

Міністерство освіти і науки України  
Національний технічний університет України «Київський  
політехнічний  
інститут імені Ігоря Сікорського"  
Факультет інформатики та обчислювальної техніки  
Кафедра автоматизованих систем обробки інформації  
і управління

Звіт

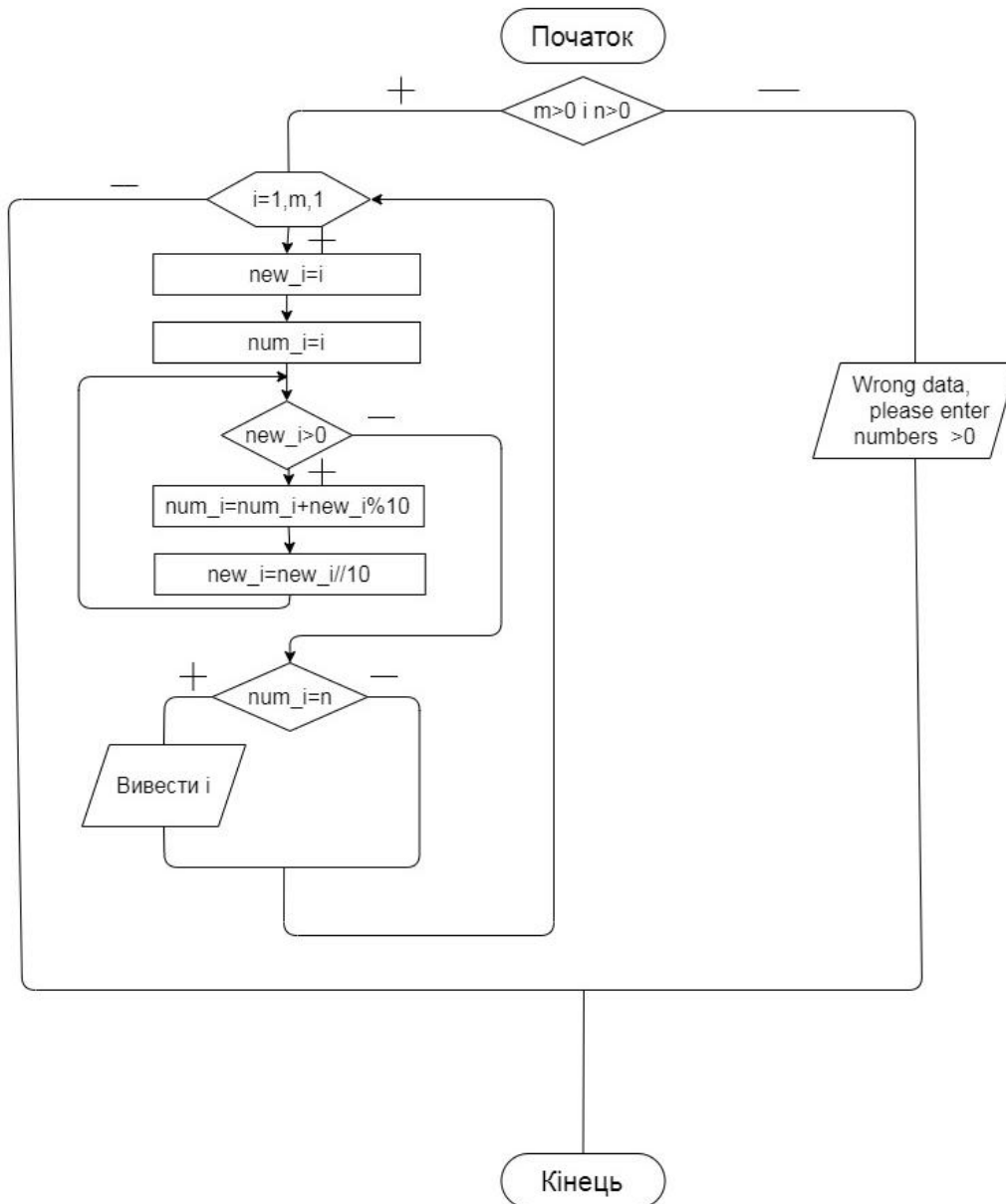
з лабораторної роботи No5 з дисципліни  
«Основи програмування»  
«Дослідження складних циклічних алгоритмів»

Виконав студент ІП-01 Адамчук Антон Іванович

## Варіант 18

**Задача.** Ввести натуральні числа  $m$  та  $n$ . Як результат вивести усі натуральні числа, що менші за  $m$ , сума цифр яких дорівнює  $n$ .

