

Увод в програмирането

Управление на процесите в компютъра (част 2)

2017-2018 г.

ФМИ, специалност „Софтуерно инженерство“

- Оператори за цикъл:
 - for
 - while
- Оператори break и continue
- Примери

Оператори за цикъл

- Операторите за цикъл се използват за реализиране на изчислителни процеси, при които оператор или група оператори се изпълняват многократно за различни стойности на техни параметри
- Съществуват два вида циклични процеси:
 - Индуктивни
 - Итеративни

Оператори за цикъл ...

индуктивен цикличен процес

- Цикличен изчислителен процес, при който броят на повторенията е известен предварително, се нарича индуктивен цикличен процес.
- Пример: По дадени цяло число n и реално число x , да се намери сумата

$$S = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \dots + \frac{x^n}{n!}.$$

Оператори за цикъл ...

индуктивен цикличен процес

- Ако S има начална стойност 1, за да се намери сумата е необходимо n пъти да се повторят следните действия:
 - конструиране на събираемо
$$\frac{x^i}{i!}, (i = 1, 2, \dots, n)$$
 - добавяне на събираемото към S .

Оператори за цикъл ...

итеративен цикличен процес

- Цикличен изчислителен процес, при който броят на повторенията не е известен предварително, се нарича итеративен цикличен процес. При тези циклични процеси, броят на повторенията зависи от някакво условие.

- Пример: По дадени реални числа x и $\varepsilon > 0$, да се намери сумата

$$S = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

- където сумирането продължава до добавяне на събираемо, абсолютната стойност на което е по-малка от ε .

Оператори за цикъл ...

итеративен цикличен процес

- Ако S има начална стойност 1, за да се намери сумата е необходимо да се повторят следните действия:
 - Конструирание на събираемо
$$\frac{x^i}{i!}, (i = 1, 2, \dots)$$
 - Проверка дали абсолютната стойност на последното добавено към сумата S събираемо е по-малка от ε
 - Ако да, то добавяне на събираемото към S
- В този случай, броят на повторенията зависи от стойностите x и ε

Оператори за цикъл ...

- В езика C++ има три оператора за цикъл:
 - *оператори while и do/while*
Използват се за реализиране на произволни циклични процеси – индуктивни и итеративни
 - *оператор for*
Чрез него също могат да се реализират произволни циклични процеси, но се използва главно за реализиране на индуктивни циклични процеси.

Оператори за цикъл ...

Оператор *while*

- *Синтаксис*

while (<условие>) <оператор>

където

- while (докато) е запазена дума;
- <условие> е булев израз;
- <оператор> е произволен оператор. Той описва действията, които се повтарят и се нарича **тяло на цикъла**.

Оператори за цикъл ...

Оператор *while*

- *Семантика*

Пресмята се стойността на <условие>. Ако тя е false, изпълнението на оператора *while* завършва без да се е изпълнило тялото му нито веднъж. В противен случай, изпълнението на <оператор> и пресмятането на стойността на <условие> се повтарят докато <условие> е true. В първия момент, когато <условие> стане false, изпълнението на *while* завършва

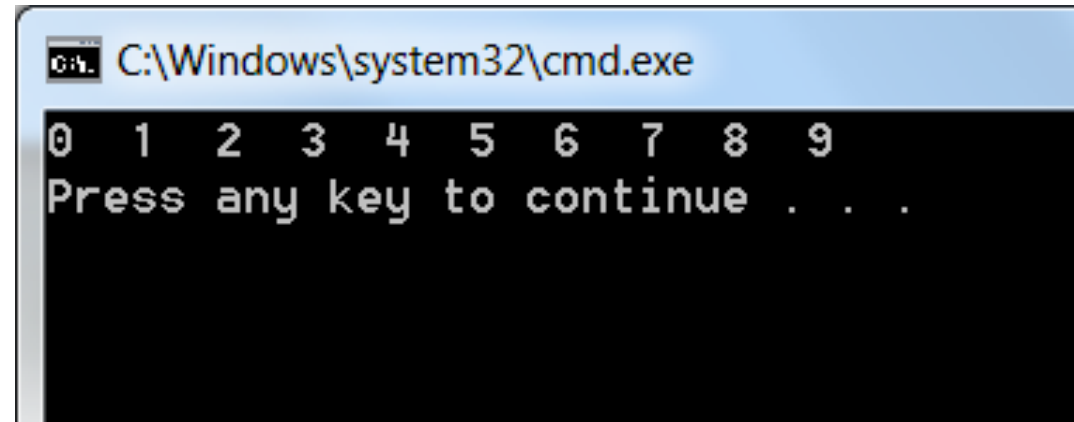
Оператори за цикъл ...

