ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Контрола на тек

Структурно програмирање

ФИНКИ 2014

Содржина

- **Вовед**
- If-else
- Циклуси
- While
- Do-while
- For
- Наредбата continue
- Наредбата break
- Switch

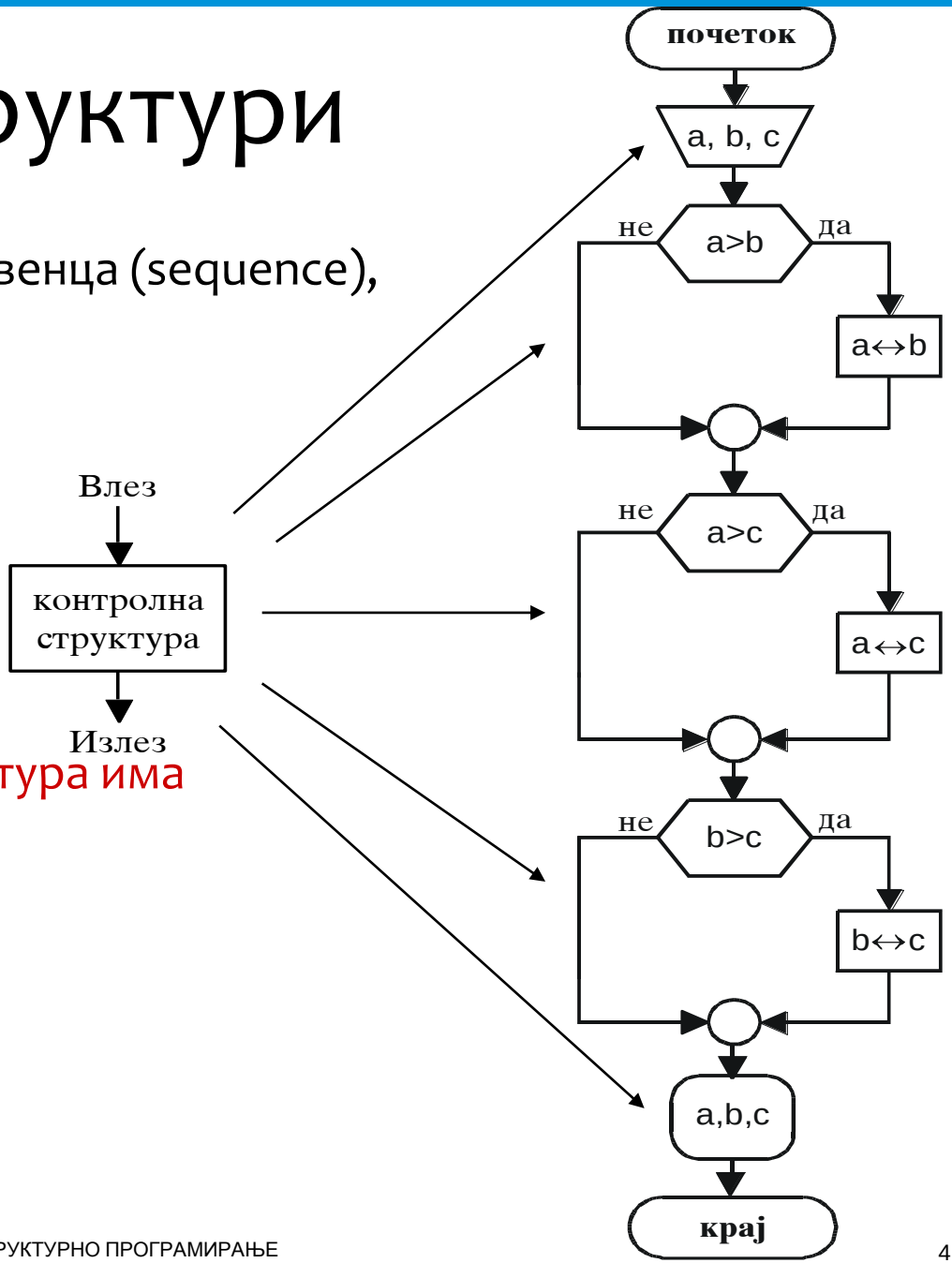
Контролни структури

- Секвенцијално (последователно) извршување
  - Чекорите (инструкциите, наредбите) се извршуваат една по друга во испишаниот редослед
- Пренос на контрола
  - Кога следната наредба што се извршува НЕ е следна во редоследот
- Bohm и Jacopini (1966)
  - Математички докажано дека сите програми може да се напишат со помош на 3 контролни структури
    - Редоследна структури: програмите по дефиниција се извршуваат секвенцијално
    - Изборна структура: ако-тогаш (if), ако-тогаш-инаку (if/else), и случај (switch)
    - Структури за повторување (циклуси): while, do/while и for

