення початкового та кінцевого значення використовується або ключове слово To (застосовується тоді, коли кінцеве значення більше від початкового – цикл йде по зростанню), або ключове слово DownTo

(застосовується тоді, коли кінцеве значення менше від попереднього – цикл йде по спаданню).

В ролі дії як правило вказується деяка команда або набір команд. Якщо команд декілька, то їх потрібно поміщувати в операторні дужки begin … end;

Приклад 1. Вивести на формі фрагмент таблиці множення для числа, яке вводиться користувачем.

Розв'язок. На формі розмістимо компоненти LabeledEdit (для введення числового значення з інтервалу від 1 до 9), Memo (для виведення таблиці множення для введеного числа), Button (для запуску програми на виконання).



Програмний код:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  Dialogs, StdCtrls, ExtCtrls;
type
  TForm1 = class(TForm)
    LabeledEdit1: TLabeledEdit;
    Button1: TButton;
    Memo1: TMemo;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
procedure TForm1.Button1Click(Sender: TObject);
var x,i :integer;
begin
 x:=strtoint (labelededit1.text);
 memo1.lines.clear;
 memo1.lines.add ('таблиця множення на '+inttostr(x));
 for i:=1 to 10 do
  memo1.lines.add (inttostr(x)+'*'+inttostr(i)+'='+inttostr(x*i));
end;
end.
```

Звертаємо увагу на одну помилку, яка досить-таки часто зустрічається. Пов'язана вона із використанням значенням змінної - лічильника після завершення циклу. Після завершення циклу значення змінної – лічильника не визначене! Тому,

якщо Ви маєте намір використовувати й надалі значення змінної, їй необхідно присвоїти нове значення явним чином.

Цикл із передумовою.

Цикл While («поки») – цикл, в якому умова знаходиться перед тілом циклу, а сам цикл виконується до тих пір, поки умова істинна (поки умова не стане хибною).

Загальний вигляд:

While {умова} Do

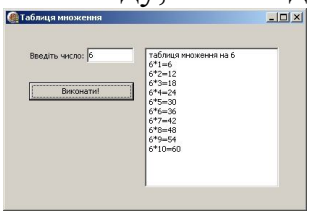
{дії}

В ролі умови задається логічний вираз. Ті операції, які будуть виконуватися в циклі, називаються тілом циклу.

Особливістю циклу із передумовою є та, що він може не виконатись жодного разу – у тому випадку, коли вказана умова зразу буде хибною. При цьому цикл може також стати «вічним» - якщо умова ніколи не набуде значення False. Саме тому необхідно слідкувати за тим, щоб завжди були задані умови для завершення роботи циклу.

Приклад 2. Вивести на формі фрагмент таблиці множення для числа, яке вводиться користувачем. Для розв'язання задачі скористатися циклом із передумовою While.

Розв'язок. Оскільки умова задачі еквівалентна прикладу 1, то зміниться лише фрагмент коду, який відповідає обробнику події OnClick кнопки Button.



Програмний код:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  Dialogs, StdCtrls, ExtCtrls;
type
  TForm1 = class(TForm)
    LabeledEdit1: TLabeledEdit;
    Button1: TButton;
    Memo1: TMemo;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
procedure TForm1.Button1Click(Sender: TObject);
var x,i: integer;
```

```
begin
 x:=strtoint (labelededit1.Text);
 memo1.lines.clear;
 memo1.lines.add ('таблиця множення на '+inttostr(x));
 i:=1;
 while i<=10 do
 begin
   memo1.Lines.Add(inttostr(x)+'*'+inttostr(i)+'='+inttostr(x*i));
   i:=i+1;
 end;
end;
end.
```

Цикл із післяумовою.

Цикл із післяумовою Repeat («повтор») працює точно так же, як і цикл із передумовою While, лише з однією відмінністю – умова циклу розміщується після тилу циклу, а не перед ним.

Загальний вигляд:

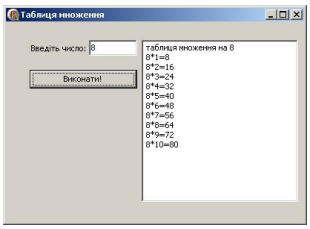
Repeat
 {дії}
Until {умова виходу із циклу};

Є декілька моментів, на які потрібно звернути увагу. По-перше, задається умова виходу із циклу, а в циклі While – умова продовження циклу. По-друге, при наявності декількох команд, які розміщені в тілі циклу, поміщувати їх в операторні дужки Begin … End; не потрібно – зарезервовані слова Repeat … Until самі складають аналогічний блок.

