

ИЗБОР: УСЛОВНИ ОПЕРАТОРИ

ЛЕКЦИОНЕН КУРС “ПРОГРАМИРАНЕ НА JAVA”



СТРУКТУРА НА ЛЕКЦИЯТА

- Контролен поток
- Оператори за избор
- “Висящ” else

КОНТРОЛЕН ПОТОК (CONTROL FLOW)

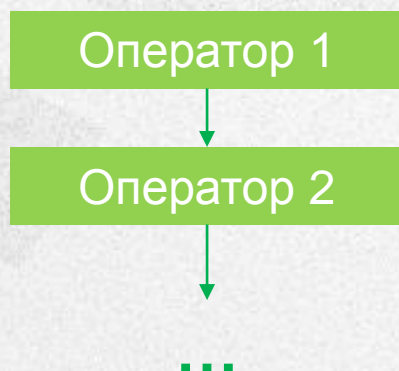
1 Какво е контролен поток?

Последователността на изпълняваните в програмата оператори

2 Как можем да хореографираме контролния поток?

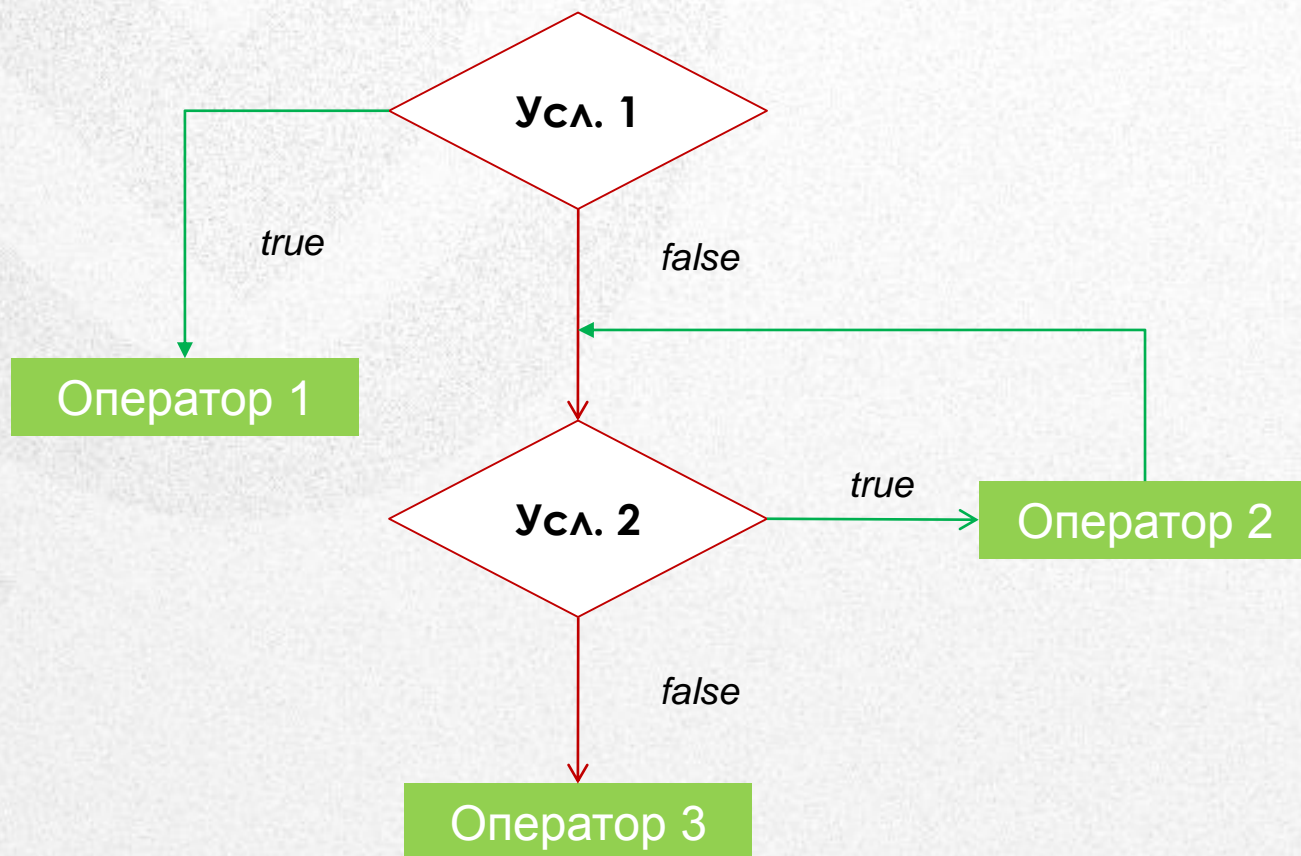
Оператори за цикли и условни оператори

2 Какъв е контролният поток от примера?



Последователен контролен поток

КОНТРОЛЕН ПОТОК С УСЛОВИЯ И ЦИКЛИ

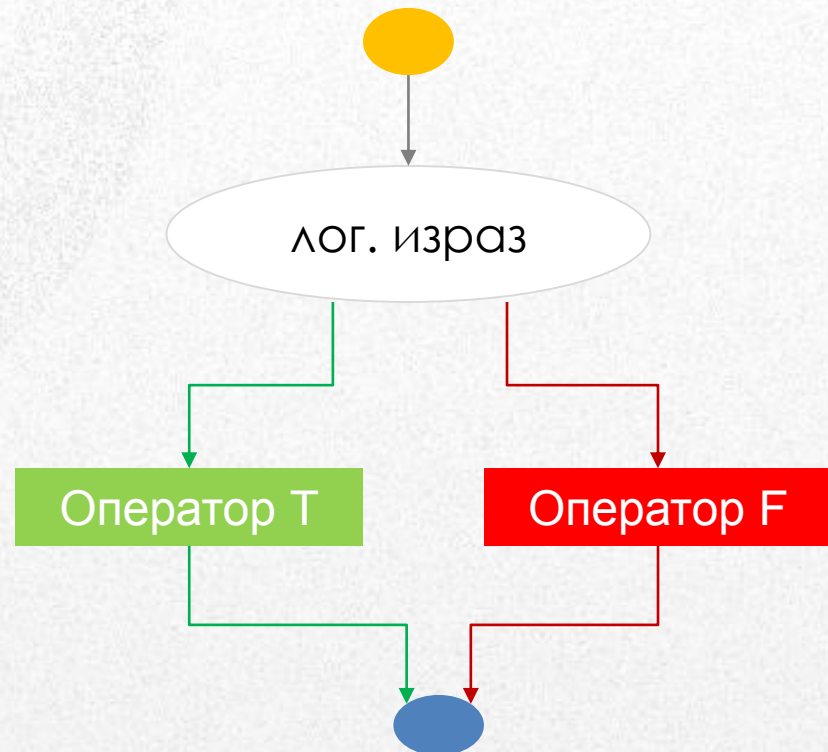


УПРАВЛЯВАЩИ КОНСТРУКЦИИ

- Процедурните езици за програмиране притежават някакъв вид управляващи конструкции
- Често между отделните езици има препокриване
- В Java ключовите думи за управляващи конструкции ВКЛЮЧВАТ:
 - if-else
 - while
 - do-while
 - for
 - switch
 - Не поддържа директно goto – възможно е програмиране на подобни, но много по-ограничени преходи

IF ОПЕРАТОР: РАЗКЛОНЯВАЩА СТРУКТУРА

```
if (логически израз) {  
    оператор T;  
}  
else {  
    оператор F;  
}
```



СИНТАКСИС

- If-оператор: избор между две алтернативи

EBNF: `if (израз) оператор else оператор`

За всяка алтернатива: един оператор !

Пример:

```
if (x == 0)
    System.out.print(0);
else
    System.out.print(y/x);
```

IF-ОПЕРАТОР: ПОВЕЧЕ ОПЕРАТОРИ

EBNF: `if (израз) оператор else оператор`



Повече оператори?

