

Java

ПРЕПОДАВАТЕЛ: ИНЖ. ВЕСЕЛИНА МАРИНОВА

Оператори за извеждане на информация на екрана:

```
/* My first program Version 1 */
```

```
public class Example1 {  
public static void main (String args []) {
```

```
    System.out.println ("My first Java program");
```

```
//отпечатва съобщението: My first Java program и преминава на нов ред
```

```
    System.out.print ("My first Java program again");
```

```
// отпечатва: My first Java program и маркера остава на същия ред  
    }  
}
```

Отпечатване на текстова и цифрова информация:

```
public class Example1 {  
    public static void main (String args []) {  
        System.out.println ("x="+15);
```

```
//x=15
```

текст + числото

```
        System.out.print ("x="+20+"\t" +"y =" +25);
```

```
// x=20    y=25
```

оставя табулация между разпечатваното

```
        System.out.print ("x="+20+"\t" +"y =" +25+"\t" +"z =" +(14+22)+"\n");
```

```
//x=20    y=25    z=36
```

символ за преминаване на нов ред

```
    }
```

изчислява израза и отпечатва резултата

```
}
```

Типове променливи (*String*, *int*, *double*)

1. *String*....използва се за съхраняване на текстова информация, напр. “Hello world”

Пример:

```
public static void main(String args[])
{
    String s = “Hello world”;
    System.out.println(s);
}
```

2. *int* използва се за съхраняване на цели числа(положителни или отрицателни)

Пример :

```
public static void main(String args[])
{
    int age = 59;
    System.out.println(age);
}
```

3. *double* Използва се за съхраняване на реални числа (с дробна част).
double означава “двойна точност”.

Пример :

```
public static void main(String args[])
{
    double d = -137.8036;
    System.out.println(d);
    d = 1.45667E23; // Научен формат, означава.. 1.45667 X 1023
}
```

Декларация и инициализация:

```
double x = 1.6;
```

Реално се извършват 2 действия. Ние декларираме *x* от тип *double* и инициализираме *x* със стойност 1.6. Това може да бъде направено и с 2 реда код:

```
double x;        // декларираме x от тип double
x = 1.6;          // инициализираме x със стойност 1.6
```

Използване на числови променливи

Оператор за присвояване:

Операторът за присвояване е стандартния знак (=) и означава , че присвояваме стойност на променлива.

```
int i = 3;          // Валидно  
3 = i;             // Не валидно
```

```
double p;
```

```
double j = 47.2;
```

```
p = j;             // присвояваме стойността на променливата j на p. Двете p и j са равни на 47.2
```

Деклариране на повече променливи :

Можем да декларираме повече еднотипни променливи на 1 ред:

```
double d, mud, puma;
```

```
//променливите са само декларирани
```

```
double x = 31.2, m = 37.09, z, p = 43.917;
```

```
//x, m, и p са декларирани и инициализирани
```

```
// z е само декларирана
```

Основни аритметични операции:

Основните аритметични операции са +, -, * (умножение), / (деление), и % (остатък от деление).

Пример:

```
System.out.println(5%3); // 2.
```

Защото 5 при целочислено деление на 3 дава остатък 2 . **Тази операция ни дава остатъка.**

Алгебрични правила:

```
System.out.println(5 + 3 * 4 - 7); //10
```

```
System.out.println(8 - 5*6 / 3 + (5 - 6) * 3); // -5
```

Разлики с алгебрата:

`count = count + 3;` //това не е възможно в алгебрата, но в програмирането означава
//новата стойност на **count** е равна на старата стойност на **count + 3**.

```
int count = 15;  
count = count + 3;  
System.out.println(count);    //18
```

Операции ++ и --:

`x++;` $\Leftrightarrow x = x + 1;$
`x--;` $\Leftrightarrow x = x - 1;$
`x++ ;` $\Leftrightarrow ++x;$
`x-- ;` $\Leftrightarrow --x;$

```
int y = 3;  
y++;  
System.out.println(y);    //4
```