Контролни структури

1.Редоследна структура или секвенца (sequence),

2. Структура избор
селекција (selection),
3. структура повторување
итерација (iteration)



Секоја контролна структура има
ЕДНА влезна точка и
ЕДНА излезна точка

Се нарекуваат и:
- линиска структура
- разгранета структура
- циклична структура

За контролните структури

- Единствен влез/единствен излез кај контролните структури
  - Излезната точка од една контролна структура е поврзана со влезната точка од следната контролна структура
  - Програмите лесно се градат на овој начин

Редоследна структура блок од наредби

- Редоследна структура, секвенца, блок-наредби, линиска структура
- Претставува низа наредби што се извршуваат една по друга
- Се одделува од остатокот од програмата со { … }
- Пример:

```
{
    int a=2, t, b=3;
    t=a;
    a=b;
    b=t;
}
```

Структура за избор

- Структура за избор, разгранета структура, селекција
- Овозможуваат да се избере извршување на една наредба (блок-наредби) од една, две или повеќе наредби:
  - ако-тогаш (if),
  - ако-тогаш-инаку (if/else), и
  - случај (switch)

if - else

Општиот облик на if наредбата е следниот

```
if (uslov)
    naredba_za_vistinit_uslov;
else
    naredba_za_nevistinit_uslov;
```

Ако има блокови наредби тогаш се означува почетокот и крајот на блокот

```
if (uslov)
    {
        blok_naredbi_za_vistinit_uslov;
    }
else
    {
        blok_naredbi_za_nevistinit_uslov;
    }
```

if - else

Делот else не мора да постои.

```
if (uslov)
    naredba_za_vistinit_uslov;
```

и

```
if (uslov)
    {
        blok_naredbi_za_vistinit_uslov;
    }
```

- Условот во заградата може да биде каков било аритметичко-логички израз.

- Пример:

```
#include <stdio.h>
int main()
{
	char c;
	printf( "Vnesi bukva: ");
	scanf("%c",&c);
	if( c == 'a' || c == 'e' || c == 'i' ||
		c == 'o' || c == 'u' )
		printf( "Vnesena e samoglaska\n");
	printf("\n");
}
```

Вгнездување на if-структура

```
if (sredstva > cena)
  kupi;
else
  if (imas_prijatel)
     pozajmi_pari;
  else
     najdi_rabota;
```

Вгнездување на if-структура

Што ќе се изврши за x<5? А што за x=11?

```
if (x > 5)
    if(x < 11)
        polozi;
else
    padna;
```

Вгнездување на if-структура

Што ќе се изврши за x<5? А што за x=11?

```
if (x > 5)
    if(x < 11)
        polozi;
else
    padna;
```

Како треба да гласи структурата за да биде логична?

Вгнездување на if-структура

Што ќе се изврши за x<5? А што за x=11?

```
if (x > 5)
   if(x < 11)
      polozi;
else
   padna;
```

Како треба да гласи структурата за да биде логична?

```
if (x > 5) {
     if(x < 11) polozi;
     }
else   padna;
```

Пишување на условот

Често се пишува

```
if(izraz)
```

наместо

```
if(izraz != 0)
```

или

`if(!izraz)` наместо `if(izraz == 0)`

Во изразите може да се најдат и наредби за доделување и/или инкрементирање/декрементирање

```
if((c=getchar())!='\n' && ++i<n) ...
```

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 или повеќе пати.

Пример:

```
while (parite < cena_na_kola)
        raboti_poveke;
    kupi_kola;
```

Што печати следнава програма?

```
#include <stdio.h>
int main()
{
    int n = 1;
    int broj, suma = 0;
    while( n <= 5 )
    {
        printf( "Vnesi broj: ");
        scanf("%d", &broj);
        suma += broj;
        n++;
    }
    printf("\nSredna vrednost na vnesenite broevi"
        " e %f\n", (float)suma / (n-1));
    return 0;
}
```

do - while

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

или

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

Слична на наредбата REPEAT - UNTIL со таа разлика што наредбите од циклусот се повторуваат сè додека условот **е** исполнет.