Оператор *while*

- *Забележки:*
 1. Ако е необходимо да се изпълнят многократно няколко оператора, те трябва да се оформят като блок.
 2. Следствие разширената интерпретация на true и false, частта <условие> може да бъде и аритметичен израз.
 3. Тъй като първото действие, свързано с изпълнението на оператора while, е проверката на условието му, операторът се нарича още **оператор за цикъл с пред-условие**.

Цикъл while

```
int main()  
{  
    int Counter = 0;  
  
    while (Counter < 10)  
    {  
        cout << Counter << " ";  
        Counter++;  
    }  
  
    cout << endl;  
}
```



A screenshot of a Windows command prompt window. The title bar shows the path 'C:\Windows\system32\cmd.exe'. The command prompt displays the output of the C++ program: the digits 0 through 9 are printed on the first line, followed by a second line that says 'Press any key to continue . . .'.

Цикъл do...while

Синтаксис

```
do  
    <тяло на цикъла>  
while( <условие> );
```

Семантика

1. Изпълнява се <тяло на цикъла>
2. Оценява се <условие>
 - А. Ако то е истина преминаваме обратно на 1.
 - В. Ако то не е истина, изпълнението на цикъла се прекратява.

Цикъл do...while

- За разлика от **while**, цикълът **do...while**, гарантира поне едно преминаване през тялото на цикъла:

```
int Input = 0;

while(Input > 0)
{
    cout << "*";
    Input--;
}
```

```
int Input = 0;

do
{
    cout << "*";
    Input--;
} while(Input > 0);
```

do...while (пример)