Съкратени оператори:

Примери:

a. +=

$x += 3;$ $x = x + 3;$

b. -=

$x -= y - 2;$ $x = x - (y - 2);$

c. *=

$z *= 46;$ $z = z * 46;$

d. /=

$p /= x - z;$ $p = p / (x - z);$

e. %=

$j \% = 2 ;$ $j = j \% 2;$

Примери :

```
int g = 409;  
g += 5;           //409+5  
System.out.println(g); //414  
double d = 20.3;  
double m =10.0;  
m*=d -1;          //10*(d-1)  
System.out.println(m); //193
```

$x++$ нараства x **след** като се използва в израза.

$++x$ нараства x **преди** да се използва в израза.

Същото важи и за $x-$ и за $-x$.

Пример:

```
int q = 78;  
int p = 2 + q++;  
System.out.println("p = " + p + ", q = " + q); //p = 80, q = 79  
int q = 78;  
int p = ++q + 2;  
System.out.println("p = " + p + ", q = " + q); //p = 81, q = 79
```

Целочислено деление:

Пример:

```
int x = 5;  
int y = 2;  
System.out.println(x / y);  
    //И двете x и y се цели числа, но реалният отговор е 2.5  
    //целочисленото деление ще ни даде резултат 2
```

Смесване на числови типове и константи

Когато различни числа участват в аритметични изрази (int и double) се получава смесване на типовете при изчисление.

Java не харесва загубата на информация:

Java няма нормално да съхрани информацията в променливите, както в математиката. (ще се получи загуба на информацията)

1. Пример : **губим** информация:

```
double d = 29.78;
```

```
int i = d;           // i=29
```

Явно преобразуване :

```
int i = (int)d;      // Преобразуваме d до цяло число.
```

2. Пример: **без загуба** на информация:

```
int j = 105;
```

```
double d = j;       //d=105.0
```

Преобразувания:

Когато имаме 2 различни типа данни в математична операция:

Пример:

```
int i = 4;  
double d = 3;  
double ans = i/d;           // 1.333333333333333...резултата е double
```

$20 + 5 * 6.0$ ще върне *double*. Въпреки , че за нас 6.0 е цяло число, но записано с десетична точка, резултата ще е от тип *double*.

Примери:

$3 + 5.0/2 + 5 * 2 - 3$	\Leftrightarrow 12.5
$3.0 + 5/2 + 5 * 2 - 3$	\Leftrightarrow 12.0
$(int)(3.0 + 4)/(1 + 4.0) * 2 - 3$	\Leftrightarrow -.2

Пример:

```
int k=12, l=10;  
double d=12.5, f=20.5;  
System.out.println("Example 1");  
System.out.println( (double)(k/l + d/f)+d/l );  
System.out.println("Example 2");  
System.out.println( (int)(k/l + d/f)+d/l );  
System.out.println("Example 3");  
System.out.println( (double)(k*(int)f)+(int)d/l );  
System.out.println("Example 4");  
System.out.println( (int)((k/l + d/f)+d/l ) );  
System.out.println("Example 5");  
System.out.println( (int)((k/l+(int)(d/f))+d/l ) );  
System.out.println("Example 6");  
System.out.println( (int)((((double)(k/l + d/f))+d/l ) ) );
```

Результат:

Example 1

2.8597560975609757

Example 2

2.25

Example 3

241.0

Example 4

2

Example 5

2

Example 6

2

Методи на *Math* Class

Един от най-често използвания метод на *Math* class е *sqrt()* ... който означава корен квадратен.

Пример:

```
double p = Math.sqrt(17);
```

Трябва да съхраним резултата като *double.... p* в този случай.

