

Управляващи оператори в C++

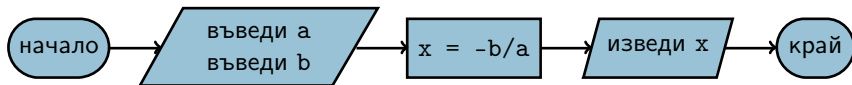
Трифон Трифонов

Увод в програмирането,
спец. Компютърни науки, 1 поток, 2018/19 г.

18 октомври 2018 г.

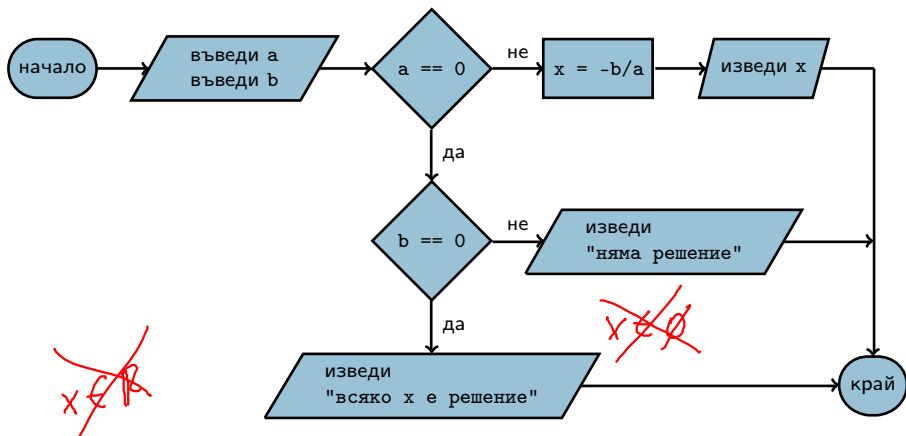
Изчислителни процеси

- Алгоритъм: последователност от стъпки за извършване на пресмятане
- Блок-схема

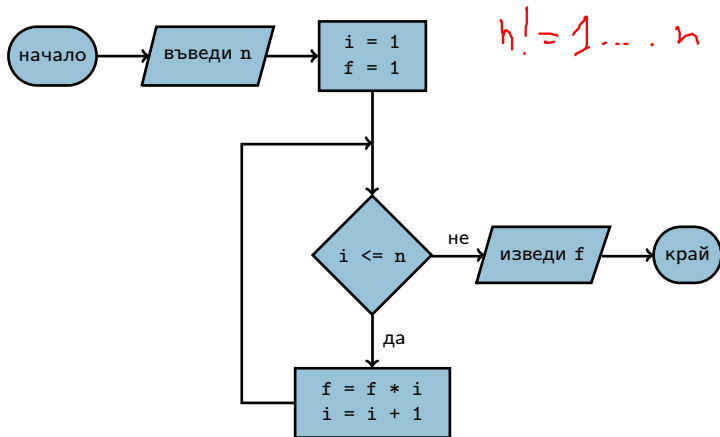


Пример за линеен процес

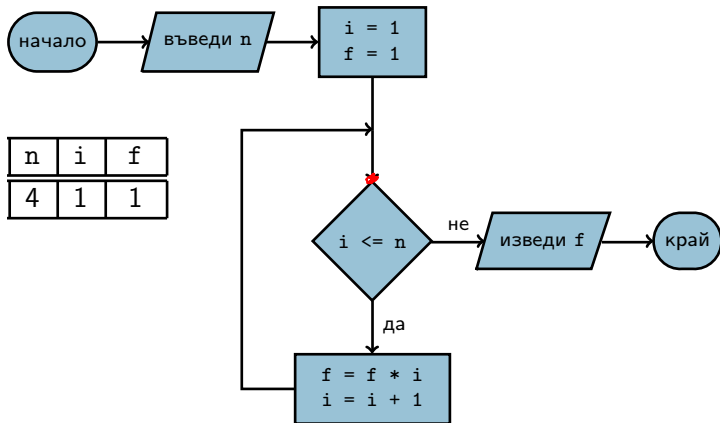
Разклоняващи се процеси



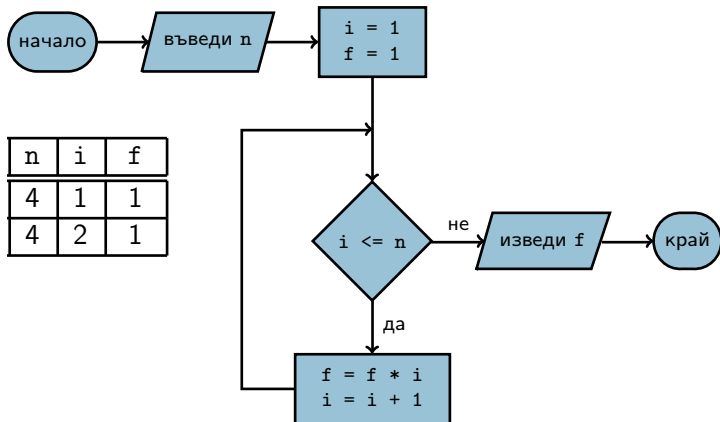
Индуктивни циклични процеси



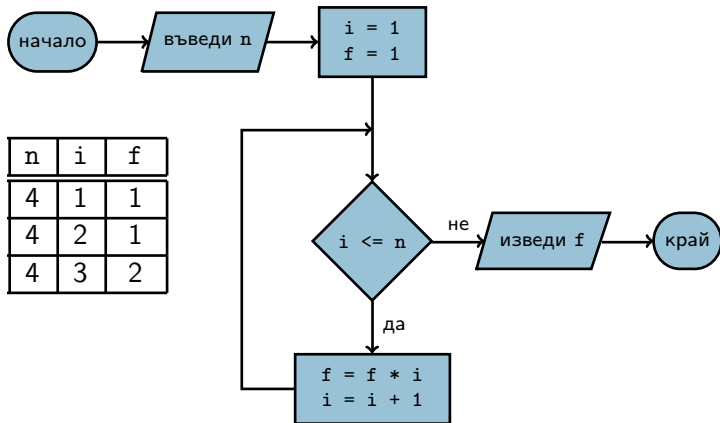
Индуктивни циклични процеси



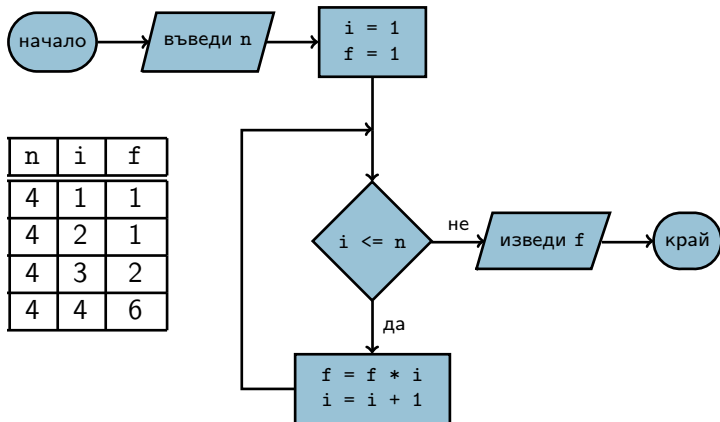
Индуктивни циклични процеси



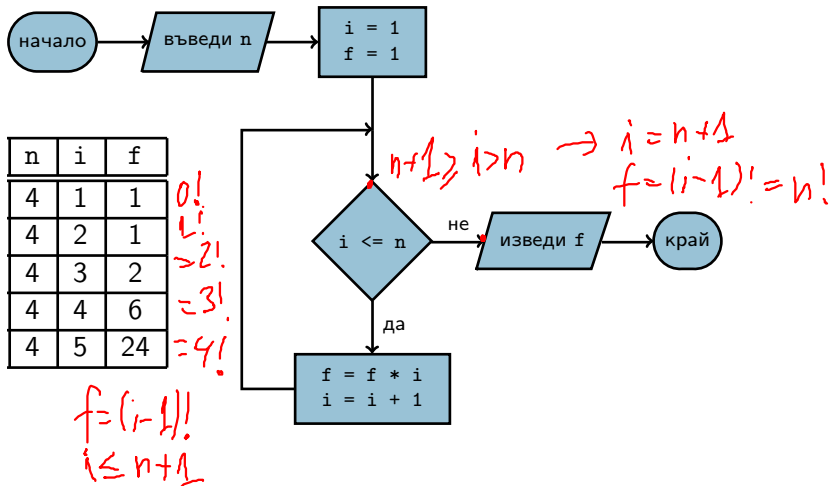
Индуктивни циклични процеси



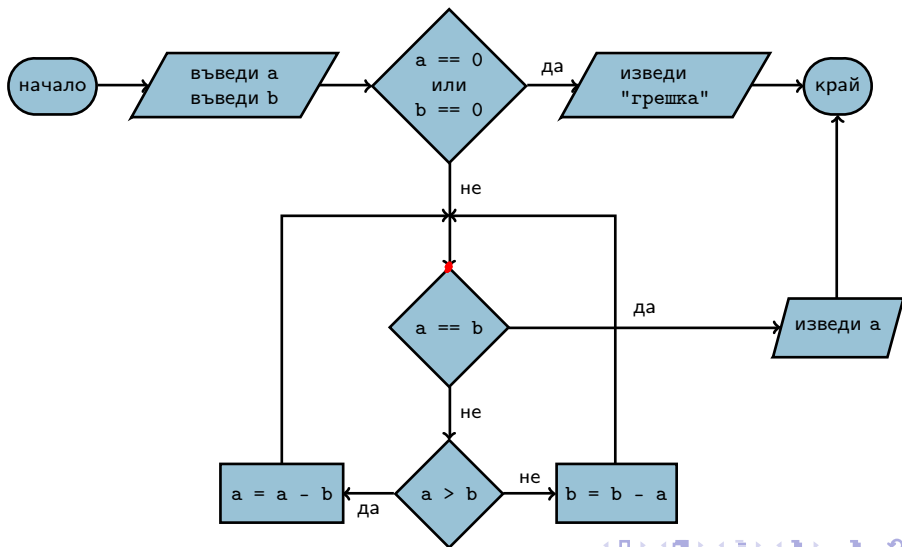
Индуктивни циклични процеси