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

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;

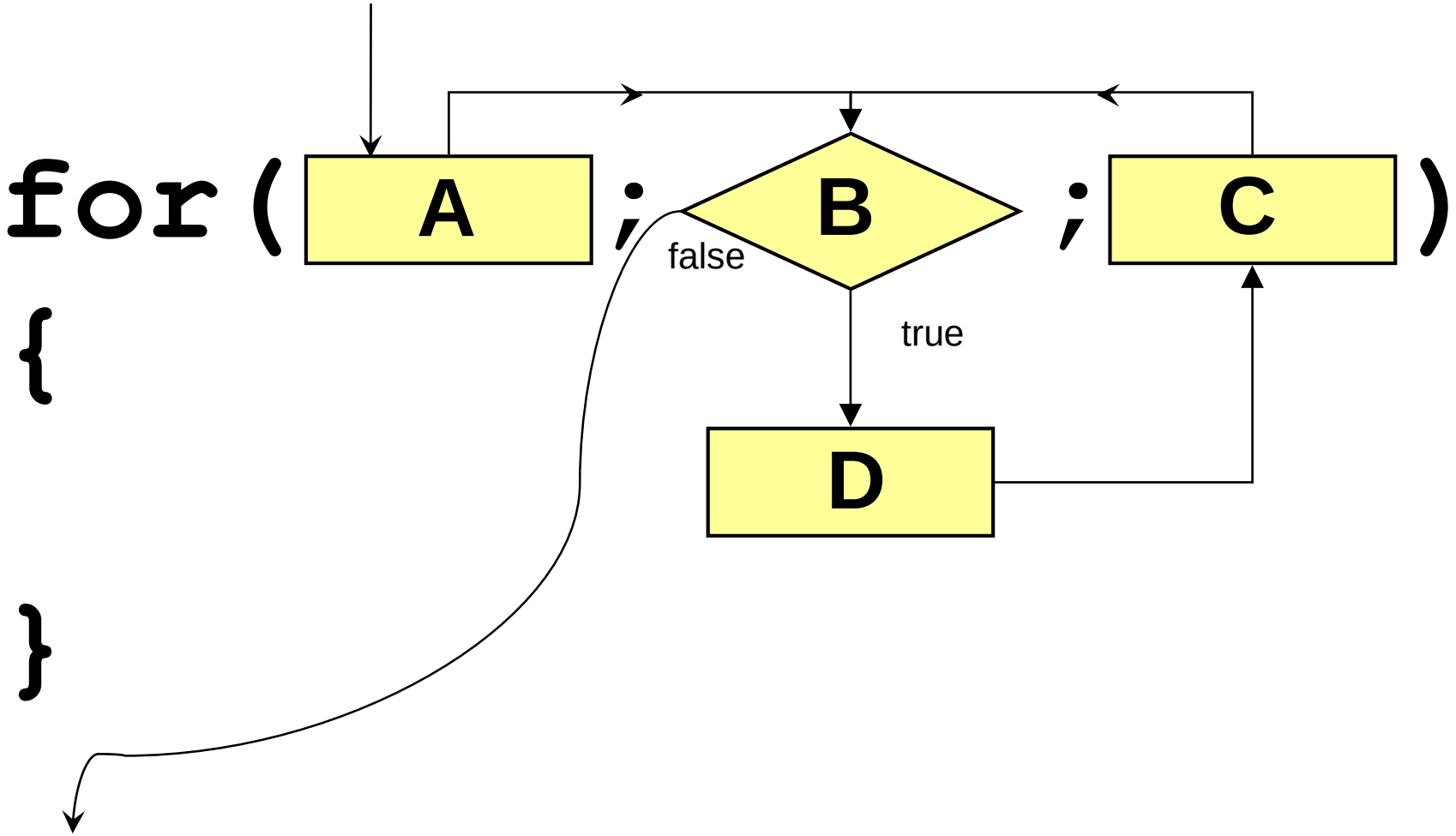
Извршување на for наредба

- 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.

