Лабораторна робота № 5. Циклічні алгоритми.

Розв'язання багатьох практичних завдань зводиться до виконання обчислень по тим самим залежностям, але при різних значеннях величин, які входять до виразу. Такий обчислювальний процес називається циклічним, а багаторазово повторювані ділянки цього процесу називаються **циклами**.

Розрізняють **регулярні** цикли з відомим числом повторень, умовою закінчення якого є досягнення параметром циклу свого кінцевого значення й **ітераційні** цикли, у яких умова повторення або закінчення циклу задається по деякому результату. В ітераційних циклах розрізняють цикли із **передумовою** і **післяумовою**. Синтаксис циклу із передумовою виглядає наступним чином:

```
While умова
   [оператори]
Wend
```

Обов'язковий елемент *умова* є числовим або строковим вираженням, яке приймає значення TRUE або FALSE.

Необов'язковий елемент *оператори* включає один або декілька операторів, що виконуються поки умова має значення TRUE.

Виконується оператор While... Wend у такий спосіб. Якщо *умова* має значення TRUE, виконуються всі оператори до інструкції Wend. Потім управління вертається інструкції While і знову перевіряється *умова*. Якщо *умова*, як і раніше, має значення True, процес повторюється. Якщо вона не має значення TRUE, виконання відновляється з інструкції, що розташована за інструкцією Wend.

Оператор циклу з післяумовою має синтаксис:

```
Do
   [оператори]
Loop Until [умова]
```

Необов'язковий елемент *умова* є числовим або строковим вираженням, яке приймає значення TRUE або FALSE.

Необов'язковий елемент *оператори* включає один або декілька операторів, що виконуються поки умова має значення FALSE.

Виконується оператор Do...Loop у такий спосіб. Якщо *умова* має значення FALSE, виконуються всі оператори після інструкції Do. Потім керування передається інструкції Until і знову перевіряється *умова*. Якщо *умова* як і раніше має значення FALSE, процес повторюється. Якщо вона має значення TRUE, керування передається наступному за інструкцією Loop Until оператору.

На рис. 5.3. наведені блок-схеми циклів із **передумовою** і **післяумовою**:

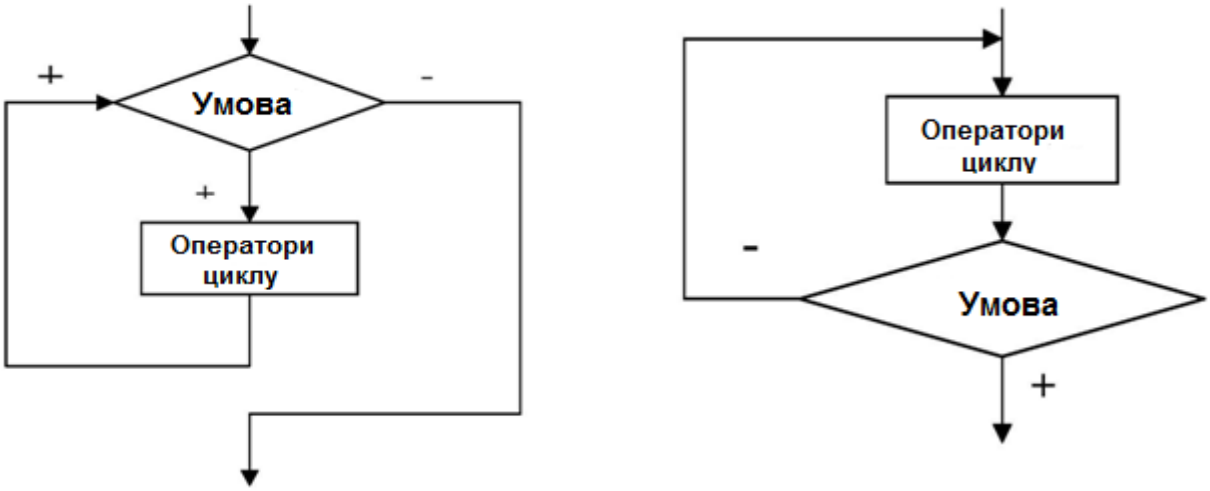


Рис. 5.3. Блок-схеми циклів із передумовою та післяумовою

Часто при складанні програм заздалегідь відома кількість повторень групи операторів; у таких випадках можна використовувати інструкцію For...Next**.** Оператор For...Next використовується для виконання наборів операторів зазначене число раз. Цикли For використовують у якості лічильника змінну, значення якої збільшується або зменшується при кожному виконанні циклу на певне значення. Синтаксис інструкції For...Next:

```
For лічильник = початок To кінець [Step крок]
  [оператори]
Next [лічильник]
```

