Универзитет „Св. Кирил и Методиј“ во Скопје
ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Контролни структури за повторување

Структурно програмирање

ФИНКИ 2024

Содржина

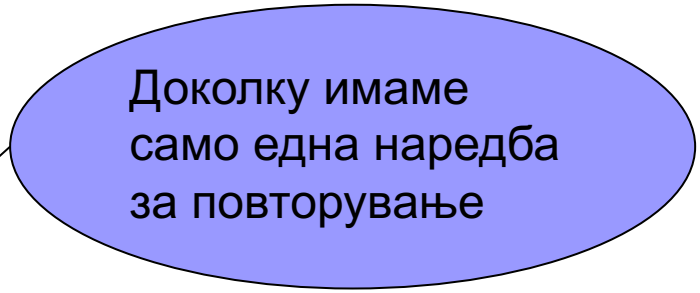
- Циклуси
  - While
  - Do-while
  - For
- Наредбата continue
- Наредбата break
- Наредбата goto

Циклуси

- Циклусите се употребуваат за повторување групи наредби сè додека некој услов е исполнет:

1. while
2. do/while
3. for

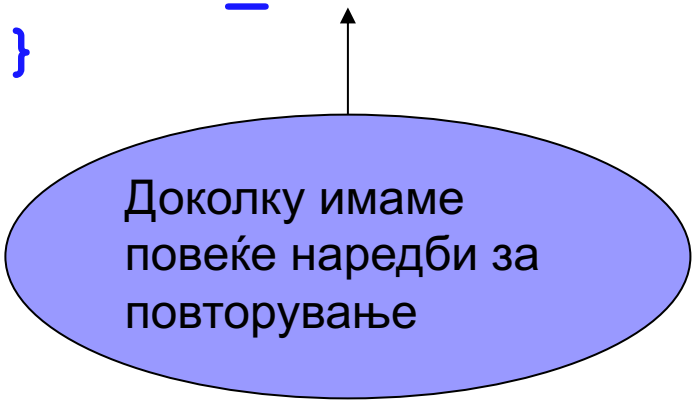
while




```
while (uslov)
  naredba;
```

или

```
while (uslov) {
  blok_naredbi;
}
```



Условот се испитува на почетокот (уште пред влезот во циклусот). Наредбите од циклусот се повторуваат:

- ниту еднаш (ако условот не е исполнет уште на самиот почеток)
- еднаш или повеќе пати

Пример:

```
while(godini < 18)
      rasti;
izvadi_licna_karta;
grizi_se_za_sebe;
```

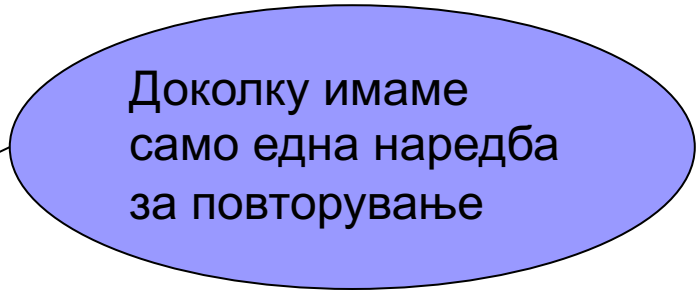
Што печати следнава програма?

```
#include <iostream>
using namespace std;
int main()
{
    int n = 1;
    int broj, suma = 0;
    while (n <= 5)
    {
        cout << "Vnesi broj: ";
        cin >> broj;
        suma += broj;
        n++;
    }
    cout << "\nSredna vrednost na vnesenite broevi e "
        << (float)suma / (n - 1) << endl;
    return 0;
}
```

```
Vnesi broj: 3
Vnesi broj: 5
Vnesi broj: 1
Vnesi broj: 3
Vnesi broj: 2

Sredna vrednost na vnesenite broevi e 2.8
```

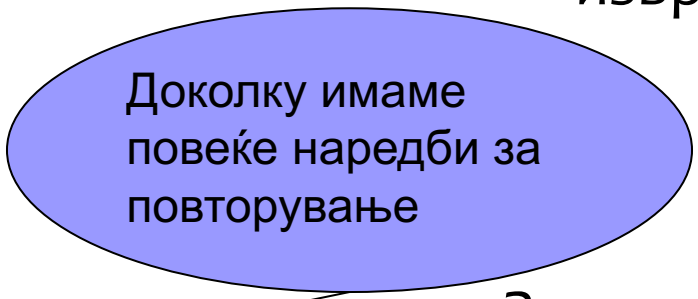
do - while




```
do
   naredba;
while (uslov);
```

Условот се испитува на крајот, поради што блокот на наредби се извршува **најмалку еднаш!**

или




```
do
{
   blok_naredbi;
}
while(uslov);
```

Задача: Внесувај цели броеви од тастатура се додека не се внесе 0. Пресметај ја сумата на внесените броеви.