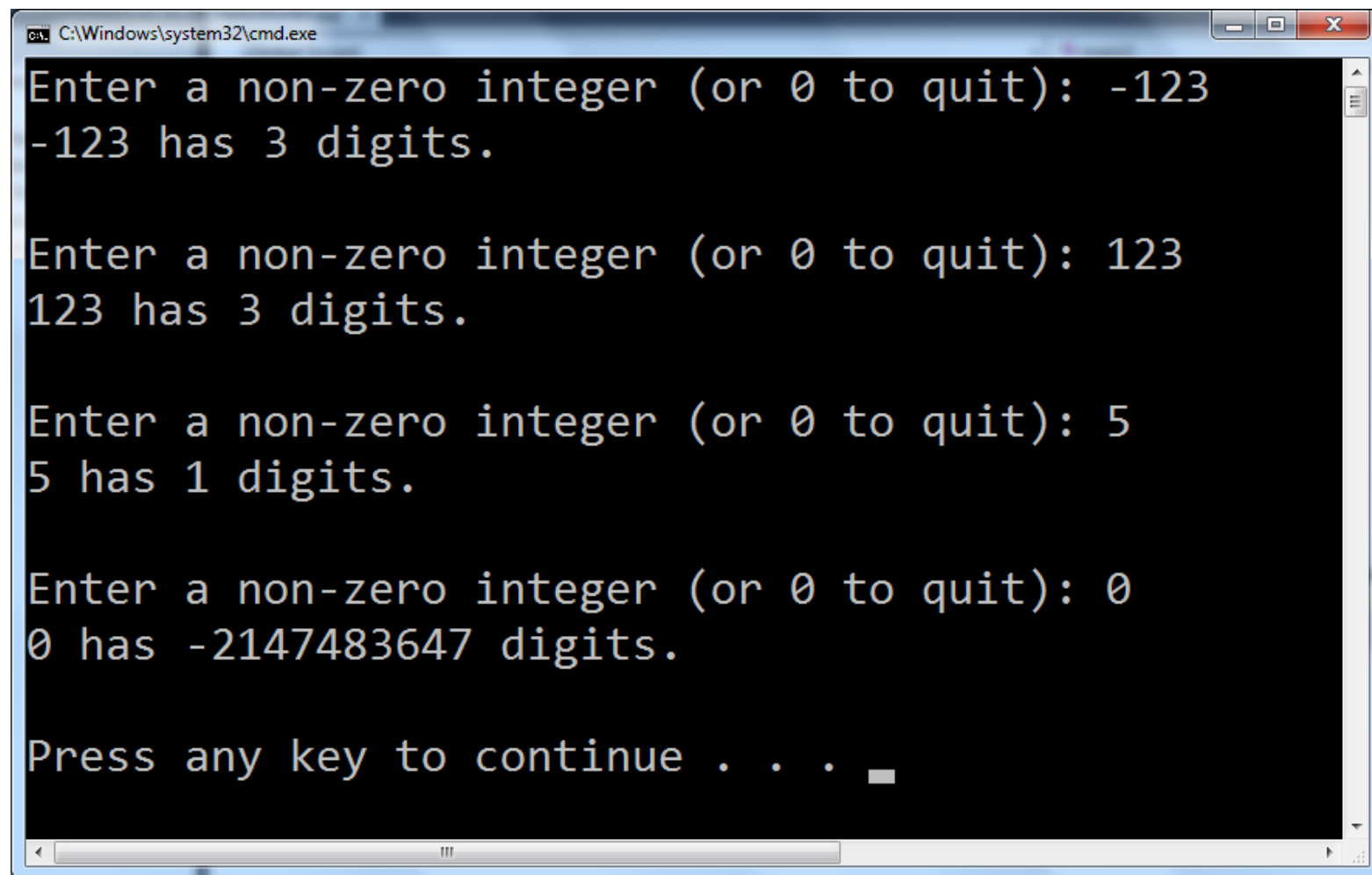
```
int main()
{
    int Value = 0;

    do {
        cout << "Enter a non-zero integer or 0 to quit: ";
        cin >> Value;

        cout << Value << " has "
             << (int)(log10((double)abs(Value)) + 1)
             << " digits.\n\n";

    } while (Value != 0);
}
```

do...while (пример)



```
C:\Windows\system32\cmd.exe
Enter a non-zero integer (or 0 to quit): -123
-123 has 3 digits.

Enter a non-zero integer (or 0 to quit): 123
123 has 3 digits.

Enter a non-zero integer (or 0 to quit): 5
5 has 1 digits.

Enter a non-zero integer (or 0 to quit): 0
0 has -2147483647 digits.

Press any key to continue . . .
```


Оператори за цикъл ...

Оператор *for*

- *Синтаксис*

for (<инициализация>; <условие>; <корекция>)
 <оператор>

където

- for (за) е запазена дума.
- <инициализация> е или **точно една** дефиниция с инициализация на една или повече променливи, или един или няколко оператора, **отделени със , и не завършващи с ;**.
- <условие> е булев израз.
- <корекция> е един или няколко оператора, **незавършващи с ;**. В случай, че са няколко, отделят се със ,.
- <оператор> е точно един произволен оператор. Нарича се **тяло на цикъла**.

Оператори за цикъл ...

Оператор *for*

- *Семантика*

Изпълнението започва с изпълнение на частта <инициализация>. След това се намира стойността на <условие>. Ако в резултат се е получило false, изпълнението на оператора for завършва, без тялото да се е изпълнило нито веднъж. В противен случай последователно се повтарят следните действия:

- Изпълнение на тялото на цикъла;
- Изпълнение на операторите от частта <корекция>;
- Пресмятане стойността на <условие>

докато стойността на <условие> е true.

Оператори за цикъл ...

Оператор *for*

- *Забележки:*
 1. Тялото на оператора `for` е точно един оператор. Ако повече оператори трябва да се използват, се оформя блок.
 2. Частта <инициализация> се изпълнява само веднъж – в началото на цикъла. Възможно е да се изнесе пред оператора `for` и остане празна.

```
int fact = 1;  
int i = 1  
for ( ; i <= n; i++)  
    fact = fact * i;
```

Оператори за цикъл ...