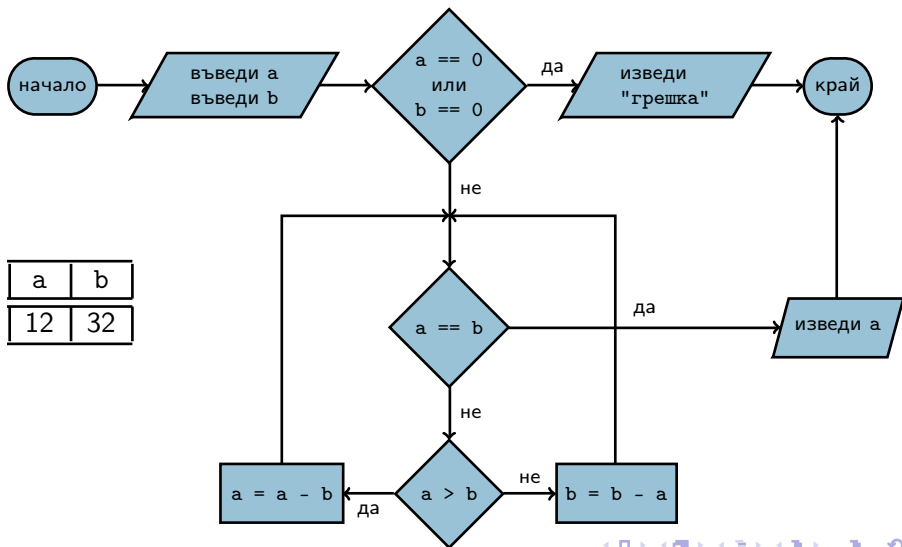
Индуктивни циклични процеси



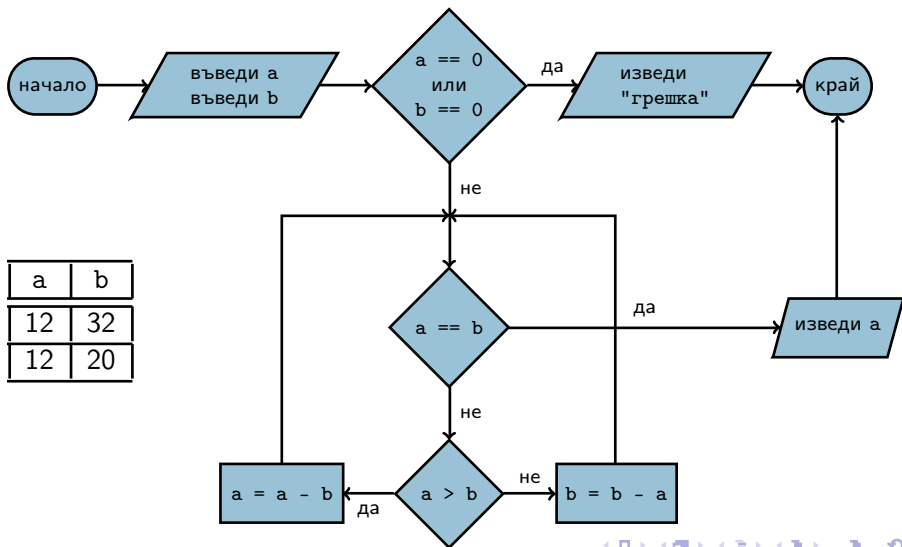
Итеративни циклични процеси



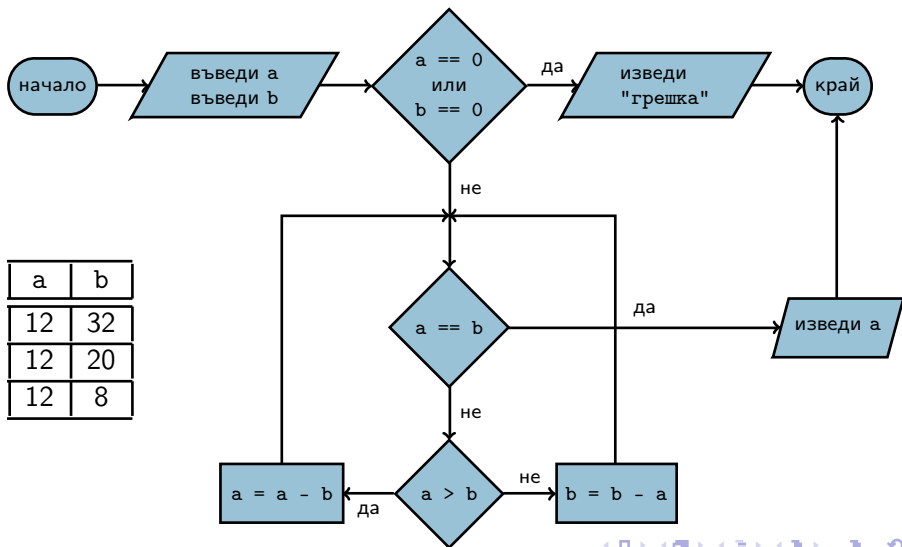
Итеративни циклични процеси



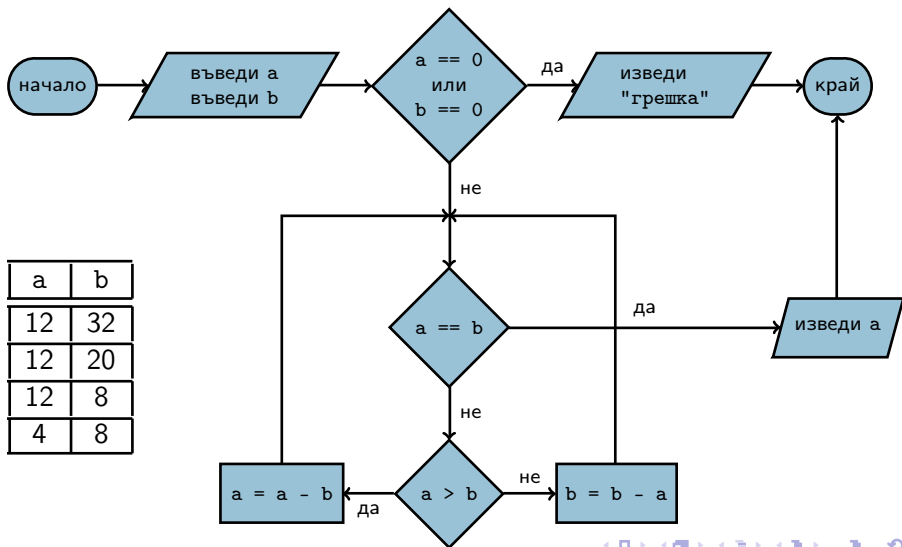
Итеративни циклични процеси



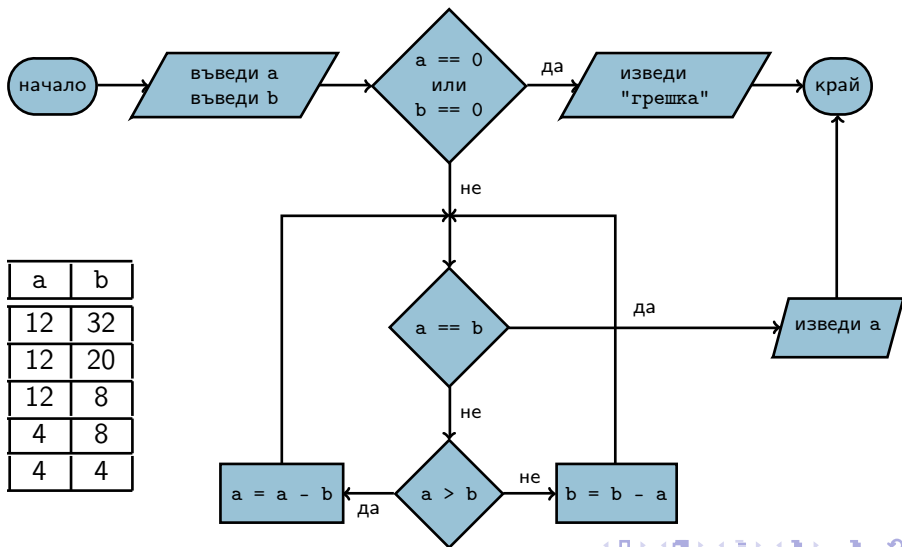
Итеративни циклични процеси



Итеративни циклични процеси



Итеративни циклични процеси



Структурни езици — разклонение

- 1 Въведи a, b
 - 2 Ако $a == 0$, към 5
 - 3 $x = -b / a$
 - 4 Премини към 9
 - 5 Ако $b = 0$, към 8
 - 6 “Няма решения”
 - 7 Премини към 9
 - 8 “Всяко x е решение”
 - 9 Край
- Въведи a, b
 - Ако $a == 0$
 - Ако $b == 0$
 - “Всяко x е решение”
 - Иначе
 - “Няма решения”
 - Иначе
 - $x = -b / a$

Структурни езици — индуктивен цикъл

- 1 Въведи n
- 2 $i = 1$
- 3 $f = 1$