Обов'язковий елемент *лічильник* повинен бути числовою змінною й не може мати тип Boolean або бути елементом масиву. Обов'язковий елемент *початок* містить початкове значення змінної *лічильник*, а обов'язковий елемент *кінець* — кінцеве значення змінної.

Необов'язковий елемент *крок* визначає значення, на яке змінюється лічильник при кожному виконанні тіла циклу. Якщо це значення не задане, за замовчуванням *крок* дорівнює одиниці. *Крок* може бути як позитивним, так і негативним.

Необов'язковий елемент *оператори* включає один або декілька операторів між For…Next, які виконуються зазначене число раз.

Інструкція For...Next працює в такий спосіб: початкове значення елемента *лічильник* порівнюється з кінцевим значенням. Якщо *крок* позитивний і початкове значення менше кінцевого, або якщо *крок* негативний і початкове значення більше кінцевого, то керування передається усередину тіла циклу. Після виконання всіх операторів у тілі циклу значення *крок* додається до поточного значення змінної *лічильник*. Після цього оператори тіла циклу або виконуються ще раз (на основі тієї ж умови, яка привела до початкового

виконання циклу), або цикл завершується й виконання триває з оператора, що розташований за Next.

При використанні циклічних конструкцій може виникнути необхідність дострокового виходу із циклу. У мові VBA оператором дострокового виходу є **`Exit`**, причому в циклах **`For`** він має вигляд **`Exit For`**, а в циклах, що починаються з **`Do`**, він має вигляд **`Exit Do.`**

Завдання 1

На території підприємства є гідрант із водовіддачею *Z* л/с. До нього послідовно підключаються *N* стовбурів, кожний з яких має витрату води *Ri* л/с. Визначити, на скільки перших стовбурів вистачить водовіддачі гідранта.

Аналіз проекту

Для реалізації проекту введемо наступні позначення. Тип *Integer*: *N* – кількість стовбурів, що підключаються, *i* – номер розглянутого стовбура; *k* – кількість стовбурів, які вже підключили. Тип *Single*: *Z* – загальна водовіддача гідранта, *Ri* – витрата води на і-*м* стовбурі; *S* – загальна витрата води по усім стовбурам.

Спочатку *S=0*. Витрата води проводиться по номерах стовбурів у наступний спосіб. Перевіряють, чи вистачить витрати води на перший стовбур (*i=1*), тобто чи виконується нерівність *S+R1 <= Z*. Якщо вистачить, то стовбур підключається до гідранта (*k=1*). Потім ті ж дії повторюють із другим стовбуром (*i=2*) і т.д. Загальна витрата води на *k* перших підключених стовбурах визначається по формулі *S=R1 +R2 +... +Rk*

Підсумовування припиняється, якщо всі стовбури підключені (*k=N*) або перевищена водовіддача (*k<i*). Остання нерівність показує, що кількість стовбурів, що підключені, менше, чим кількість розглянутих. Величина *k* може змінюватися в межах від *0* до *N*. Випадок *k=0* відповідає тому, що води не вистачило навіть на перший стовбур.