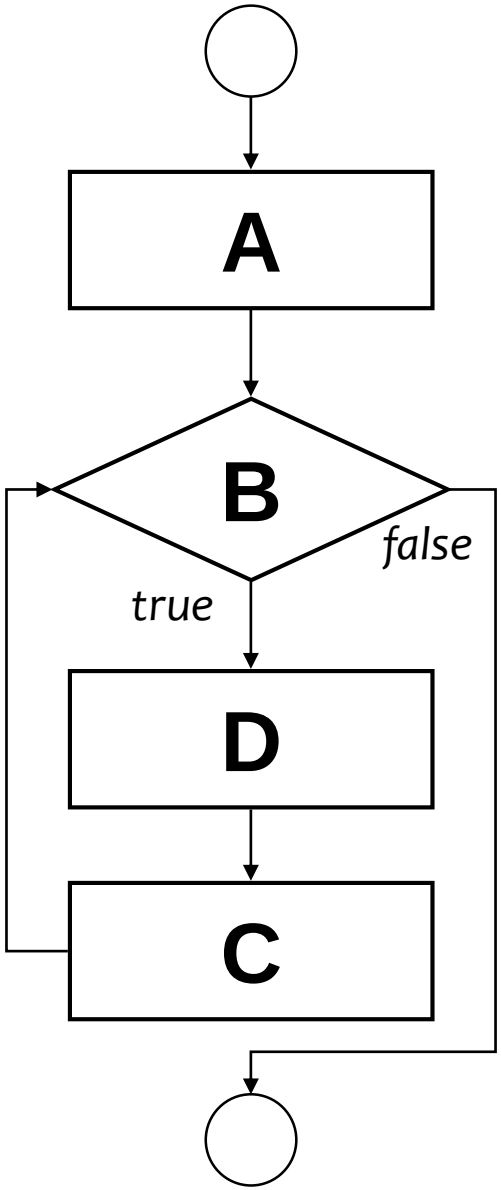
Редослед на извршување на деловите од for наредбата



Пример

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

```c
#include <stdio.h>
int main()
{
  int i=0;
  printf("Ke pocne ciklus...\n");
  for(printf("A");
    printf("B"),i<3;
    printf("C"),i++)
      printf("D");
  printf("\nCiklusot zavrsi.");
  return 0;
}
```

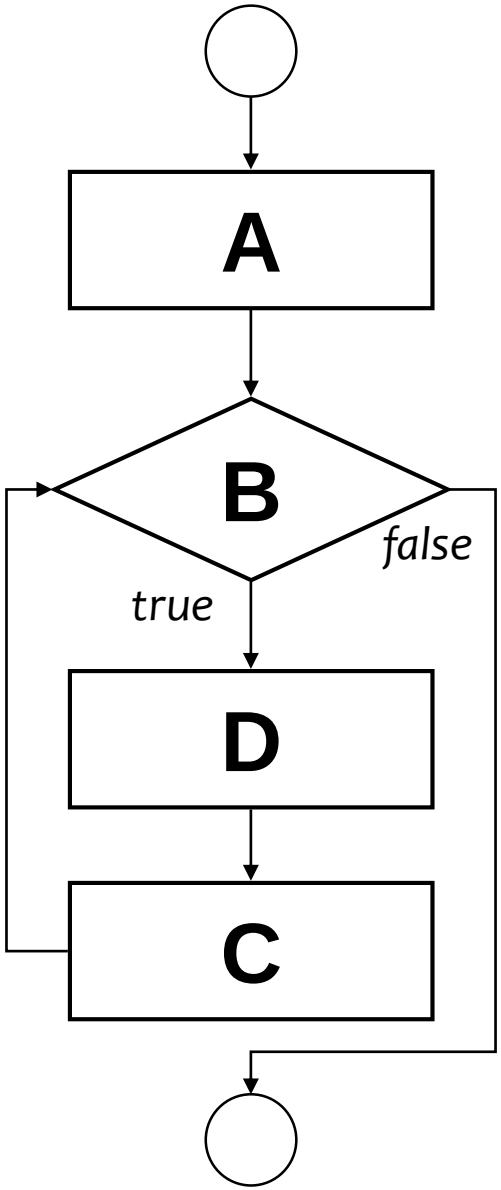


Пример

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

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

```
Ke pocne ciklus...
```