- 4 Ако $i > n$, към 8
- 5 $f = f * i$
- 6 $i = i + 1$
- 7 Премини към 4
- 8 Изведи f
- 9 Край

- Въведи n
- $i = 1$
- $f = 1$
- Повтаряй n пъти
 - $f = f * i$
 - $i = i + 1$
- Изведи f

Структурни езици — итеративен цикъл

- 1 Въведи a , b
 - 2 Ако $a == b$, към 6.
 - 3 Ако $a > b$, към 5.
 - 4 $b = b - a$; към 2.
 - 5 $a = a - b$; към 2.
 - 6 Изведи a
 - 7 Край
- Въведи a , b
 - Докато $a != b$
 - Ако $a > b$
 - $a = a - b$
 - В противен случай
 - $b = b - a$
 - Изведи a

Основни понятия

- **Операция (operator)** $+, -, *, /, \&\&, ||, \<=, \<\< \dots$
- **Израз (expression)**
- **Оператор/команда (statement)**
- $\langle \text{израз} \rangle ::= \langle \text{константа} \rangle \mid \langle \text{променлива} \rangle \mid$
 $\langle \text{едноместна_операция} \rangle \langle \text{израз} \rangle \mid$
 $\langle \text{израз} \rangle \langle \text{двуместна_операция} \rangle \langle \text{израз} \rangle$
- $\langle \text{оператор} \rangle ::= \langle \text{израз} \rangle ;$

Оператор за присвояване

- `<променлива> = <израз>;`

Оператор за присвояване

- $\langle \text{променлива} \rangle = \langle \text{израз} \rangle ;$
- $\langle \text{lvalue} \rangle = \langle \text{rvalue} \rangle ;$

Оператор за присвояване

- $\langle \text{променлива} \rangle = \langle \text{израз} \rangle ;$
- $\langle \text{lvalue} \rangle = \langle \text{rvalue} \rangle ;$
- $\langle \text{lvalue} \rangle$ — място в паметта със стойност, която може да се променя
 - **Пример:** променлива

Оператор за присвояване

- $\langle \text{променлива} \rangle = \langle \text{израз} \rangle ;$
- $\langle \text{lvalue} \rangle = \langle \text{rvalue} \rangle ;$
- $\langle \text{lvalue} \rangle$ — място в паметта със стойност, която може да се променя
 - **Пример:** променлива
- $\langle \text{rvalue} \rangle$ — временна стойност, без специално място в паметта
 - **Пример:** константа, литерал, резултат от пресмятане

Оператор за присвояване

<израз>

- $\langle \text{променлива} \rangle = \langle \text{израз} \rangle ;$
- $\langle \text{lvalue} \rangle = \langle \text{rvalue} \rangle ;$
- $\langle \text{lvalue} \rangle$ — място в паметта със стойност, която може да се променя
 - **Пример:** променлива
- $\langle \text{rvalue} \rangle$ — временна стойност, без специално място в паметта
 - **Пример:** константа, литерал, резултат от пресмятане
- стандартно преобразуване на типовете:
 $\langle \text{rvalue} \rangle$ се преобразува до типа на $\langle \text{lvalue} \rangle$