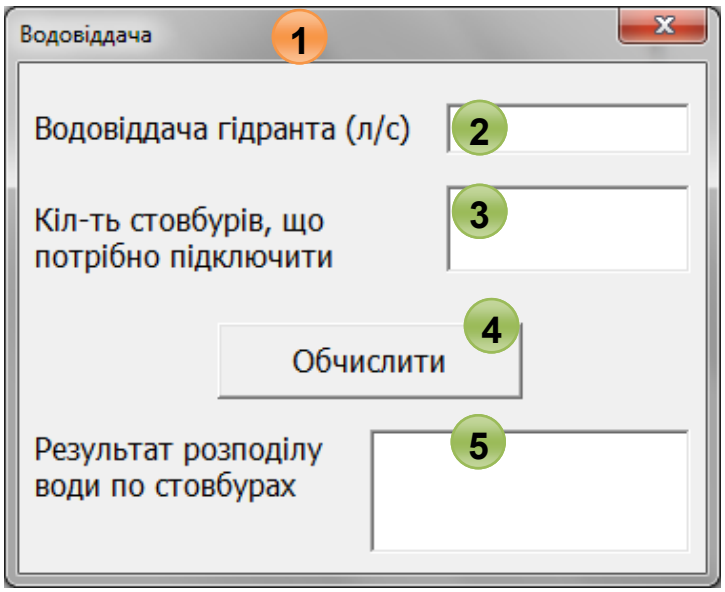
Порядок виконання

1. Перебуваючи в середовищі *Ms Word* або іншого додатку з *Microsoft Office*, натисніть **ALT+ F11**, щоб перейти в середовище VBA.

2. На панелі інструментів клацніть на кнопку **Insert Userform**  . З'являється вікно з формою *Userform1*.

3. Створіть форму згідно з ескізом, установивши у вікні властивостей **Properties** значення властивостей (**Name**) і **Caption,** які наведені

нижче. Для компонента *txtResult* також установіть властивість **Multiline**=*True*, що забезпечить багаторядковий вивід тексту.



Властивість (**Name**)
1 — *frmWater_loss*
2 — *txtZ*
3 — *txtN*
4 — *cmdCalculate*
5 — *txtResult*

4. Перейдіть у вікно коду, двічі клацнувши кнопку **Обчислити**, і введіть наступний текст процедури:

```
Private Sub cmdCalculate_Click()
Dim i As Integer 'Номер розглянутого стовбура
Dim k As Integer 'Кількість підключених стовбурів
Dim Ri As Single 'Витрата води на і-му стовбурі
Dim S As Single  'Загальна витрата води
'Початковий стан
S = 0
i = 0
k = 0
'Розподіл води по стовбурах
Do
  i = i + 1
  Ri = Val(Inputbox("Уведіть витрату води", i & "-й
стовбур"))
If S + Ri <= Val(txtz) Then
   S = S + Ri
   k = k + 1
Else
   Exit Do
End If
Loop Until (k = Cint(txtn))
'Висновок результату
If k = Cint(txtn) Then
  txtresult = "Води вистачить на всі стовбури"
Else
  txtresult = "Води вистачить тільки на " & Str(k) &
" перших стовбура(ів)"
```

```
End If
End Sub
```

5. Запустіть програму на виконання, клацнувши на вільному місці форми, а потім на кнопці **Run** (або натисніть клавішу **F5**).

6. У поле **Водовіддача гідранта (л/с)** уведіть число *100*, а в поле **Кіл-ть стовбурів, що потрібно підключити**, уведіть *3* і клацніть на кнопку **Обчислити**. У діалоговому вікні, що з'явилося, по черзі введіть числа: *20, 40, 30*. У поле **Результат розподілу води по стовбурах** з'явиться текст *Води вистачить на всі стовбури* . Не змінюючи значень у полях **Водовіддача гідранта (л/с)** та **Кіл-ть стовбурів, що потрібно підключити**, клацніть кнопку **Обчислити**. У діалоговому вікні, що з'явилося, по черзі введіть числа: *50, 40, 30*. У поле **Результат розподілу води по стовбурах** з'явиться текст *Води вистачить тільки на 2 перших стовбура(ів)*. Проаналізуйте роботу програми при Ваших вихідних значеннях.

7. Збережіть документ у файлі *Лаб5_1.doc* у своїй папці на жорсткому диску.

Завдання 2

Знайти середнє арифметичне з *N* чисел, що послідовно вводяться із клавіатури.