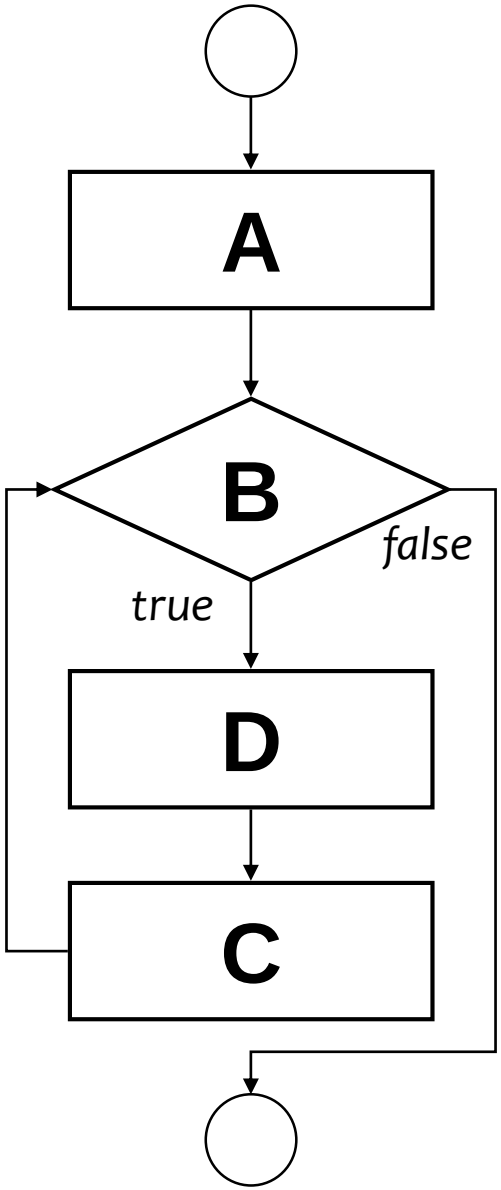


Пример

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

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

```
Ke pocne ciklus...
ABDCBDCBDCB
```

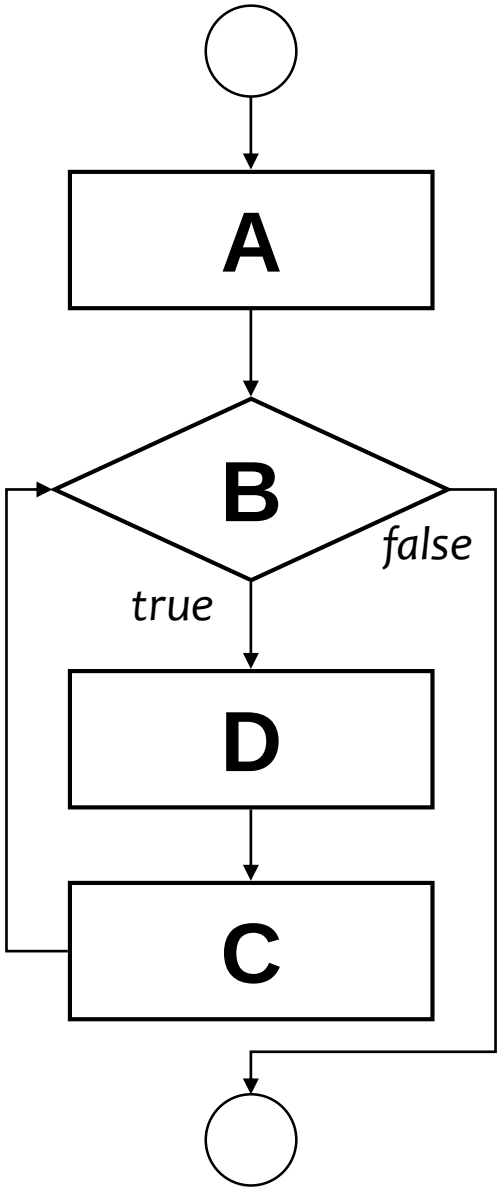


Пример

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

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

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



Пример

```
#include <stdio.h>
int main()
{
   int suma, x, y;
   suma = 0;
   y = 5;
   for (x = 1; x < y; x++);
       suma = suma + x * y;
   printf(" Sumata e %d\n", suma);
   return 0;
}
```

Пример

```
#include <stdio.h>
int main()
{
   int suma, x, y;
   suma = 0;
   y = 5;
   for (x = 1; x < y; x++);
       suma = suma + x * y;
   printf(" Sumata e %d\n", suma);
   return 0;
}
```

```
Sumata e 25
```

Пример

```
#include <stdio.h>
int main()
{
   int suma, x, y;
   suma = 0;
   y = 5;
   for (x = 1; x < y; x++);
       suma = suma + x * y;
   printf(" Sumata e %d\n", suma);
   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 <stdio.h>
int main()
{
	int c;
	printf("Vnesuvaj znakovi:\n(Vnesi x za kraj)\n");
	for (    ; c != 'x';    )
	{
		c = getchar();
		putchar(c);
	}
	printf("\nKraj na ciklusot!\n");
	return 0;
}
```

Пример

Некои делови на for наредбата можат да бидат празни:

```
#include <stdio.h>
int main()
{
	int c;
	printf("Vnesuvaj znakovi:\n(Vnesi x za kraj)\n");
	for (    ; c != 'x';    )
	{
		c = getchar();
		putchar(c);
	}
	printf("\nKraj na ciklusot!\n");
	return 0;
}
```

Каде е грешката во програмава?