Присвояването като операция

- **дясноасоциативна** операция

Присвояването като операция

- **дясноасоциативна** операция
- $a = b = c = 2;$

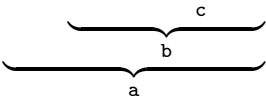
Присвояването като операция

- дясноасоциативна операция
- $a = (b = (c = 2));$
- ~~$((a = b) = c) = 2);$~~

Присвояването като операция

- **дясноасоциативна** операция

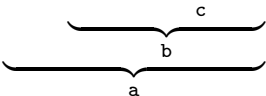
- $a = (b = (c = 2))$



Присвояването като операция

- **дясноасоциативна** операция

- $a = (b = (c = 2))$



- **Пример:** `cout << x + (b = 2);`

Присвояването като операция

- **дясноасоциативна** операция

- $a = (b = (c = 2))$

- **Пример:** `cout << x + (b = 2);`

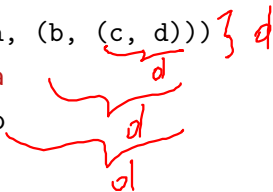
- **Пример:** $(a = b) = a + 3;$

Операция за изброяване

- `<израз1>, <израз2>`
- оценява и двата израза, но крайният резултат е оценката на втория израз

Операция за изброяване

- $\langle \text{израз1} \rangle, \langle \text{израз2} \rangle$
- оценява и двата израза, но крайният резултат е оценката на втория израз
- $a, b, c, d \Leftrightarrow (a, (b, (c, d))) \} d$
- **дясноасоциативна**
- използва се рядко



Операция за изброяване

- `<израз1>`, `<израз2>`
- оценява и двата израза, но крайният резултат е оценката на втория израз
- $a, b, c, d \Leftrightarrow (a, (b, (c, d)))$
- **дясноасоциативна**
- използва се рядко
- **Пример:** `a = (cout << x, x);`

Съкратени оператори за присвояване

- $a = a + 2 \Leftrightarrow a += 2$

Съкратени оператори за присвояване

- $a = a + 2 \Leftrightarrow a += 2$
- $--, *=, /=, \% =$

Съкратени оператори за присвояване

- $a = a + 2 \Leftrightarrow a += 2$
- $--, *=, /=, \% =$
- $a = a + 1 \Leftrightarrow ++a$

Съкратени оператори за присвояване

- $a = a + 2 \Leftrightarrow a += 2$
- $--, *=, /=, %=$
- $a = a + 1 \Leftrightarrow ++a$
- $a = a - 1 \Leftrightarrow --a$

Съкратени оператори за присвояване

- $a = a + 2 \Leftrightarrow a += 2$
- $-=$, $*=$, $/=$, $\%=$
- $a = a + 1 \Leftrightarrow ++a$
- $a = a - 1 \Leftrightarrow --a$
- $a++$ увеличава a с 1, но връща предишната стойност на a

Съкратени оператори за присвояване

- $a = a + 2 \Leftrightarrow a += 2$
- $-=$, $*=$, $/=$, $\%=$
- $a = a + 1 \Leftrightarrow ++a$
- $a = a - 1 \Leftrightarrow --a$
- $a++$ увеличава a с 1, но връща предишната стойност на a
 - $a++ \Leftrightarrow (a = (tmp = a) + 1, tmp)$

Съкратени оператори за присвояване

- $a = a + 2 \Leftrightarrow a += 2$
- $-=$, $*=$, $/=$, $\%=$
- $a = a + 1 \Leftrightarrow ++a$
- $a = a - 1 \Leftrightarrow --a$
- $a++$ увеличава a с 1, но връща предишната стойност на a
 - $a++ \Leftrightarrow (a = (tmp = a) + 1, tmp)$
- $a--$ действа аналогично

Съкратени оператори за присвояване

- $a = a + 2 \Leftrightarrow a += 2$
- $-=, *=, /=, \%=$
- $a = a + 1 \Leftrightarrow ++a$
- $a = a - 1 \Leftrightarrow --a$
- $a++$ увеличава a с 1, но връща предишната стойност на a
 - $a++ \Leftrightarrow (a = (tmp = a) + 1, tmp)$
- $a--$ действа аналогично
- $++a$ връща a , което е **<lvalue>**

Съкратени оператори за присвояване

- $a = a + 2 \Leftrightarrow a += 2$
- $-=$, $*=$, $/=$, $\%=$
- $a = a + 1 \Leftrightarrow ++a$
- $a = a - 1 \Leftrightarrow --a$
- $a++$ увеличава a с 1, но връща предишната стойност на a
 - $a++ \Leftrightarrow (a = (tmp = a) + 1, tmp)$
- $a--$ действа аналогично
- $++a$ връща a , което е **<lvalue>**
 - **Пример:** $++a += 5;$

Съкратени оператори за присвояване

- $a = a + 2 \Leftrightarrow a += 2$
- $-=$, $*=$, $/=$, $\%=$
- $a = a + 1 \Leftrightarrow ++a$
- $a = a - 1 \Leftrightarrow --a$
- $a++$ увеличава a с 1, но връща предишната стойност на a
 - $a++ \Leftrightarrow (a = (tmp = a) + 1, tmp)$
- $a--$ действа аналогично
- $++a$ връща a , което е **<lvalue>**
 - **Пример:** $++a += 5;$
- $a++$ връща предишната стойност на a , което е **<rvalue>**