Аналіз проекту

Середнє арифметичне - це значення суми чисел, поділеної на їхню кількість. Тому, для розв'язку завдання будуть потрібні змінні для накопичення суми *S* (тип *Single*) і кількості чисел *k* (тип *Integer*). Основу алгоритму складе циклічна процедура: увести число, додати його до суми *S*. Ці оператори потрібно повторити *k* раз. На виході із циклу буде підрахована сума чисел, залишиться тільки розділити *S* на *k* і запам'ятати результат у змінну *R*, її тип - *Single* – результат розподілу не завжди буде цілим числом.

Порядок виконання

1. Перейдіть у середовище VBA, нажавши клавіші **ALT+F11.**

2. У вікні властивостей **Project** клацніть на кнопку **View code** (або виконаєте подвійне клацання мишкою на пункті **Thisdocument**). З'являється вікно коду VBA.

3. Перейдіть у вікно коду й уведіть наступний текст процедури:

```
Sub Середнє()
Dim S, R, a As Single, k As Integer
```

```
k = Inputbox("Уведіть кількість чисел для обчислення
середнього значення")
S = 0 'Очищення змінної для підсумовування
For i = 1 To k
a = Inputbox("Уведіть " & Str(i) & "-е число")
S = S + a 'Накопичуємо суму
Next
R = S / k
Msgbox ("Середнє арифметичне з " & Str(k) &     "
чисел- "& Str(R))
End Sub
```

4. Запустіть програму на виконання, клацнувши на кнопці **Run**  (або натисніть клавішу **F5**).

5. Перевірте правильність роботи програми шляхом введення різних чисел і визначення їх середнього арифметичного.

Зверніть увагу, що змінна *S* попередньо обнуляється («чиститься») до початку циклу. Чищення необхідно робити, тому що при перекладі програми на машинну мову під змінні приділяється пам'ять, у якій може перебувати стара інформація. Якщо скасувати оператор `S = 0`, то в наступному присвоюванні `S = S + a` при обчисленні правої частини до значення *a* буде додано невідоме значення *S* і результат буде невірний. Типова помилка, коли операції обнуління поміщають у тіло циклу, після заголовку.

6. Збережіть документ у файлі *Лаб5_2.doc* у своїй папці на жорсткому диску.

Завдання 3

Порахувати добуток чисел, що вводяться із клавіатури доти, поки не зустрінеться значення 0.

Аналіз проекту

Для розв'язання завдання будуть потрібні змінні для накопичення добутку *P* (тип *Single*). Тут заздалегідь невідомо, скільки чисел буде введено, тому краще скористатися циклом While…Wend .

Порядок виконання

1. Перейдіть у середовище VBA, нажавши клавіші **ALT+F11.**

2. У вікні властивостей **Project** клацніть на кнопку **View code** (або виконаєте подвійне клацання мишкою на пункті **Thisdocument**). З'являється вікно коду VBA.

3. Перейдіть у вікно коду й уведіть наступний текст процедури:

```
Sub Добуток()
Dim a, P As Integer
a =Val(Inputbox("Уведіть ненульове число"))
P = 1 'Очищення змінної для добутку
While a <> 0
P = P * a 'Накопичуємо добуток
a = Val(Inputbox("Введіть наступне число"))
Wend
Msgbox (" Добуток чисел дорівнює: "& Str(P))
End Sub
```

4. Запустіть програму на виконання, клацнувши на кнопці **Run** (або натисніть клавішу **F5**).

5. Перевірте правильність роботи програми шляхом уведення різних чисел і визначення їх добутку.

6. У першому рядку ми вимагали, щоб спочатку було введено ненульове число. А що, якщо все-таки користувач програми ввів 0? Тоді цикл не проробить жодного разу, а результат буде *P=1*. «Захиститися» від першого нуля можна додавши в рядок:

```
a =Val(Inputbox("Уведіть ненульове число"))
```

«оберігаючий» оператор Do…Loop з перевіркою на нуль наступного виду:

```
Do
 a =Val(Inputbox("Уведіть ненульове число"))
Loop Until a <> 0
```

Зверніть увагу, що «чищення» змінної *P* полягає в присвоєнні їй значення 1, тому що *P* бере участь у добутку *P=P*a* й обнуління *P* привело б до нульового результату всієї програми. Виконання циклу триває доти, поки не введено 0 у змінну *a*.