Пример

Некои делови на for наредбата можат да бидат празни:

```
#include <stdio.h>
int main()
{
	int c; /* ne e inicijalizirana */
	printf("Vnesuvaj znakovi:\n(Vnesi x za kraj)\n");
	for (    ; c != 'x';    )
	{
		c = getchar();
		putchar(c);
	}
	printf("\nKraj na ciklusot!\n");
	return 0;
}
```

Каде е грешката во програмава?

Пример

Некои делови на for наредбата можат да бидат празни:

```
#include <stdio.h>
int main()
{
	int c='.'; /* sto bilo razlicno od 'x' */
	printf("Vnesuvaj znakovi:\n(Vnesi x za kraj)\n");
	for (    ; c != 'x';    )
	{
		c = getchar();
		putchar(c);
	}
	printf("\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 <stdio.h>
int main() {
    int broj=0;
    for(printf( "vnesuvaj broevi\n");
                broj != 6;
                scanf("%d", &broj));
    printf("Toj broj go sakam!\n");
    return 0;
}
```

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 0

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 0

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
    printf( "Ha ");
 do {
       j -= 2;
       printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
       printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 1

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 1

```
Ha
```

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 1

```
Ha
```

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
    printf( "Ha ");
 do {
       j -= 2;
       printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
       printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 2

Ha

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 2

```
Ha  Ha
```

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 2

```
Ha  Ha
```

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
    printf( "Ha ");
 do {
       j -= 2;
       printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
       printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 3

```
Ha  Ha
```

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 3

```
Ha  Ha  Ha
```

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 3

```
Ha  Ha  Ha
```

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 4

```
Ha  Ha  Ha
```

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 2

```
Ha  Ha  Ha
```

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 2

Ha Ha Ha Hi

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 3

```
Ha  Ha  Ha  Hi
```

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  return 0;
}
```

j = 3

```
Ha  Ha  Ha  Hi
```

```c
#include <stdio.h>
int main()
{
 int j = 0;
 while (j++ < 3)
   printf( "Ha ");
 do {
      j -= 2;
      printf( "Hi ");
 }
 while ( ++j );
 for(j = 1; j <= 3; j++)
      printf( "Ho ");
  printf("\n");
  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` или `switch` пред условот за напуштање на циклусот да биде исполнет.

```
#include <stdio.h>
int main()
{
  int x = 0;
  for (;;++x)
  {
    if(x > 20000)
    break;
  }
  printf("%d",x);
  return(0);
}
```

Наредба - continue

Наредбата **continue** не носи директно во следниот чекор на јамката запоставувајќи ги наредбите до крајот на јамката.

```
#include <stdio.h>
int main()
{
  char c;
  while ((c = getchar()) != EOF)
  {
    if(c >= '0' && c <= '9')
        continue;
    putchar(c);
  }
return 0;
}
```

Наредба - continue

Наредбата **continue** не носи директно во следниот чекор на јамката запоставувајќи ги наредбите до крајот на јамката.

```
#include <stdio.h>
int main()
{
  char c;
  while ((c = getchar()) != EOF)
  {
    if(c >= '0' && c <= '9')
        continue;
    putchar(c);
  }
return 0;
}
```

```
u65tf43d9i765z
utfdiz
```

Наредба - continue

```
#include <stdio.h>
int main()
{
   int i;
   for(i=0; i<10; i++)
   {
      if(i<5) continue;
      printf("%d\n", i);
   }
return 0;
}
```