Оператор *for*

```
int fact;  
int i;  
for (fact = 1, i = 1; i <= n; i++)  
    fact = fact * i;
```

- Частта <корекция> се нарича така, тъй като обикновено чрез нея се модифицират стойностите на променливите, инициализирани в частта <инициализация>. Тя може да се премести в тялото на оператора for като се оформи блок от вида {<оператор> <корекция>;

Цикъл for

```
int main()
{
    int Counter = 0;

    while (Counter < 10)
    {
        cout << Counter << " ";
        Counter++;
    }

    cout << endl;
}
```

```
int main()
{
    for (int Counter = 0; Counter < 10; Counter++)
        cout << Counter << endl;
}
```

Няколко бележки относно циклите

- Няма нещо, което да можете да направите с `for` и да не можете да направите с `while` и обратното.
- Обикновено избираме между двете възможности така, че кодът да бъде оформен по-добре.
- Когато използвате цикли, обикновено има различни начини да напишете един и същ итеративен процес, като пренасяте код между клаузите на цикъла и/или тялото му.
 - В това отношение, цикълът `for` е най-гъвкав
 - На следващия слайд е даден пример как може да се направи едно и също нещо по няколко начина

```
for (int i = 0; i <= 10; i++)
    cout << i << endl;
//-----
int i = 0;
for (; i <= 10; i++)
    cout << i << endl;
//-----
int i = 0;
for (; i <= 10;)
    cout << i++ << endl;
//-----
int i = 0;
for (;;)
{
    if (i > 10) break;
    cout << i++ << endl;
}
//-----
for (int i = 0; i <= 10; cout << i << endl, i++);
```

Безкраен цикъл

- Ако в който и да е оператор за цикъл, в <условие> има израз, който никога не приема стойност false, то цикълът никога няма да завърши

```
int Counter = 0;  
for (;;)Counter++)  
    cout << Counter << endl;
```


Безкраен цикъл

```
int Counter = 0;
```

```
while (Counter < 10)  
    cout << Counter << endl;
```

```
int Counter = 0;  
for (;;)   
    cout << Counter << endl;
```

Оператори за цикъл ...

Оператор *for*

- Ако частта <условие> е празна, подразбира се true. За да се избегне зацикляне, от тялото на цикъла при определени условия трябва да се излезе принудително, например чрез оператора break.

```
for (int i = 1; ; i++)  
    if (i > n)  
        break;  
    else  
        fact = fact * i;
```