Сигнатура на методите:

За да опишем методите на *Math* class... ще трябва да използваме допустимата сигнатура на метода. Какво означава сигнатура: Описанието за сигнатура на *sqrt()* метода изглежда така:

```
double sqrt( double x )
```

Method	Signature	Description
abs int	abs(int x)	Връща абсолютната стойност на x
abs double	abs(double x)	Връща абсолютната стойност на x
pow double	pow(double b, double e)	Връща b вдигнато на степен e
sqrt double	sqrt(double x)	Връща корен квадратен от x
ceil double	ceil(double x)	Връща следващото по-голямо от x
floor double	floor(double x)	Връща следващото по-малко от x
min double	min(double a, double b)	Връща по-малкото от a и b
max double	max(double a, double b)	Връща по-голямото от a и b
min int	min(int a, int b)	Връща по-малкото от a и b
max int	max(int a, int b)	Връща по-голямото от a и b
random double	random()	Случайно генерирано число (range $0 \leq r < 1$)
round long	round(double x)	Закръглява x до цяло число
PI double	PI	Връща 3.14159625.....

Пример:

```
import java.util.*;
public class Example_new {
    public static void main(String[] args) {
        Scanner kb= new Scanner(System.in);
        int i = kb.nextInt();
        int j = kb.nextInt();
        System.out.println("min ->" + Math.min(i,j));
        double x = kb.nextDouble();
        double y = kb.nextDouble();
        System.out.println( " method abs ->" + Math.abs(x));
        System.out.println( " method abs ->" + Math.abs(y));
        System.out.println(" max ->" + Math.max(x,y));
        System.out.println(" sqrt ->" +Math.sqrt(x));
        double p=Math.sqrt(y);
        System.out.println(" p=sqrt(y) -> " +p);
        double k=Math.pow(i,j);
        System.out.println(" k=pow(i,j)-> " +k);
    }
}
```

Вход на данни от клавиатура:

Цели числа:

Метод *nextInt* :

```
import java.util.*;

public class Tester{
    public static void main( String args[] ){

        Scanner kb = new Scanner(System.in);
        System.out.print("Enter your integer here. ");    //enter 3,001
        int i = kb.nextInt( );
        System.out.println(3*i);                            //prints 9003
    }
}
```

Реални числа от тип *double*:

Метод *nextDouble* :

```
import java.util.*;
public class Tester{
    public static void main( String args[] ){

        Scanner kb = new Scanner(System.in);
        System.out.print("Enter your decimal number here. ");
        //1,000.5
        double d = kb.nextDouble( );
        System.out.println( 3*d );           //prints 1001.5
    }
}
```

Текстова информация от тип *String*:

Метод *next* :

```
import java.util.*;
public class Tester{
    public static void main( String args[] ){

        Scanner kb = new Scanner(System.in);
        System.out.print("Enter your String here. ");      //Enter One Two
        String s = kb.next( );                             //inputs up to first white space
        System.out.println( "This is first part of the String,... " + s);
        s = kb.next( );
        System.out.println( "This is next part of the String,... " + s);
    }
}
```

The boolean Type and boolean Operators

Този тип се характеризира с 2 възможни стойности...*true* или *false*.

Само 2 стойности:

Примери:

Нека $x = 3$ и $y = 97$. Какъв ще е резултата от следните булеви изрази ?

($(x < 10) \text{ AND } (y = 97)$) и двете части са *true* , **резултат** : *true*.

($(x < 10) \text{ AND } (y = -3)$) първата част е *true*, втората е *false*, **резултат** : *false*

($(x < 10) \text{ OR } (y = 97)$) и двете са *true*, **резултат** : *true*.

($(x < 10) \text{ OR } (y = -3)$) първата част е *true* , **резултат** : *true*

Правилен синтаксис:

1. За да сравняваме 2 стойности, като ($y = 97$) ние трябва да го напишем по този начин:

$(y == 97)$ Тъй като “=” е оператор за присвояване.

Аналогично $y != 97$ означава: “у не е равно на 97”.

2. В Java не можем да използваме думата “and” за да означим **AND** операция.

Използва се “&&”, напр. :.....($(x < 10) \&\& (y == 97)$)

3. В Java не можем да използваме думата “or” за да означим **OR** операция.