Пример:

```
#include <iostream>
using namespace std;
int main() {
   int broj, suma = 0;
   do {
      cin >> broj;
      suma+=broj;
   } while(broj!=0);
   cout << "Suma = “ << suma << endl;
   return 0;
}
```

for циклуси

for циклусот во C++ се дефинира во три дела на следниот начин:

```
for( inicijalizacija ; uslovi ; inkrementi_ili_dekrementi )
   naredba;
```

или

```
for( inicijalizacija ; uslovi ; inkrementi_ili_dekrementi )
{
   blok_naredbi;
}
```

Иницијализација

Во делот за иницијализација вообичаено се доделуваат почетните вредности на бројачите.

```
for(x = 1; ...
```

Ако има повеќе бројачи нивните иницијализации се одделуваат со запирки.

```
for(x = 1, a = 0, z = start; ...
```

Условите се логички и релациски изрази со кои се поставуваат услови што ќе го контролираат извршувањето на циклусот.

Услов/инкремент, декремент

Сè додека условот е исполнет се инкрементираат или декрементираат бројачите и се повторуваат наредбите од циклусот.

```
for (j = 0; j < 30000; j++)
```

Во третиот дел обично се инкрементираат или декрементираат една или повеќе променливи, но може да се стави и која и да е друга наредба.

```
for (x = 0, j = 0; j < 100; j++, x+=5)
```

Извршување на for наредба

- Во секој од овие делови може да се стават произволни наредби, но редоследот на нивното извршување и интерпретацијата на нивните резултати е точно одреден:
- Наредбите од делот ***inicijalizacija*** се извршуваат точно еднаш, **на почетокот - пред влезот во циклусот;**

Извршување на for наредба

- Наредбите од делот ***uslovi*** се извршуваат пред почетокот на секој нов циклус и ако резултираат со вредност која се интерпретира како логичка вистина се повторуваат наредбите од циклусот, инаку се завршува повторувањето на циклусот и се продолжува со наредбите по циклусот;