- Област на променливите, дефинирани в заглавната част на for

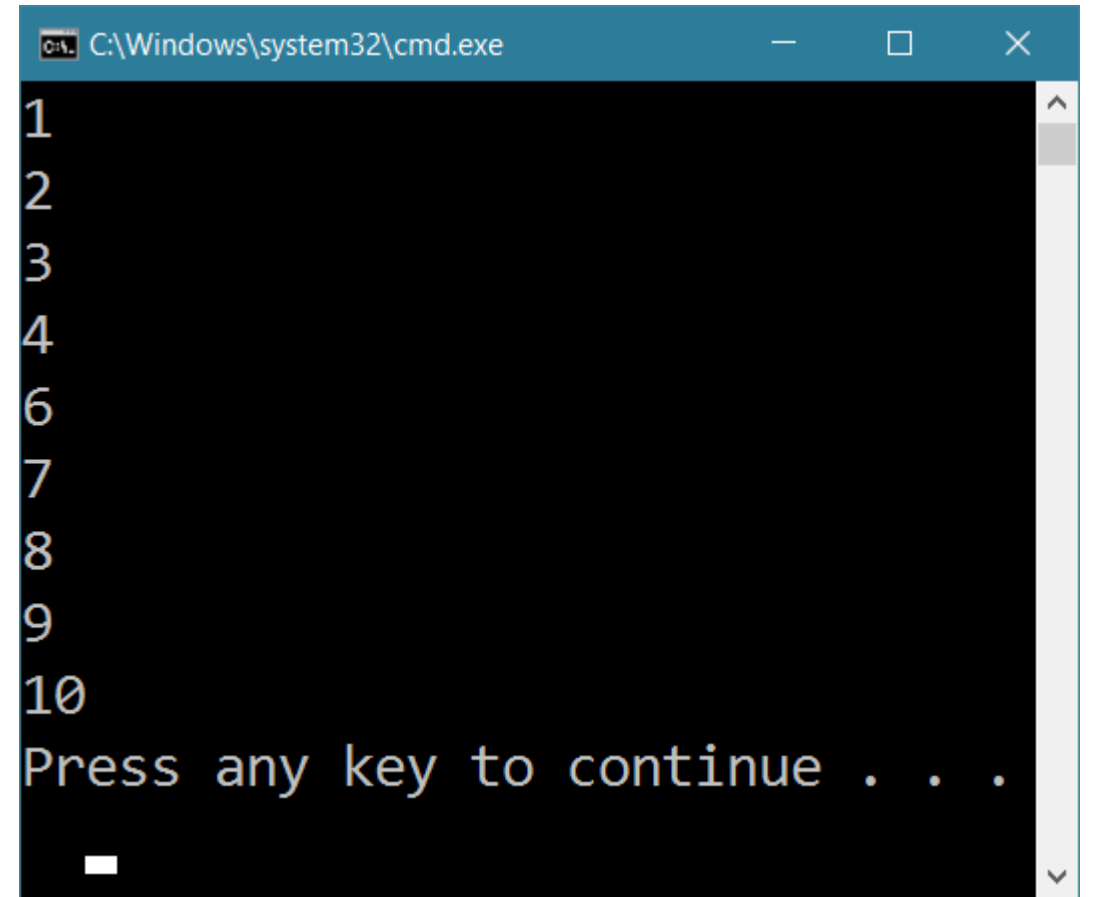
Jump statements (инструкции за преход)

Оператор continue

```
int i = 0;

while (++i <= 10)
{
    if (i == 5)
        continue;

    cout << i << endl;
}
```



```
C:\Windows\system32\cmd.exe

1
2
3
4
6
7
8
9
10
Press any key to continue . . .
```

Оператор continue

Ако срещнем continue в **цикъл while**:

1. Оценява се <условие>
2. Ако <условие> е истина, изпълняваме следващата итерация на цикъла.

Ако срещнем continue в **цикъл for**:

1. Изпълнява се <корекция>
2. Оценява се <условие>
3. Ако <условие> е истина, изпълняваме следващата итерация на цикъла.

// Когато използвате continue, **много внимавайте** да не
// предизвиквате безкраен цикъл

```
int i = 0;
```

```
while (i <= 10)  
{
```

```
    if (i == 5)  
        continue;
```