**Блок-схема:**



# Код програми на C++:

```
#include <iostream>

using namespace std;

int main()
{
    int m,           // Межа чисел, які будемо перевіряти (1,m)
    n;               // Сума цифр числа, за якою ми будемо шукати числа-результати

    cout<<"Enter number m: ";
    cin>> m;

    cout<<"Enter number n: ";
    cin>> n;

    if((m>0)and(n>0))
    {
        for(int i=1; i<m;i++)
        {
            int new_i=i;    // Змінна для обробки значення і без його втрати
            int num_i=0;    // Сума цифр числа і
            while(new_i>0)  // Знаходження суми цифр числа і
            {
                num_i=num_i+new_i % 10;
                new_i=new_i/10;
            }
            if(num_i==n)    // Перевірка умови рівності суми цифр числа та n
            {
                cout<< i<<" ";
            }
        }
    }

    else { cout<<"Wrong data, please enter numbers >0"; }
}
```

Копії екранних форм:

### Тест №1

```
Enter number m: -3
Enter number n: 30
Wrong data, please enter numbers >0
```

### Тест №2

```
Enter number m: 30
Enter number n: -3
Wrong data, please enter numbers >0
```

### Тест №3

```
Enter number m: 14
Enter number n: 3
3 12
```

Обчислимо вручну для перевірки правильності роботи програми:

Вхідні дані	m = 14    n = 3
Обробка даних:	<p>1) <u>Перевірка умови <math>m &gt; 0</math> і <math>n &gt; 0</math> -&gt; true</u></p> <p>2) <u>Запускається цикл від 1 до m</u></p> <p><i>Ітерація №1</i></p> <ol style="list-style-type: none"><li>1. <math>i=1</math>    <math>new\_i=0</math>    <math>num\_i=0</math></li><li>2. Обчислюється суми цифр числа <math>i</math>, результат: <math>num\_i=1</math></li><li>3. Перевірка чи сума цифр числа дорівнює заданому <math>n</math>: <math>num\_i == n?</math>    -&gt; false</li><li>4. Перевірка дала false, отже дане число <math>i</math> не виводиться</li></ol> <p><i>Ітерація №2</i></p> <ol style="list-style-type: none"><li>1. <math>i=2</math>    <math>new\_i=0</math>    <math>num\_i=0</math></li><li>2. Обчислюється суми цифр числа</li></ol>

i, результат: num\_i=2

3. Перевірка чи сума цифр числа дорівнює заданому n:

num\_i==n? -> false

4. Перевірка дала false, отже дане число i не виводиться

#### *Ітерація №3*

1. i=3 new\_i=0 num\_i=0

2. Обчислюється суми цифр числа i, результат: num\_i=3

3. Перевірка чи сума цифр числа дорівнює заданому n:

num\_i==n? -> true

4. Перевірка дала true, отже дане число i виводиться на екран

.....

#### *Ітерація №10*

1. i=10 new\_i=0 num\_i=0

2. Обчислюється суми цифр числа i, результат: num\_i=1+0=1

3. Перевірка чи сума цифр числа дорівнює заданому n:

num\_i==n? -> false

4. Перевірка дала false, отже дане число i не виводиться

#### *Ітерація №11*

1. i=11 new\_i=0 num\_i=0

2. Обчислюється суми цифр числа i, результат: num\_i=1+1=2

3. Перевірка чи сума цифр числа дорівнює заданому n:

num\_i==n? -> false

4. Перевірка дала false, отже дане число i не виводиться

#### *Ітерація №12*

1. i=12 new\_i=0 num\_i=0

2. Обчислюється суми цифр числа

	<p>i, результат: num_i=1+2=3</p> <p>3. Перевірка чи сума цифр числа дорівнює заданому n: num_i==n? -&gt; true</p> <p>4. Перевірка дала true, отже дане число і виводиться на екран</p> <p><i>Ітерація №13</i></p> <p>1. i=13 new_i=0 num_i=0</p> <p>2. Обчислюється суми цифр числа i, результат: num_i=1+3=4</p> <p>3. Перевірка чи сума цифр числа дорівнює заданому n: num_i==n? -&gt; false</p> <p>4. Перевірка дала false, отже дане число і не виводиться</p> <p><u>3) Цикл зупиняється, адже і набуває значення 14, в той час як заданий цикл – строгого порядку (i&lt;14)</u></p>
<b>Вихідні дані:</b>	<b>3 12</b>

## Код програми на Python:

```

m=int(input("Enter number m: "))      # Межа чисел, які будемо перевіряти (1,m)
n=int(input("Enter number n: "))      # Сума цифр числа, за якою ми будемо шукати числа-результати
if((m>0) and (n>0)):
    for i in range(1,m,1):
        new_i=i                      # Змінна для обробки значення і без його втрати
        num_i=0                      # Сума цифр числа i
        while (new_i>0):              # Знаходження суми цифр числа i
            num_i=num_i+new_i%10;
            new_i=new_i//10
        if (num_i==n):
            print(i, end=" ")          # Перевірка умови рівності суми цифр числа та n
    else:
        print("Wrong data, please enter numbers >0")

```

## Копії екранних форм:

### Тест №1

Підставимо раніше перевірені значення

m=14 n=3

```
Enter number m: 14
Enter number n: 3
3 12
```

Числа дорівнюють тому, що ми отримали при перевірці алгоритму вручну( перевірка вище).

### Тест №2

```
Enter number m: -30
Enter number n: 3
Wrong data, please enter numbers >0
```

### Тест №3

```
Enter number m: 30
Enter number n: -3
Wrong data, please enter numbers >0
```

**Висновок.** Особливістю складних циклічних операцій є обробка великого об'єму інформації та виконання складних обчислень, які потребують два або більше цикли. Так, розв'язання завдань такого типу стає значно легшим, швидшим та ефективнішим. Якщо задача не містить обчислень, які потребують більше одного цикла, то використання складних циклічних операцій слід уникати, щоб не витратити зайвих ресурсів вашого пристрою.