Използва се “||”, напр.($(x < 10) || (y == 97)$)

Оператор за отрицание:

Този оператор е известен като **not** оператор (!).

Това означава, че ние реално казваме **not true** (!true)? ... = **false**.

1. `System.out.println(!true); //false`
2. `System.out.println(!false); //true`
3. `System.out.println(!(3 < 5)); //false`
4. `System.out.println(!(1 == 0)); //true`

Приоритет на операциите:

Редът е следния:

! == != && ||

Пример 1:

```
System.out.println( true || false && false); //true
```

false && false = false.

След това true || false = true.

Example 2:

```
System.out.println( true && false || false); //false
```

true && false = false.

След това false || false = false.


Пример:

```
System.out.println(" Next Example :");  
boolean b=true;  
boolean c=false;  
System.out.println(b&& c);  
System.out.println(b||c);  
System.out.println(!b&&c);  
System.out.println(false&&true);  
System.out.println(false||true);  
System.out.println(!false&&true);  
boolean d=b&&c;  
System.out.println("d=b&&c -->>" + d);  
int k=12;  
System.out.println("k = " + k);  
c=k>2;  
System.out.println("c = " + c);  
b=(!(k<0)&&(k>3));  
System.out.println("b = " + b);
```


Условен оператор *if* –*else*:

Общ вид на оператора:

```
if ( булев израз)
{
    оператори 1;
}
```




```
else
{
    оператори 2 ;
}
```



Съкратен вариант:

```
if ( булев израз)
{
    оператори 1;
}
```



Действие на оператора:

Определя се стойността на булевия израз

Ако е истина се изпълняват оператори 1, намиращи се в частта на *if*.

Ако не е истина се изпълняват оператори 2, намиращи се в частта на *else*.

Ако е истина се изпълняват оператори 1, намиращи се в частта на *if*, ако не е истина, програмата продължава със следващия оператор след *if*.

Условен оператор *if –else*:

Пример :

```
        //Get a grade from the keyboard
Scanner kb = new Scanner(System.in);
System.out.print("What is your grade? ");
int myGrade = kb.nextInt( );
        //Make a decision based on the value of the grade you entered
if (myGrade >= 70)
{
        //Execute code here if the test above is true
System.out.println("Congratulations, you passed.");
}
else
{
        //Execute code here if the test above is false
System.out.println("Better luck next time.");
}
```

3. Съставен оператор(блок) -съвкупност от няколко оператора, написани един след друг и разположени между отваряща и затваряща фигурна скоба.

```
{  
Оператор 1;  
Оператор 2;  
...  
Оператор n;  
}
```

Възможно е съставния оператор да съдържа и само една команда.

Блок от оператори може да се използва навсякъде в програмата, където синтаксисът изисква оператор.

Дефинициите в блока, се отнасят само за него, т.е. Не могат да се използват извън него.

За разлика от другите оператори блокът не завършва с ;