с `{ ... }` обединени като един

```
if (x == 0)    { //Exception: Div by 0
    System.out.print(0);
    x = y;
}
else {
    System.out.print(y/x);
    y = x;
}
```

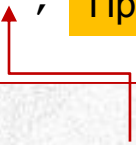

IF-ОПЕРАТОР: КРАТКА ФОРМА

EBNF (пълна форма):

if (израз) оператор **else** оператор

```
if ((a + b) <= c)
    System.out.print("не е триъгълник");
else ;
```

Празен оператор



1

Какво следва от синтаксиса?

EBNF (кратка форма):

if (израз) оператор

```
if ((a + b) <= c)
    System.out.print("не е триъгълник");
```

ТЕСТ: ТИПИЧНИ ГРЕШКИ

1 Какви грешки в примерите?

1

```
if a > b
...
else
...
```

Липсва скоба

2

```
if (a = 1)
...
else
...
```

Стойност на a?

3

```
if (x > y)
    x = y; y = 0;
if (y == 0)
...
```

Стойност на y?

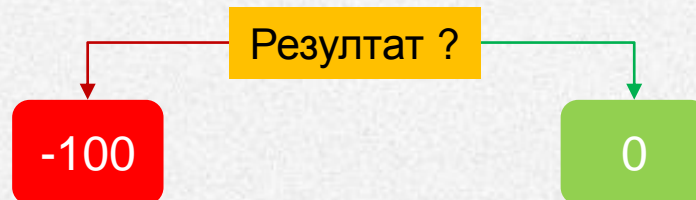
“ВИСЯЩ” ELSE (DANGLING ELSE)

Входни стойности:

x	y	z	ok
2	1	100	-100

```
if (x > y)
    if (y > z)
        ok = 1;
else
    ok = 0;
```

```
if (x > y)
    if (y > z)
        ok = 1;
else
    ok = 0;
```



“ВИСЯЩ” ELSE: ПРОБЛЕМ

1 Разлики между двете версии?

```
if (a)
    if (b)
        s1;
else
    s2;
```

```
if (a)
    if (b)
        s1;
else
    s2;
```

Двата варианта са идентични

ДВАТА ВАРИАНТА: ИДЕНТИЧНИ

Само по различен начин изразен (Layout)

→ намерение на програмиста:

else s2; принадлежи към ...

- 1. Форма: външен if
- 2. Форма: вътрешен if

Значение на една програма: независимо от Layout

1

Какво правим?

```
if (a)    if (b)    s1;  else s2;
```



Констатация: 2. форма е обвързваща !

“ВИСЯЩ”-ELSE-ПРОБЛЕМ: ПРИЧИНА

Граматиката на Java не е еднозначна !