Наредба - continue

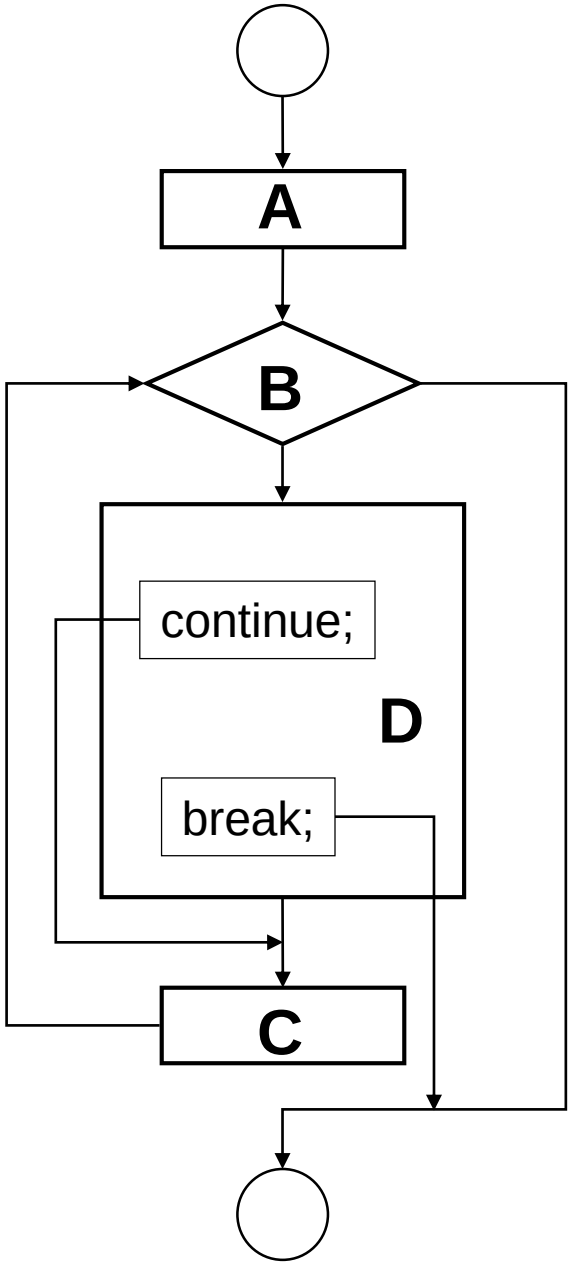
```
#include <stdio.h>
int main()
{
   int i;
   for(i=0; i<10; i++)
   {
      if(i<5) continue;
      printf("%d\n", i);
   }
return 0;
}
```

```
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<stdio.h>
int main()
{
  int i,j,k,n,x = 0;
  printf("Vnesi broj ");
  scanf("%d",&n);
  printf("Prosti broevi pomali od %d se\n",n);
  for(i = 1; i < n; i++)
  {
    k = 1;
    for(j = 2; j <= i/2; j++)
      if(i%j == 0)
      {
        k = 0;
      }
    if(k)
    {
      printf("%d ",i);   x++;
    }
  }
  printf("\n Vkupno %d prosti broevi",x);
  return(0);
}
```

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

```
#include<stdio.h>
int main()
{
  int i,j,k,n,x = 0;
  printf("Vnesi broj ");
  scanf("%d",&n);
  printf("Prosti broevi pomali od %d se\n",n);
  for(i = 1; i < n; i+=2)
  {
    k = 1;
    for(j = 2; j <= i/2; j++)
      if(i%j == 0)
      {
        k = 0;
      }
    if(k)
    {
      printf("%d ",i);   x++;
    }
  }
  printf("\n Vkupno %d prosti broevi",x);
  return(0);
}
```

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

```
#include<stdio.h>
int main()
{
  int i,j,k,n,x = 0;
  printf("Vnesi broj ");
  scanf("%d",&n);
  printf("Prosti broevi pomali od %d se\n",n);
  for(i = 1; i < n; i+=2)
  {
    k = 1;
    for(j = 2; j <= i/2; j++)
      if(i%j == 0)
      {
        k = 0; break;
      }
    if(k)
    {
      printf("%d ",i);   x++;
    }
  }
  printf("\n Vkupno %d prosti broevi",x);
  return(0);
}
```

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

```
#include<math.h>
#include<stdio.h>
int main()
{
  int i,j,k,n,x = 0;
  printf("Vnesi broj ");
  scanf("%d",&n);
  printf("Prosti broevi pomali od %d se\n",n);
  for(i = 1; i < n; i+=2)
  {
    k = 1;
    for(j = 2; j <= sqrt(i); j++)
      if(i%j == 0)
      {
        k = 0; break;
      }
    if(k)
    {
      printf("%d ",i);   x++;
    }
  }
  printf("\n Vkupno %d prosti broevi",x);
  return(0);
}
```

Избор од повеќе можности

switch - case

```
switch(izraz)
{
   case konstanta1: blok_naredbi1;
   case konstanta2: blok_naredbi2; break;
   . . .
   case konstantan: blok_naredbin;
   default:         naredbi;
}
```

izraz мора да резултира во **int** или **char вид**.