Извршување на for наредба

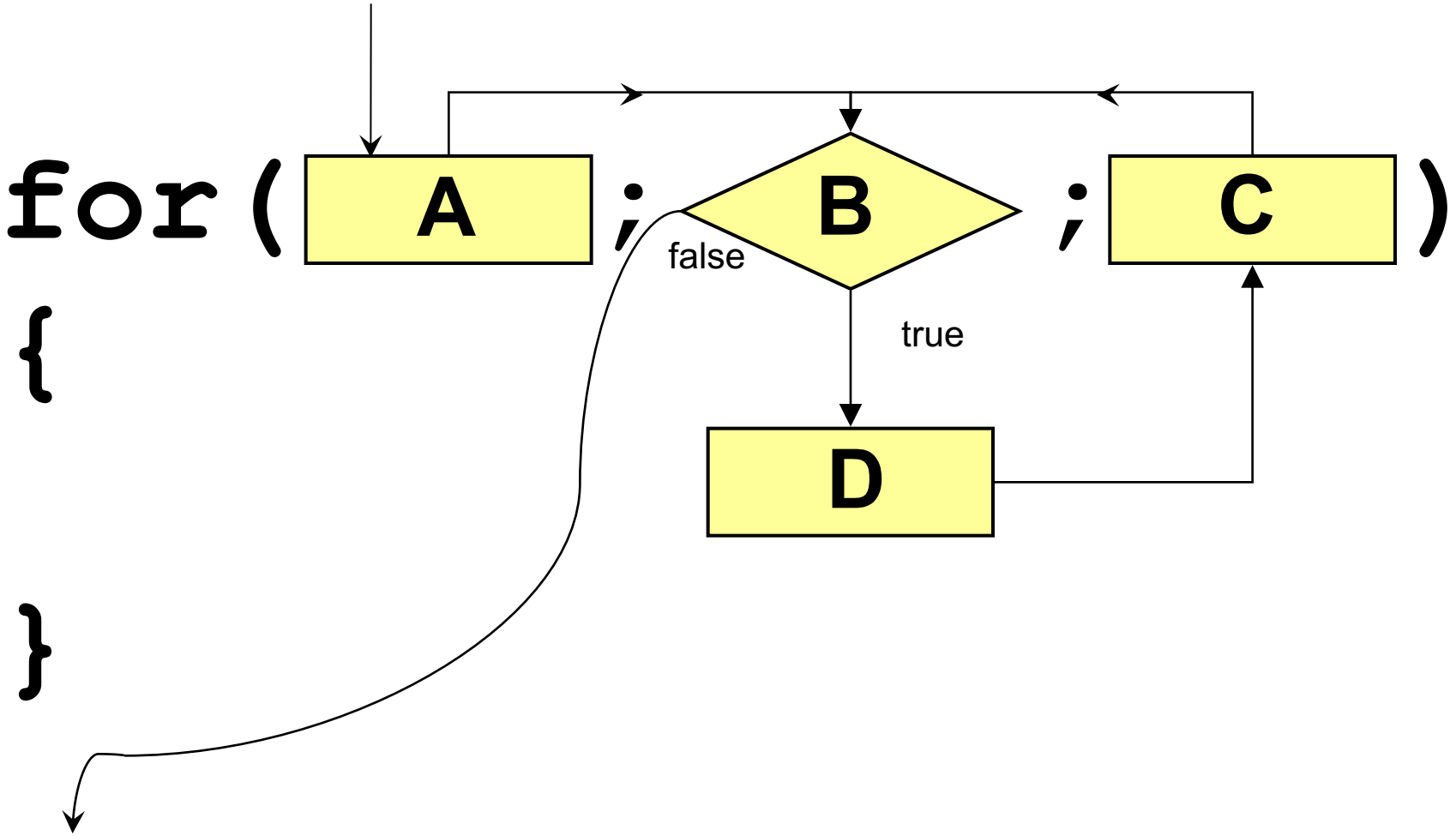
- Наредбите од делот ***inkrementi_ili_dekrementi*** се извршуваат на **крајот на секој циклус** (по извршувањето на сите наредби од телото на циклусот ***blok_naredbi***) по што се извршуваат наредбите од делот ***uslovi*** и ако се задоволени, циклусот се повторува.

Пример за for

Да се пресмета сумата на непарните позитивни броеви помали од 20.

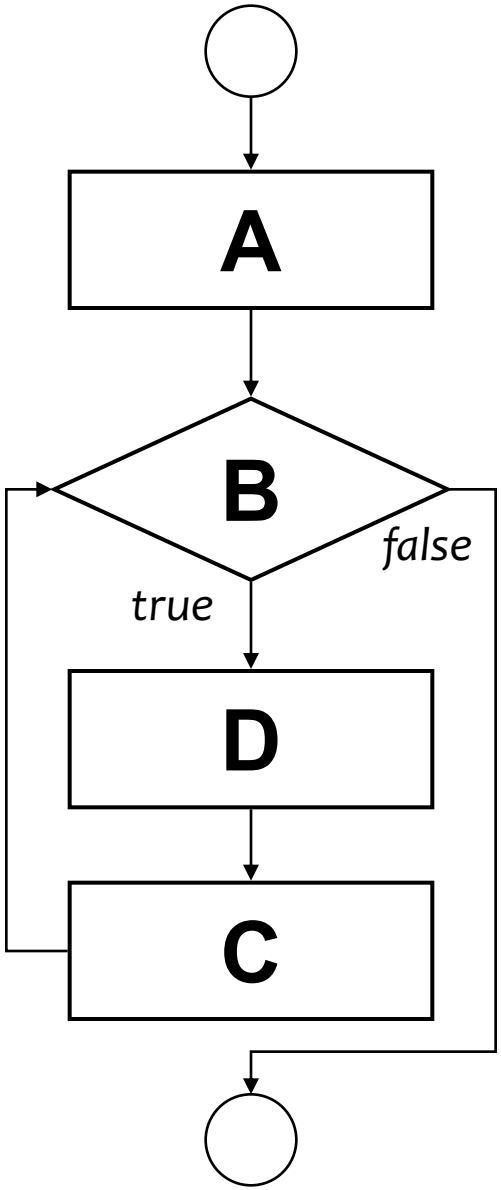
```
#include <iostream>
using namespace std;
int main() {
   int i, suma = 0;
   for(i=1; i<20; i+=2) {
      suma+=i;
   }
   cout << "Suma = “ << suma << endl;
   return 0;
}
```