Оператор за избор ::=

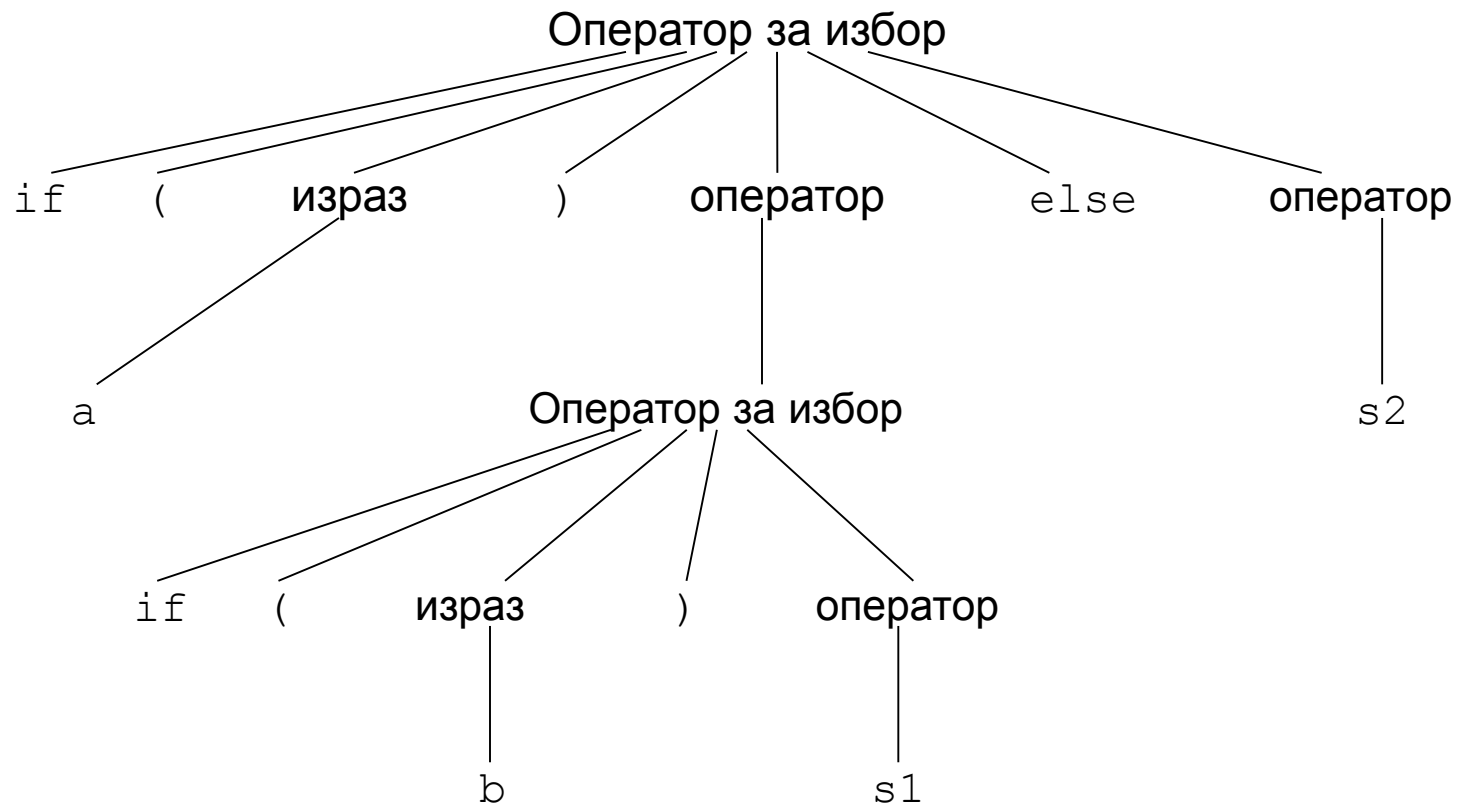
```
if ( израз ) оператор [ else оператор ] |  
switch ...
```

3 правила

Оператор за избор ::=

```
if ( израз ) оператор |  
if ( израз ) оператор else оператор |  
switch ...
```