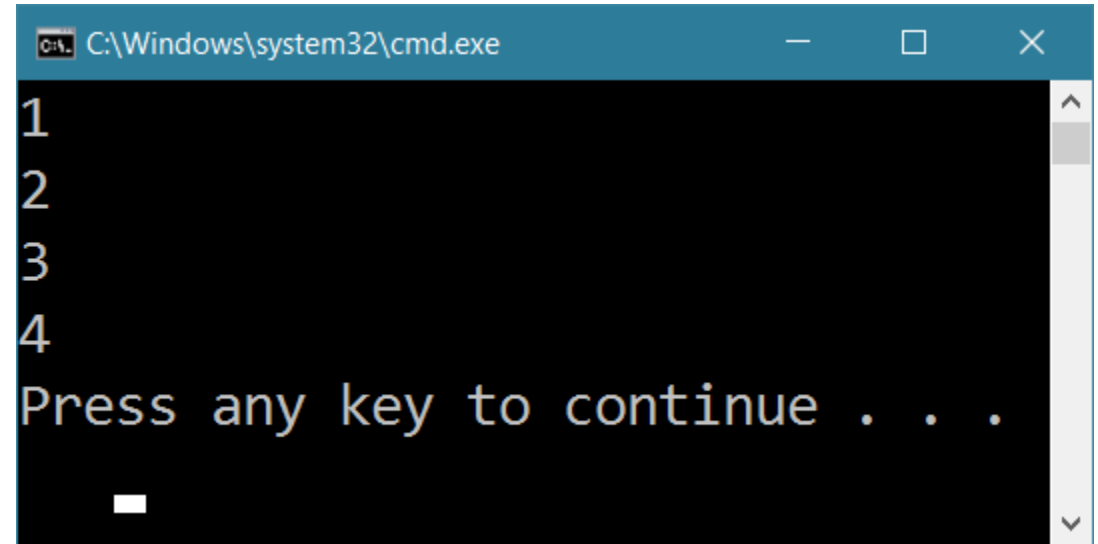
```
    cout << i++ << endl;  
}
```

Оператор break

```
int i = 0;

while (++i <= 10)
{
    if (i == 5)
        break;

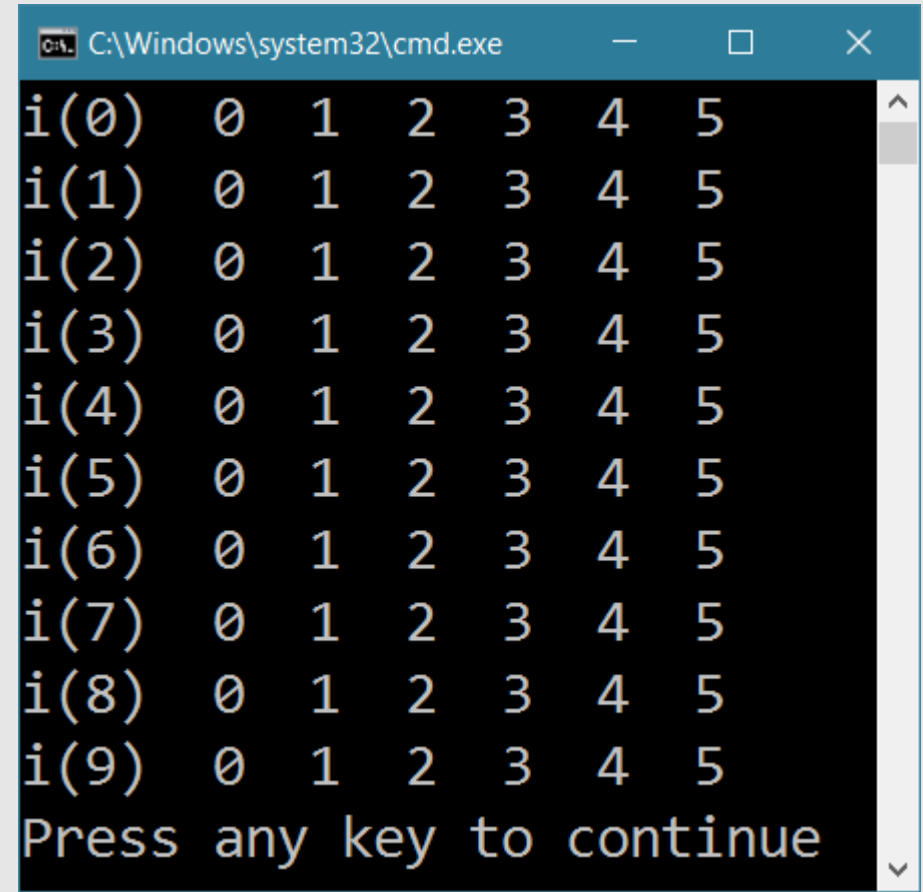
    cout << i << endl;
}
```



A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The window has a black background with white text. The output of the program is visible: the numbers 1, 2, 3, and 4 are printed on separate lines. Below these numbers, the text "Press any key to continue . . ." is displayed. A small white cursor is visible on the line below the prompt text.