Съкратени оператори за присвояване

- $a = a + 2 \Leftrightarrow a += 2$
- $-=$, $*=$, $/=$, $\%=$
- $a = a + 1 \Leftrightarrow ++a$
- $a = a - 1 \Leftrightarrow --a$
- $a++$ увеличава a с 1, но връща предишната стойност на a
 - $a++ \Leftrightarrow (a = (tmp = a) + 1, tmp)$
- $a--$ действа аналогично
- $++a$ връща a , което е **<lvalue>**
 - **Пример:** $++a += 5;$
- $a++$ връща предишната стойност на a , което е **<rvalue>**
 - **Пример:** $x = a++ * b;$ ~~$a++ += 5;$~~

Оператор за блок

- { { <оператор> } }

Оператор за блок

- { { <оператор> } }
- { <оператор₁> <оператор₂> ... <оператор_n> }

Оператор за блок

- { { <оператор> } }
- { <оператор₁> <оператор₂> ... <оператор_n> }
- Вложени блокове

```
{  
    int x = 2;  
    {  
        x += 2;  
        cout << x;  
    }  
}
```

Област на действие (scope)

- областта на действие се простира от дефиницията на променливата до края на блока, в който е дефинирана

Област на действие (scope)

- областта на действие се простира от дефиницията на променливата до края на блока, в който е дефинирана
- дефиниция на променлива със същото име в същия блок е забранена

Област на действие (scope)

- областта на действие се простира от дефиницията на променливата до края на блока, в който е дефинирана
- дефиниция на променлива със същото име в същия блок е забранена
- дефиниция на променлива във вложен блок покрива всички външни дефиниции със същото име

Област на действие (scope) — пример

```
int x = 0;
{
    x++;
    double y = 2.3;
    {
        double x = 1.6;
        y = x * x;
    }
    double y = 2.4;
    x += 3;
}
x += 4;
y /= 2.1;
```

Област на действие (scope) — пример

```

int x = 0;
{
    x++;
    double y = 2.3;
    {
        double x = 1.6;
        y = x * x;
    }
    double y = 2.4;
    x += 3;
}
x += 4;
y /= 2.1;

```



Област на действие (scope) — пример

```

int x = 0;
{
    x++;
    double y = 2.3;
    {
        double x = 1.6;
        y = x * x;
    }
    double y = 2.4;
    x += 3;
}
x += 4;
y /= 2.1;

```



Област на действие (scope) — пример

```

int x = 0;
{
    x++;
    double y = 2.3;
    {
        double x = 1.6;
        y = x * x;
    }
    double y = 2.4;
    x += 3;
}
x += 4;
y /= 2.1;

```

	x	y		
...	1	2.3		...

Област на действие (scope) — пример

```

int x = 0;
{
    x++;
    double y = 2.3;
    {
        double x = 1.6;
        y = x * x;
    }
    double y = 2.4;
    x += 3;
}
x += 4;
y /= 2.1;

```

	x	y	x	
...	1	2.3	1.6	...

Област на действие (scope) — пример

```

int x = 0;
{
    x++;
    double y = 2.3;
    {
        double x = 1.6;
        y = x * x;
    }
    double y = 2.4;
    x += 3;
}
x += 4;
y /= 2.1;

```

	x	y	x	
...	1	2.56	1.6	...

Област на действие (scope) — пример

```

int x = 0;
{
    x++;
    double y = 2.3;
    {
        double x = 1.6;
        y = x * x;
    }
    double y = 2.4;
    x += 3;
}
x += 4;
y /= 2.1;

```

	x	y		
...	1	2.56		...

Област на действие (scope) — пример

```

int x = 0;
{
    x++;
    double y = 2.3;
    {
        double x = 1.6;
        y = x * x;
    }
    double y = 2.4;
    x += 3;
}
x += 4;
y /= 2.1;

```

	x	y		
...	4	2.56		...