СИНТАКТИЧНО ДЪРВО: ВАРИАНТ 1



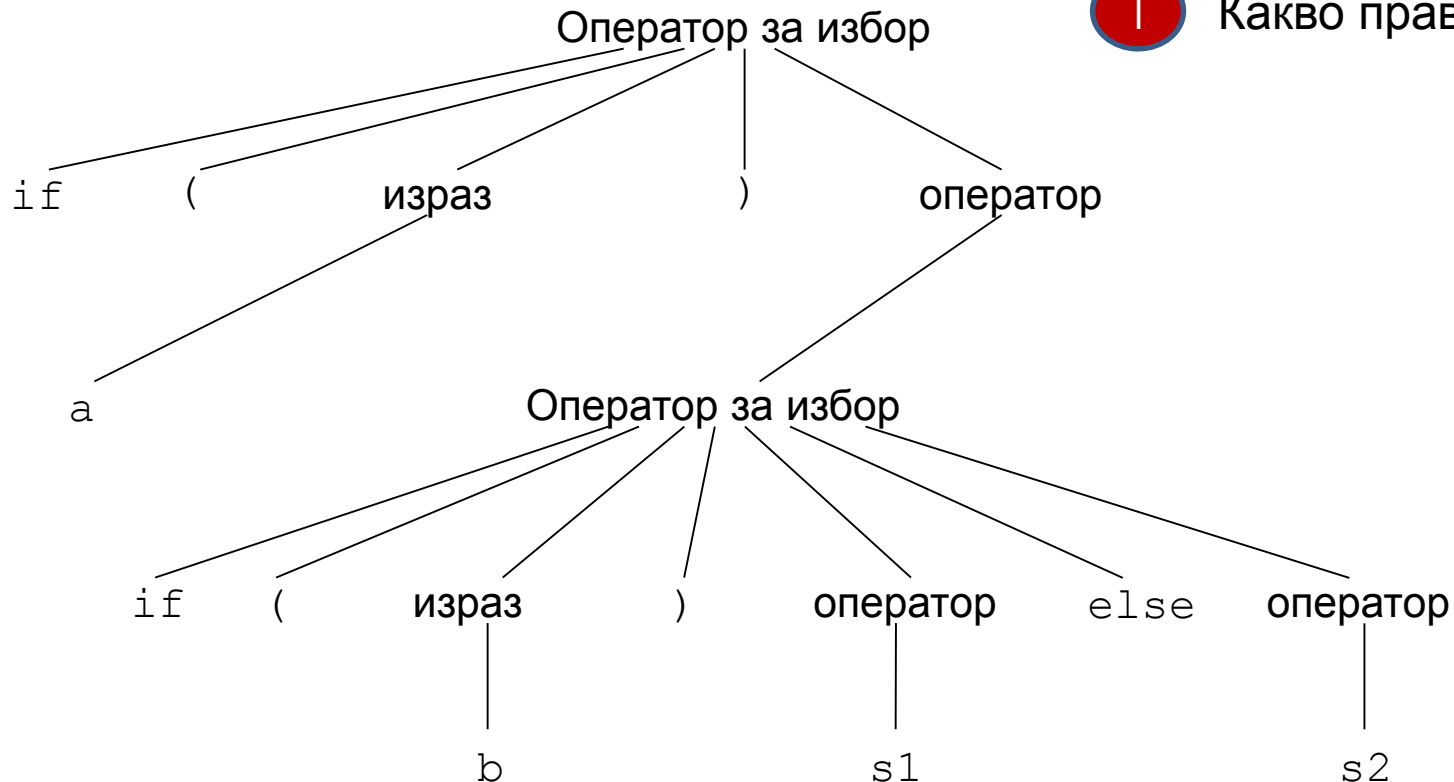
Изведен:

```
if ( a ) if ( b ) s1 else s2
```

СИНТАКТИЧНО ДЪРВО: ВАРИАНТ 2

Две синтактични дървета за същия оператор

1 Какво правим?



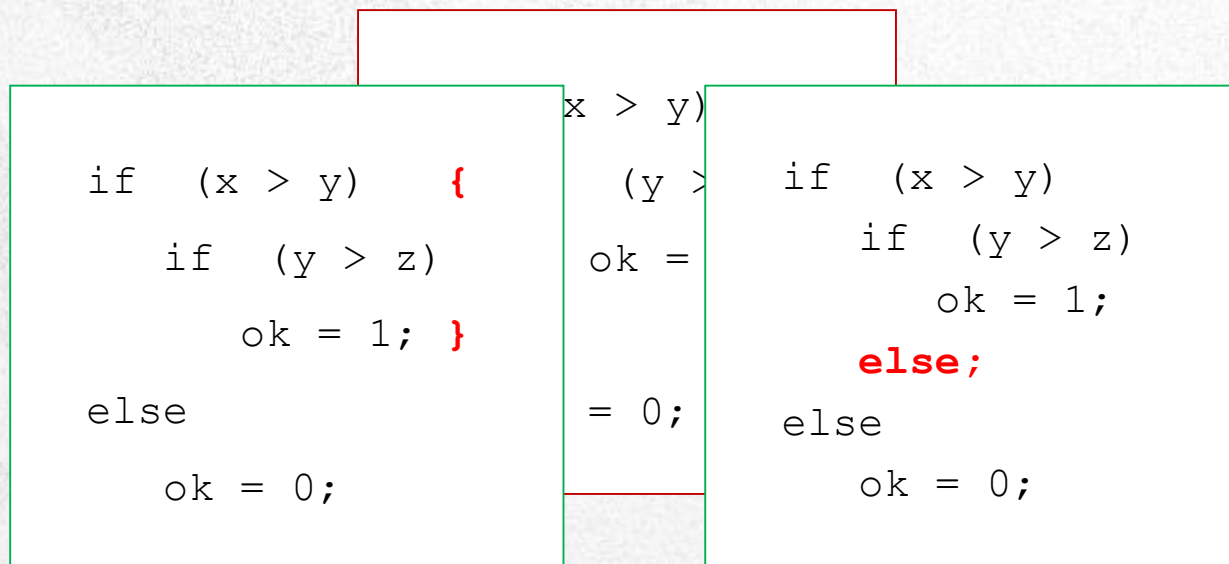
Изведен:

if (a) if (b) s1 else s2

ПРОБЛЕМИ

1

Какво правим, ако все пак искаме вариант1?



2

Как би могъл да се направи синтаксисът еднозначен?

endif

ИЗБОР НА УСЛОВЕН ОПЕРАТОР: SWITCH

Условен оператор

Избор между n оператора

$n = 1$
(съств. $n=2$
с празен оператор)

```
if (B)
    S1
```

$n = 2$

```
if (B)
    S1
else
    S2
```

$n > 2$

```
if (B)
    S1
else if (B2)
    S2
else if (B3)
    ...
```

$n \geq 2$ (Форма на условие)

$\text{expr} == \text{стойност } v_1$
 expr с тип byte, short, int, char

```
switch (expr)
    case v1: ...
    case v2: ...
```

SWITCH-ОПЕРАТОР: МОТИВАЦИЯ

```
if (n == 0)
    System.out.print(" нула ");
else if (n == 1)
    System.out.print(" едно ");
else if (n == 2)
    System.out.print(" две ");

. . . // else if до 9

else // n > 9
    System.out.print(" > девет ");
```

Стойност на израз
(променлива n)

Повтарящо тестване при
равенство с
предварително зададени
стойности (0, 1, 2 ...)

SWITCH-ОПЕРАТОР: ИЗБОР ОТ ПОВЕЧЕ ВАРИАНТИ

```
switch (n)    {  
    case 0:    System.out.print( "0" );  
               break;  
    case 1:    System.out.print( "1" );  
               break;  
    case 2:    System.out.print( "2" );  
               break;  
               ....  
    default:   System.out.println( " >9" );  
}
```

→ Семантиката се запазва

SWITCH: ПРИМЕР

1 Коментар на примера?

```
final int jan = 1, feb = 2, mar = 3, ... dec = 12;
int month, year, numDays;
... // read month, year
switch (month) {
    case feb:
        if (((year % 4) == 0) && ((year % 100) != 0)
            || ((year % 400) == 0)) //high year test
            numDays = 29;
        else numDays = 28;
        break;
    case apr: case jun: case sep: case nov:
        numDays = 30; break;
    default: numDays = 31;
}
```

ПРИМЕР

1 Какъв резултат?

```
int n = 1;
switch (n) {
    case 0: System.out.print( "0" );
    case 1: System.out.print( "1" );
    case 2: System.out.print( "2" );
    case 3: System.out.print( "3" );
}
System.out.println();
```

123

BREAK-ОПЕРАТОР

- Ако всеки оператор в switch завършва с break
 - Това е еквивалентно на вградена последователност от if оператори
- Предназначение на break
 - Завършва изпълнението на оператора switch в разклонението, където е даден
- По принцип всяка алтернатива на switch трябва да има break
 - При пропускане изпълнението на switch продължава до неговия край

BREAK-ОПЕРАТОР

1

Какъв резултат?

```
int n = 1;
switch (n) {
    case 0: System.out.print( "0" ); break;
    case 1: System.out.print( "1" ); break;
    case 2: System.out.print( "2" ); break;
    case 3: System.out.print( "3" ); break;
}
System.out.println();
```

1

БЛАГОДАРЯ ЗА ВНИМАНИЕТО!

КРАЙ “ИЗБОР: УСЛОВНИ ОПЕРАТОРИ”