// При вложени цикли, break и continue влияят само
// върху цикъла, в който се намират

```
for (int i = 0; i < 10; i++)  
{  
    cout << "i(" << i << ")";  
  
    for (int j = 0; j < 10; j++)  
    {  
        if (j > 5)  
            break;  
        cout << setw(3) << j);  
    }  
  
    cout << endl;  
}
```



```
C:\Windows\system32\cmd.exe  
i(0)  0  1  2  3  4  5  
i(1)  0  1  2  3  4  5  
i(2)  0  1  2  3  4  5  
i(3)  0  1  2  3  4  5  
i(4)  0  1  2  3  4  5  
i(5)  0  1  2  3  4  5  
i(6)  0  1  2  3  4  5  
i(7)  0  1  2  3  4  5  
i(8)  0  1  2  3  4  5  
i(9)  0  1  2  3  4  5  
Press any key to continue
```

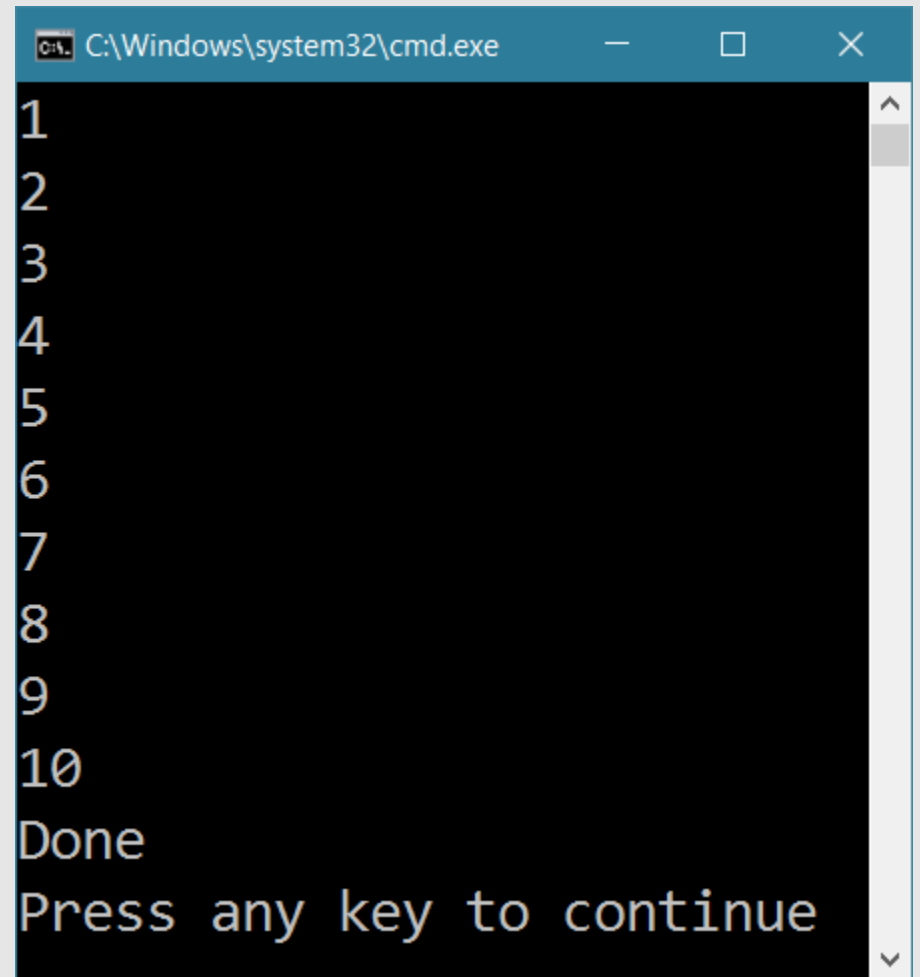

Оператор goto

```
int main()
{
    int Counter = 1;

LOOP_START:
    cout << Counter << endl;

    if (Counter < 10)
    {
        Counter++;
        goto LOOP_START;
    }

    cout << "Done\n";
}
```

A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The window has a black background with white text. The output of the program is displayed as a list of numbers from 1 to 10, each on a new line. Below the numbers, the word "Done" is printed. At the bottom of the window, the text "Press any key to continue" is shown, indicating that the program has finished execution and is waiting for a key press to close the window.

```
C:\Windows\system32\cmd.exe
1
2
3
4
5
6
7
8
9
10
Done
Press any key to continue
```

- За подготовката на тази презентация са използвани слайдове на:
 - Доц. Александър Григоров
 - Доц. Атанас Семерджиев
 - Доц. Трифон Трифонов