Вложени условни оператори

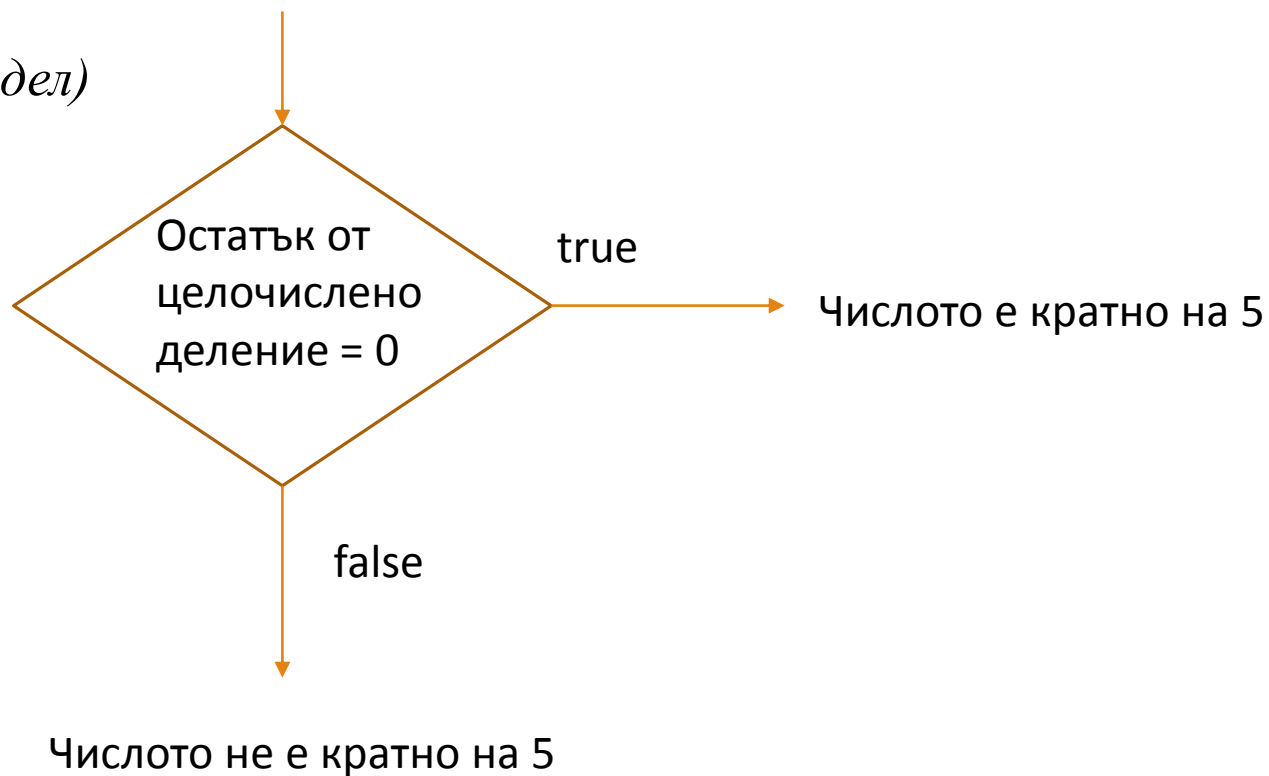
В случая когато на мястото на **оператор1** (или) **оператор2** в конструкцията **if-else** се налага използването на друг условен оператор , тогава говорим за вложени условни оператори.

Синтаксис:

```
if (израз_1)
{
    if (израз_2)
        оператор_1;
    else оператор_2;
}
else
{
    if (израз_3)
        оператор_3;
    else оператор_4;
}
```

Зад. Да се напише програма, която въвежда цяло число от клавиатурата и проверява дали числото е кратно на 5. Да се отпечатаат подходящи съобщения от вида : „числото е/ не е / кратно на 5“.