Цикл із післяумовою, на відміну від циклу із передумовою, завжди виконується хоча б один раз! Але, як і в циклі While, при неправильно написаній умові він може стати «вічним».

Приклад 3. Вивести на формі фрагмент таблиці множення для числа, яке вводиться користувачем. Для розв’язання задачі скористатися циклом із післяумовою Repeat … Until.

Розв’язок. Оскільки умова задачі еквівалентна прикладу 1 та прикладу 2, то зміниться лише фрагмент коду, який відповідає обробнику події OnClick кнопки Button.



Програмний код:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  Dialogs, StdCtrls, ExtCtrls;
type
  TForm1 = class(TForm)
```

```
    LabeledEdit1: TLabeledEdit;
    Button1: TButton;
    Memo1: TMemo;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
procedure TForm1.Button1Click(Sender: TObject);
var x, i: integer;
begin
 x:=strtoint (labelededit1.Text);
 memo1.Lines.Clear ;
 memo1.Lines.Add('таблиця множення на '+inttostr(x));
 i:=1;
 repeat
   memo1.Lines.Add(inttostr(x)+'*'+inttostr(i)+'='+inttostr(x*i));
   i:=i+1;
 until i>10;
end;
end.
```

2. Розв’язування задач, в яких використовуються обрахунки за ітераційними формулами.

Досить часто в задачах програмування для цілого числа необхідно отримати доступ до його цифр. Це може знадобитись для перетворення числа із однієї системи числення в іншу, для відшукання найменшої та найбільшої цифр даного числа тощо.

Задача 1. Дане чотирьохзначне ціле додатне число. Необхідно отримати суму цифр, з яких складається це число.

Розв’язок. На питання, яка цифра міститься в розряді одиниць, десятків, сотень, Ви легко відповісте, подивившись на число. Складемо алгоритм перетворення – розбиття чотирьохзначного числа на цифри за розрядами.

Для прикладу візьмемо число 1234.

Прогнозована відповідь до задачі: 1234 -> 1 + 2 + 3 + 4 = 10

Спробуємо виконати ці дії послідовно.

Виділимо розряд одиниць: 1234 mod 10 = 4

Зменшимо число в 10 разів (щоб розряд десятків перемітився на останнє місце в записі числа): 1234 div 10 = 123

Виділимо розряд десятків: 123 mod 10 = 3

Зменшимо число в 10 разів (щоб розряд сотень перемітився на останнє місце в записі числа): 123 div 10 = 12

Виділимо розряд сотень: 12 mod 10 = 2

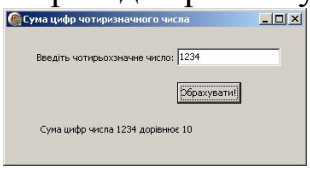
Зменшимо число в 10 разів (щоб розряд тисяч перемітився на останнє місце в записі числа): 12 div 10 = 1

Виділимо розряд тисяч: 1 mod 10 = 1

Тепер можна знайти суми чисел, які утворились в результаті виконання операції «залишок від ділення націло» (mod): 1 + 2 + 3 + 4 = 10

Проаналізувавши створений нами алгоритм, бачимо, що спостерігається певна циклічність у послідовності дій.

Форма для розв'язування задачі має вигляд:



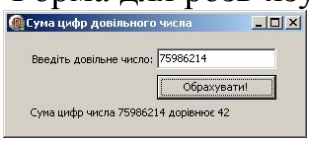
Програмний код має вигляд:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  Dialogs, StdCtrls, ExtCtrls;
type
  TForm1 = class(TForm)
    LabeledEdit1: TLabeledEdit;
    Button1: TButton;
    Label1: TLabel;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
procedure TForm1.Button1Click(Sender: TObject);
var x, i, s, ost, y: integer;
begin
 x:=strtoint(labelededit1.Text);
 s:=0;
 y:=x;
 for i:=1 to 4 do
   begin
     ost:=x mod 10;
     x:=x div 10;
     s:=s+ost;
   end;
  x:=y;
  label1.Caption :='Сума цифр числа '+inttostr(x)+' дорівнює '+inttostr(s);
end;
end.
```

Зауваження. Використання циклу For для розв'язування подібних задач доцільне лише у випадку, коли заздалегідь відомо кількість цифр у числі (як у нашому випадку). Якщо ж кількість цифр заздалегідь невідома, то набагато зручніше користуватись циклом While.

Задача 2. Дане довільне ціле додатне число. Необхідно отримати суму цифр, з яких складається це число.