7. Перевірте правильність роботи програми шляхом уведення першого нульового числа. Перший оператор циклу не дозволить продовжити програму, якщо вводяться нулі: умова виходу із циклу *a* <>0.

8. Збережіть документ у файлі *Лаб5_3.doc* у своїй папці на жорсткому диску.

Завдання 4

Знайти максимальне число з 10-ти довільних чисел, що вводяться із клавіатури.

Аналіз проекту

При знаходженні максимуму в послідовності чисел, потрібно визначити змінну для зберігання максимального числа *Max* (тип *Single*). Потім кожне введене число порівнюється зі значенням *Max* і, якщо це число перевищує *Max,* то воно тепер вважається максимальним і привласнюється змінній *Max,* стираючи попереднє значення. Таким чином, основний оператор алгоритму розв'язку - це цикл, у якому тіло становлять дві дії: уведення нового значення й перевірка, чи є це значення максимальним. По закінченню циклу в *Max* буде перебувати найбільше зі значень.

Для порівняння першого числа потрібно визначити початкове значення змінної *Max.* Вірне рішення – обрати будь яке з аналізованої послідовності, наприклад, перше. Невірне рішення - обрати будь-яке число, наприклад, 0. Нуль згодиться, якщо вводяться тільки позитивні числа (тоді кожне з них «закриє» первісний максимум). Але, якщо можуть бути введені тільки негативні числа, те цей 0 і виявиться максимальним, хоча й не був уведений. Рішення буде невірним.

Отже, оберемо в якості початкового значення перше із чисел, що вводяться, і відкриємо цикл із перевіркою 9-ти чисел, що залишилися на максимум. Оскільки число кроків відомо, простіше скористатися циклом For…Next.

Порядок виконання

1. Перейдіть у середовище VBA і викличте вікно коду VBA.
2. У вікні коду введіть наступний текст процедури:

```
Sub МаксЧисло()
Dim a, max As Single 'тип змінних – Single
max = Val(Inputbox("Уведіть 1-е число"))
For i = 2 To 10
a = Inputbox("Уведіть " & Str(i) & "-е число")
If a > max Then max = a
Next
Msgbox ("Максимальне число - " & Str(Max))
End Sub
```

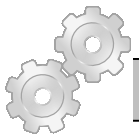
3. Запустіть програму на виконання й перевірте правильність роботи програми шляхом уведення різних чисел і визначення з них максимального.

Аналогічно вирішуються завдання на пошук мінімуму, потрібно тільки замінити знак нерівності й перепозначити змінну (для ясності): замість *Max* побрати, наприклад, *Min*.

4. Збережіть документ у файлі *Лаб5_4.doc* у своїй папці на жорсткому диску

Завдання для самостійної роботи

1. Знайти максимальне з негативних елементів серед довільних 20 чисел, що вводяться із клавіатури.
2. Знайти перший негативний член послідовності *cos(n)* для *n*, що змінюється так: *n*=0,1; 0,2; 0,3…
3. Обчислити позитивні значення функції *y=cos(3x)+4sin(x-3)* для *x*, що змінюється на відрізку [-15,10] із кроком 1. Результат вивести на робочий аркуш *Ms Excel* з використанням функції *Cells*.
4. Знайти кількість чисел, кратних трьом, з послідовності, що вводиться із клавіатури доти, поки не зустрінеться нуль.



Результат роботи:

1. Файл *Лаб5_1.doc* з формою *Водовіддача*
2. Файли *Лаб5_2.doc, Лаб5_3.doc, Лаб5_4.doc* з виконаними завданнями.

Питання для самоконтролю

1. Що таке циклічні алгоритми?
2. Яким елементом на блок-схемі позначається оператор циклу?
3. Що таке регулярні й ітераційні цикли? У чому їх відмінність?
4. Приведіть синтаксис оператора циклу Do…Loop і While…Wend.
5. У чому відмінність циклу з післяумовою від циклу із передумовою? У яких випадках який цикл потрібно використовувати?
6. Приведіть синтаксис оператора циклу For…Next. При розв'язку яких завдань використовується даний оператор?
7. Яких правил слід дотримуватися при побудові вкладених циклів?