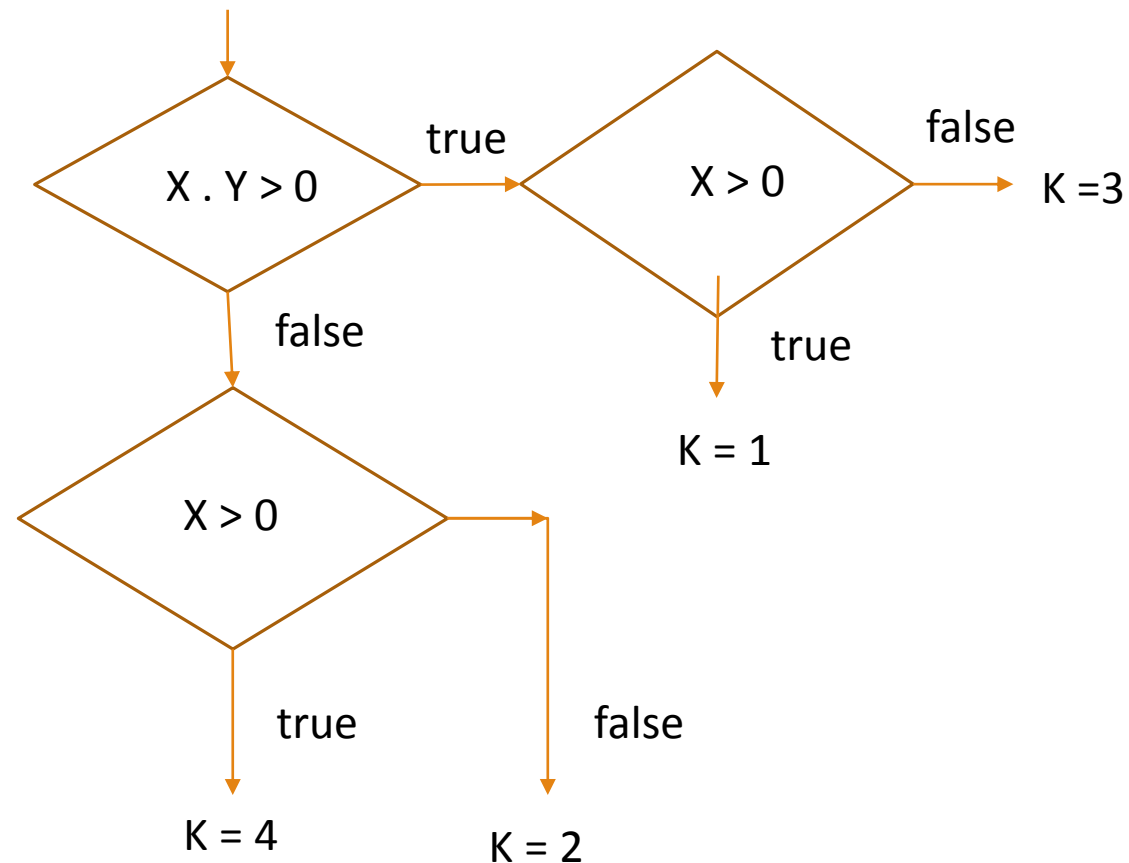
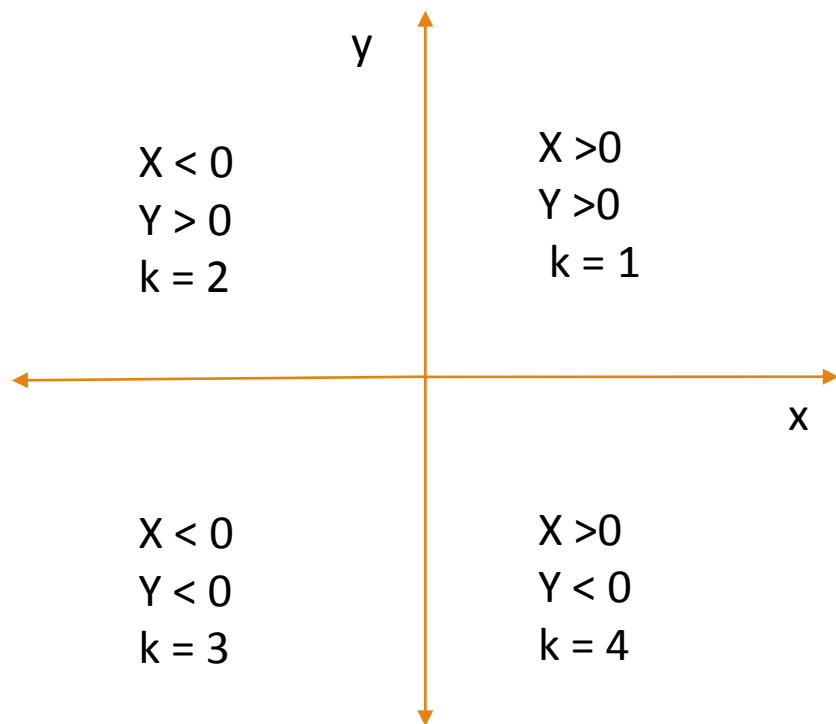
Алгоритъм: (математичен модел)