Форма для розв'язування задачі має вигляд:



Програмний код має вигляд:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  Dialogs, StdCtrls, ExtCtrls;
type
  TForm1 = class(TForm)
    LabeledEdit1: TLabeledEdit;
    Button1: TButton;
    Label1: TLabel;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
procedure TForm1.Button1Click(Sender: TObject);
var x, i, s, ost, y: integer;
begin
 x:=strtoint (labelededit1.Text);
 y:=x;
 s:=0;
 while x>0 do
 begin
  ost:=x mod 10;
  x:=x div 10;
  s:=s+ost;
 end;
 x:=y;
 label1.Caption:='Сума цифр числа '+inttostr(x)+' дорівнює '+inttostr(s);
end;
end.
```

3. Вкладені цикли.

Багатократна повторювана частина алгоритму або програми, яка знаходиться в тілі іншого циклу, називається вкладеним циклом.

Іншими словами, для кожного значення змінної – лічильника зовнішнього циклу лічильник вкладеного циклу встигає «пробігти» всі свої значення.

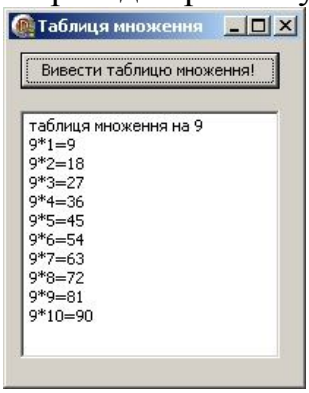
Ніяких спеціальних конструкцій для вкладених циклів не існує. Все працює так, як і у випадку використання звичайних циклів. Застосовуються вкладені цикли в основному для обробки таблиць (двовимірних масивів).

Найпростіший приклад використання вкладеного циклу – виведення не фрагменту, а всієї таблиці множення.

Задача 3. Вивести таблицю множення.

Розв'язання. Необхідно створити цикл, який послідовно надає значень від 1 до 9, формуючи стовпчик результатів добутків. А щоб вивести таблицю (перемноживши числа від 1 до 10), потрібний ще один такий же цикл.

Форма для розв'язування задачі має вигляд:



Програмний код має вигляд:

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  Dialogs, StdCtrls;
type
  TForm1 = class(TForm)
    Button1: TButton;
    Memo1: TMemo;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
implementation
{$R *.dfm}
procedure TForm1.Button1Click(Sender: TObject);
var i, j: integer;
begin
```

```
memo1.Lines.clear;
for i:=1 to 9 do
  begin
    memo1.Lines.add ('таблиця множення на '+inttostr(i));
    for j:=1 to 10 do
     memo1.Lines.add (inttostr(i)+'*'+inttostr(j)+'='+inttostr(i*j));
  end;
end;
end.
```

4. Переривання та продовження циклу.

Виконання фіксованої кількості ітерацій не завжди призводить до потрібного результату. Інколи в процесі виконання можуть виникнути ситуації, при яких логічно було б перервати цикл, не пройшовши його до кінця, тобто, відмінивши всі подальші ітерації. Така можливість існує – для цього необхідно скористатись командою Break. Ця команда завершує цикл, який виконується в даний момент, та продовжує виконання подальших команд програми. При цьому поточна ітерація до кінця не виконується – переривання відбувається саме в тому рядку, на якому записана команда Break.

Якщо цикл, виконання якого переривається командою Break, вкладений в інший цикл, то «зовнішній» цикл буде продовжувати своє виконання, тобто, команда Break зупиняє тільки один цикл, а не всі наявні.

Команда Break – це вихід із циклу. Інколи ж потрібно просто пропустити поточну ітерацію та перейти до наступної. Вручну це можна зробити, помітивши всі команди в блок умовного оператора, проте такий спосіб не зовсім зручний. Саме тому існує команда продовження циклу, вона називається Continue. Ця команда примушує цикл тут же перейти до наступної ітерації, не продовжуючи виконання поточної.

5. Теоретичні питання для самоконтролю.

1. Вкажіть причини використання циклів.
2. У чому відмінність циклу For, який використовує опцію To та DownTo?
3. Скільки разів виконається цикл For i:=2 To 8 Do?
4. Скільки разів виконається For i:=10 To 4 Do?
5. У чому відмінність циклів While та Repeat … Until?
6. Який цикл може не виконатися жодного разу?
7. Який цикл обов’язково виконається хоча б один раз?
8. Поясніть на прикладах використання операцій div та mod.
9. Опишіть алгоритм переведення цілого позитивного числа із десяткової системи у двійкову.
10. У чому особливість використання циклів For та While при переведенні числа до двійкової системи числення?
11. Що таке вкладені цикли?
12. Яким оператором здійснюється переривання циклу? В яких випадках таке переривання необхідне?