Област на действие (scope) — пример

```

int x = 0;
{
    x++;
    double y = 2.3;
    {
        double x = 1.6;
        y = x * x;
    }
    double y = 2.4;
    x += 3;
}
x += 4;
y /= 2.1;

```



Област на действие (scope) — пример

```

int x = 0;
{
    x++;
    double y = 2.3;
    {
        double x = 1.6;
        y = x * x;
    }
    double y = 2.4;
    x += 3;
}
x += 4;
y /= 2.1;

```



Празен оператор

- ;
- ; \Leftrightarrow {}
- няма никакъв ефект

Условен оператор

- `if (<израз>) <оператор> [else <оператор>]`

Условен оператор

- `if (<израз>) <оператор> [else <оператор>]`
- Съкратената форма \Leftrightarrow пълна форма с празен оператор

Условен оператор

- `if (<израз>) <оператор> [else <оператор>]`
- Съкратената форма \Leftrightarrow пълна форма с празен оператор
 - `if (A) X; \Leftrightarrow if (A) X; else;`

Условен оператор

- `if (<израз>) <оператор> [else <оператор>]`
- Съкратената форма \Leftrightarrow пълна форма с празен оператор
 - `if (A) X; \Leftrightarrow if (A) X; else;`
- Пример: `if (x < 2) y = 2;`

Условен оператор

- `if (<израз>) <оператор> [else <оператор>]`
- Съкратената форма \Leftrightarrow пълна форма с празен оператор
 - `if (A) X; \Leftrightarrow if (A) X; else;`
- Пример: `if (x < 2) y = 2;`
- Пример: `if (x > 5) y = 5; else y = 3;`

Вложени условни оператори

Какво имаме предвид, когато пишем:

```
if (a > 0) (if (b > 0) cout << 1; ) else cout << 3;
```

Вложени условни оператори

Какво имаме предвид, когато пишем:

```
if (a > 0) if (b > 0) cout << 1; else cout << 3;
```

```
if (a > 0)
    if (b > 0)
```

```
    cout << 1;
```

```
else
```

```
    cout << 3;
```

или

```
if (a > 0)
    if (b > 0)
```

```
    cout << 1;
```

```
else
```

```
    cout << 3;
```

Вложени условни оператори

Какво имаме предвид, когато пишем:

```
if (a > 0) if (b > 0) cout << 1; else cout << 3;
```

```
if (a > 0) {
    if (b > 0)
```

```
    cout << 1;
```

```
else
```

```
    cout << 3;
```

```
}
```

или

```
if (a > 0) {
    if (b > 0)
```

```
    cout << 1;
```

```
}
```

```
else
```

```
    cout << 3;
```

Вложени условни оператори

Какво имаме предвид, когато пишем:

```
if (a > 0) if (b > 0) cout << 1; else cout << 3;
```

```
if (a > 0) {
    if (b > 0)
        // a > 0 && b > 0
        cout << 1;
    else
        // a > 0 && b ≤ 0
        cout << 3;
}
```

или

```
if (a > 0) {
    if (b > 0)
        // a > 0 && b > 0
        cout << 1;
}
else
    // a ≤ 0
    cout << 3;
```

Вложени условни оператори

Какво имаме предвид, когато пишем:

```
if (a > 0) if (b > 0) cout << 1; else cout << 3;
```

```
if (a > 0) {  
    if (b > 0)  
        // a > 0 && b > 0  
        cout << 1;  
    else  
        // a > 0 && b ≤ 0  
        cout << 3;  
}
```

Съкратено оценяване на логически операции

Представяне на логически операции с вложени условни оператори:

```
if (!A) X;  
else    Y;           ⇔           if (A) Y;  
                                     else X;
```


Съкратено оценяване на логически операции

Представяне на логически операции с вложени условни оператори:

```
if (!A) X;  
else    Y;      ⇔      if (A) Y;  
                                     else X;
```

```
if (A && B) X;  
else      Y;      ⇔      if (A)  
                                     if (B) X;  
                                     else Y;  
                                     else Y;
```

Съкратено оценяване на логически операции

Представяне на логически операции с вложени условни оператори:

```
if (!A) X;
else    Y;      ⇔      if (A) Y;
                    else    X;
```

```
if (A && B) X;
else      Y;      ⇔      if (A)
                        if (B) X;
                        else   Y;
                        else Y;
```

```
if (A || B) X;
else      Y;      ⇔      if (A) X;
                        else
                        if (B) X;
                        else   Y;
```

Съкратено оценяване на логически операции

Представяне на логически операции с вложени условни оператори:

```
if (!A) X;
else    Y;      ⇔      if (A) Y;
                    else    X;
```

```
if (A && B) X;
else      Y;      ⇔      if (A)
                    if (B) X;
                    else  Y;
                    else Y;
```

```
if (A || B) X;
else      Y;      ⇔      if (A) X;
                    else
                    if (B) X;
                    else  Y;
```

Пример: `if (x > 0 && log(x) < 5) ...`

Пример: `if (x == 0 || y / x == 1) ...`

Дано: a, b

$$ax^2 > b$$

Пример: $a=2, b=8 \rightarrow 1$

$$2x^2 > 8 \\ (-\infty; -2) \cup (2; +\infty)$$



Всё x 3

Ничего x 4