Редослед на извршување на деловите од for наредбата



Пример

```
. . .
for(A;B;C)   D;
. . .
```




```
#include <iostream>
using namespace std;
int main()
{
   int i = 0;
   cout << "Ke pocne ciklus...\n";
   for (cout << "A";
      cout << "B", i<3;
      cout << "C", i++)
      cout << "D";
      cout << "\nCiklusot zavrsi.\n";
   return 0;
}
```

```
Ke pocne ciklus...
ABDCBDCBDCB
Ciklusot zavrsi.
```

Пример

```
#include <iostream>
using namespace std;
int main()
{
int suma, x, y;
suma = 0;
y = 5;
for (x = 1; x < y; x++);
    suma = suma + x * y;
cout << "Sumata e " << suma << endl;
return 0;
}
```

Кај **for** наредбата треба да се внимава на тоа дека таа **нема ;** по заградите.

```
Sumata e 25
```

Пример

Некои делови на for наредбата можат да бидат празни:

```
#include <iostream>
using namespace std;
int main()
{
   char c='.'; /* sto bilo razlicno od 'x' */
   cout << "Vnesuvaj znakovi:\n(Vnesi x za kraj)\n";
   for (  ; c != 'x'; )
   {
      cin >> c;
      cout << c;
   }
   cout << "\nKraj na ciklusot!\n";
   return 0;
}
```

Каде е грешката во програмава?

Пример

Илустрација на употреба на влезно - излезна наредба во рамките на for наредба

```
#include <iostream>
using namespace std;
int main() {
    int broj = 0;
    for (cout << "Vnesuvaj broevi\n";
        broj != 6;
        cin >> broj);
    cout << "Toj broj go sakam!\n";
    return 0;
}
```

```
#include <iostream>
using namespace std;
int main()
{
 int j = 0;
 while (j++ < 3)
    cout << "Ha ";
 do {
       j -= 2;
       cout << "Hi ";
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
       cout << "Ho ";
  cout << endl;
  return 0;
}
```

```
j = 3
j = 1
j = 2
j = 0
j = 1
j = -1
j = 0
```

```
Ha  Ha  Ha  Hi  Hi  Hi  Hi  Ho  Ho  Ho
```

Наредба за излегување од циклус - break

Наредбата break овозможува излегување од циклус реализиран со `for`, `while`, `do-while` пред условот за напуштање на циклусот да биде исполнет.

```
#include <iostream>
using namespace std;
int main()
{
   int x = 0;
   for (;;++x)
   {
      if (x > 20000)
         break;
   }
   cout << x;
   return(0);
}
```

Наредба - continue

Наредбата **continue** не носи директно во следниот чекор на јамката без да бидат извршени наредбите до крајот на јамката.

```
#include <iostream>
using namespace std;
int main()
{
  char c;
  while ((c=cin.get()) != cin.eof())
  {
    if (c >= '0' && c <= '9') // if(isdigit(c))
      continue;
    cout << c;
  }
  return 0;
}
```

```
u65tf43d9i765z
utfdiz
```

Наредба - continue

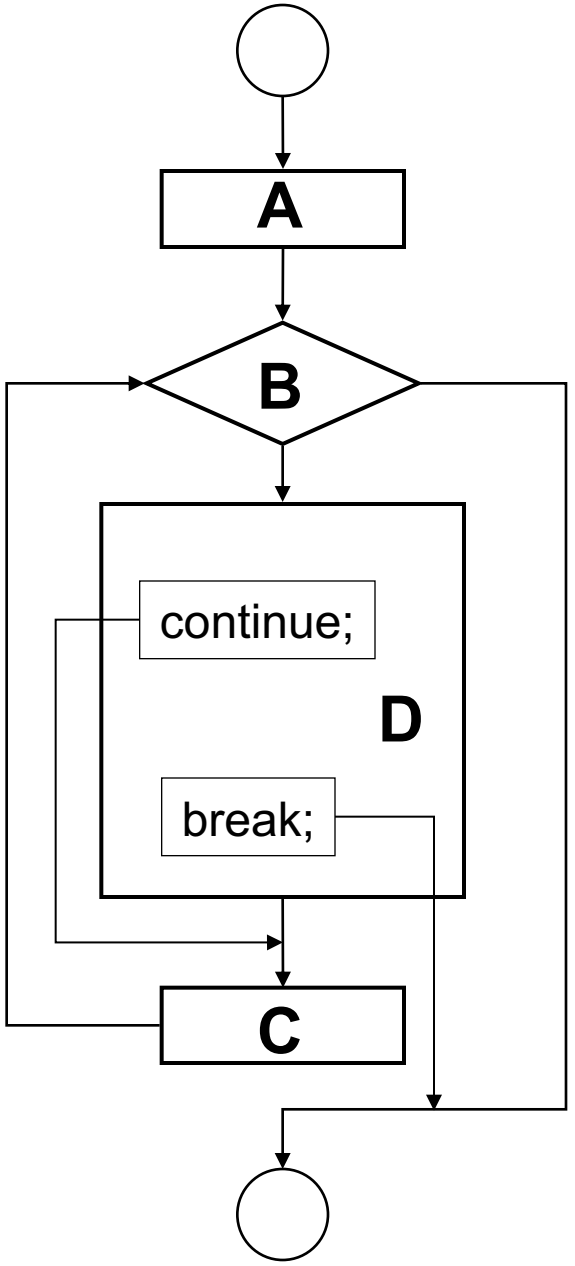
```
#include <iostream>
using namespace std;
int main()
{
   int i;
   for (i = 0; i<10; i++)
   {
      if (i<5) continue;
      cout << i << endl;
   }
   return 0;
}
```

```
5
6
7
8
9
```