Избор од повеќе можности

`switch - case`

- Не смее да има два или повеќе case изрази со иста вредност.
- Програмата продолжува со наредбите зад case наредбата со вредноста на пресметаниот израз од switch наредбата.
- Се извршуваат следните наредби сè додека не се наиде на наредбата break или до крајот на switch-case блокот.
- Ако нема case наредба со соодветна вредност се извршуваат наредбите од default блокот.
- Ако не е наведен default блок на наредби, не се случува ништо - се продолжува со наредбите зад switch-case блокот.

Што печати следнава програма?

```
#include <stdio.h>
int main(){
  char c;
  c = getchar();
  switch (c) {
    case '1': printf("eden");    break;
    case '2': printf("dva");     break;
    case '3': printf("tri");     break;
    case '4': printf("cetiri");  break;
    case '5': printf("pet");     break;
    case '6': printf("sest");    break;
    case '7': printf("sedum");   break;
    case '8': printf("osum");    break;
    case '9': printf("devet");   break;
    case '0': printf("nula");    break;
    default:  printf("ne e cifra");
  }
  return(0);
}
```

Што ќе отпечати следнава програма?

```
#include<stdio.h>
int main() {
    int j = 0;
    while ( j < 6 )
    {
       switch ( j )
       {
          case 0: j++;
          case 1: j++;    break;
          case 2:
          case 3: j += 2; break;
          default: j = j - 1;
        }
        printf("Vrednosta na  j  e  %d\n", j++);
    }
    return(0);
}
```

Што ќе отпечати следнава програма?

```
#include<stdio.h>
int main() {
    int j = 0;
    while ( j < 6 )
    {
       switch ( j )
       {
          case 0: j++;
          case 1: j++;    break;
          case 2:
          case 3: j += 2; break;
          default: j = j - 1;
        }
        printf("Vrednosta na  j  e  %d\n", j++);
    }
    return(0);
}
```

```
Vrednosta na  j  e  2
Vrednosta na  j  e  5
```

Што ќе отпечати следнава програма?

```
#include<stdio.h>
int main() {
    int j = 0;
    while ( j < 6 )
    {
       switch ( j )
       {
          case 0: j++;
          case 1: j++;    break;
          case 2:
          case 3: j += 2; break;
          default: j = j - 1;
        }
        printf("Vrednosta na  j  e  %d\n", j++);
    }
    return(0);
}
```

```
Vrednosta na  j  e  2
Vrednosta na  j  e  5
```

како ќе работи програмата без ова ++?

Напишете програма...

... која пресметува вредност на едноставен аритметички израз (без приоритети, само цели броеви):

```
#include <stdio.h>
int main(){
  char operator = '+';
  int broj, resenie = 0;
  do
  {
     scanf("%d",&broj);
     switch(operator){
       case '+': resenie += broj; break;
       case '-': resenie -= broj; break;
       case '*': resenie *= broj; break;
       case '/': resenie /= broj; break;
     }
  }
  while((operator = getchar()) != '=');
  printf("Resenieto e %d",resenie);
  return 0;
}
```

```
15+2*2-4/3+1=
Resenieto e 11
```

Напишете програма

Пример програма со повеќе броила:

```
#include <stdio.h>
int main()
{
   int i, j;
   for (i=0, j=8; i<8; i++, j--)
      printf("%d + %d = %d\n", i,j,i+j);
   return 0;
}
```

Напишете програма

Пример програма со повеќе броила:

```
#include <stdio.h>
int main()
{
   int i, j;
   for (i=0, j=8; i<8; i++, j--)
      printf("%d + %d = %d\n", i,j,
   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
```

Напишете програма

Програма која ги печати фибоначевите броеви помали од 1000.

```
#include <stdio.h>
int main()
{
    int c1, c2;
    const int n = 1000;
    for(printf("%d ",(c1=c2=1)); c2<n;
            c1=(c2+=c1)-c1)
       printf("%d ",c2);
    printf("\n");
    return(0);
}
```

Напишете програма

Програма која ги печати фибоначевите броеви помали од 1000.

```
#include <stdio.h>
int main()
{
    int c1, c2;
    const int n = 1000;
    for(printf("%d ",(c1=c2=1)); c2<n;
            c1=(c2+=c1)-c1)
       printf("%d ",c2);
    printf("\n");
    return(0);
}
```

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

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:
```

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

Прашања?