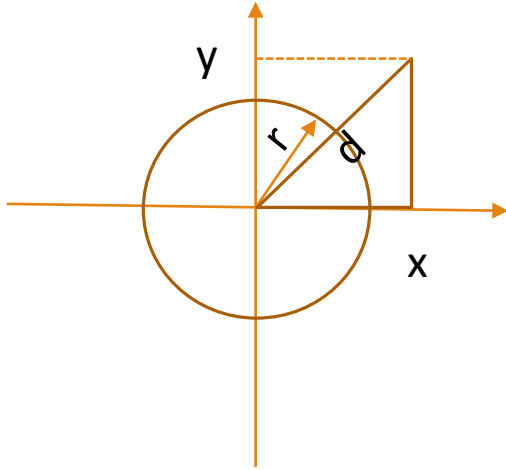
Използва се операцията $\%$,
която ни дава остатъка от
целочислено деление на
въведеното число с числото 5.

Задача : Да се напише програма, която на цялата променлива k присвоява номера на квадранта, в който се намира точка с координати (x, y) . Да се провери дали точката не лежи на координатните оси, т.е. $x.y \neq 0$

Алгоритъм: (математичен модел)



Задача : Да се напише програма, която въвежда точка с координати (x, y) в Декартова координатна система. Въвежда се от клавиатурата и радиуса на окръжност с център (0,0). (фигурата по-долу)
Да се провери къде попада точката : вътре в окръжността, на окръжността или извън нея.



Алгоритъм: (математичен модел)

Като се използва Питагоровата теорема се намира :

$$d = \sqrt{x^2 + y^2}$$

и се сравнява с радиуса на окръжността.

Използват се вложени if-else конструкции от вида:

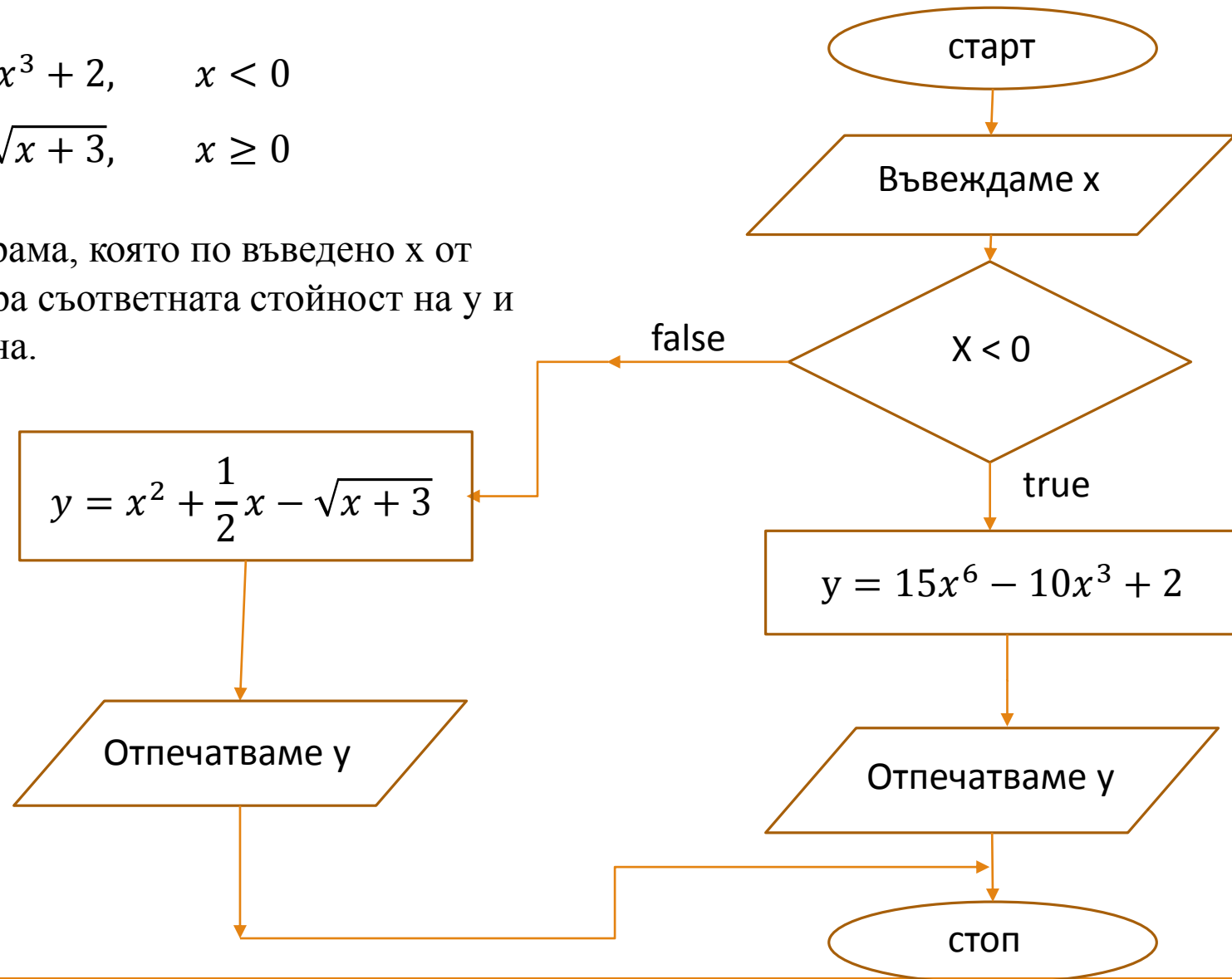
```
if (d < r)
{.....}
else
    if (d == r)
    {.....}
    else
    {.....}
```