continue и break

```
. . .
for(A ; B ; C)
{
   . . .
      break;
   . . .
      continue;
   . . .
}
. . .
```

D



Да се состави програма што ќе ги отпечати сите прости броеви помали од даден број.

```
#include <cmath>
#include <iostream>
using namespace std;
int main()
{
   int i, n, x = 0;
   cout << "Vnesi broj ";
   cin >> n;
   cout << "Prosti broevi pomali od " << n << " se:\n";
   for (i = 1; i < n; i+=2)
   {
      int j, k = 1;
      for (j = 2; j <= sqrt(i); j++)
         if (i%j == 0)
         {
            k = 0; break;
         }
      if (k)
      {
         cout << i << ' ';    x++;
      }
   }
   cout << "\n Vkupno " << x << " prosti broevi\n";
   return(0);
}
```

Напишете програма

Програма која ги печати фибоначевите броеви помали од 1000.

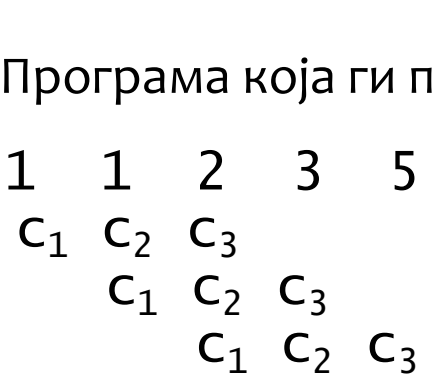
```
#include <iostream>
using namespace std;
int main()
{
   int c1, c2;
   const int n = 1000;
   for (cout << (c1 = c2 = 1); c2<n;
                              c1 = (c2 += c1) - c1)
        cout << ' ' << c2;
   cout << endl;
   return(0);
}
```

```
1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
```

Напишете програма

Програма која ги печати фибоначевите броеви помали од 1000.

1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987




```
#include <iostream>
using namespace std;
int main()
{
   int c1, c2, c3;
   const int n = 1000;

   c1 = c2 = 1;
   cout << c1;
   do
   {
      cout << c2;
      c3 = c2 + c1;
      c1 = c2;
      c2 = c3;
   } while (c2 < n);
   cout << endl;
   return(0);
}
```

Напишете програма

Пример програма со повеќе бројачи:

```
#include <iostream>
using namespace std;

int main()
{
   int i, j;
   for (i = 0, j = 8; i<8; i++, j--)
      cout << i << " + " << j << " = "
         << i + j <<  endl;
   return 0;
}
```

```
0 + 8 = 8
1 + 7 = 8
2 + 6 = 8
3 + 5 = 8
4 + 4 = 8
5 + 3 = 8
6 + 2 = 8
7 + 1 = 8
```

Наредба goto

Основниот облик на оваа наредба е следниот

```
goto (ime_na_oznaka);
```

со тоа што некаде во програмата го имаме името на ознаката во облик

```
ime_na_oznaka:
```

Лош стил на програмирање!
НЕ УПОТРЕБУВАЈТЕ goto !!!

Контролата на програмата по извршувањето на оваа наредба се пренесува на наредбата со ознаката.

Прашања?