Задача : Променливата y зависи от променливата x . Зависимостта е следната:

$$y = \begin{cases} 15x^6 - 10x^3 + 2, & x < 0 \\ x^2 + \frac{1}{2}x - \sqrt{x+3}, & x \geq 0 \end{cases}$$

Да се напише програма, която по въведено x от клавиатурата, намира съответната стойност на y и я отпечатва на екрана.

Алгоритъм:



```
import java.util.Scanner;
public class Example_kvadratno {
    /**
     * намиране корените на квадратно уравнение
     */
    public static void main(String[] args) {
        // въвеждане на коефициентите a, b , c :
        Scanner kbd=new Scanner(System.in);
        System.out.println("Input a:");
        double a=kbd.nextDouble();
        System.out.println("Input b:");
        double b=kbd.nextDouble();
        System.out.println("Input c:");
        double c=kbd.nextDouble();
        double x1,x2;
        double d=b*b-4*a*c;
```

```
    if (d>0)
    {
        x1=((-1)*b + Math.sqrt(d))/(2*a);
        x2=((-1)*b - Math.sqrt(d))/(2*a);
        System.out.println("x1="+x1);
        System.out.println("x2="+x2);
    }
    else
        if(d==0)
        {
            x1=(-1)*b/(2*a);
            System.out.println("x1=x2="+x1);
        }
        else
            System.out.println("Няма решение.");
    }
}
```

```
import java.util.Scanner;
public class Example_menu_if {
/**
 * Използване на меню за основните аритметични операции
 */

    public static void main( String[] args) {
        // меню:
        System.out.println("1.Add.");
        System.out.println("2.Subtraction");
        System.out.println("3.Multiplication");
        System.out.println("4.Division");
        System.out.println("Enter your choice:");
        Scanner kbd = new Scanner(System.in);
        int i = kbd.nextInt();
        System.out.println("Input x=");
        double x= kbd.nextDouble();
```

```
System.out.println("Input y=");  
double y = kbd.nextDouble();  
if (i==1)  
System.out.println("x + y="+ (x + y));  
else  
    if (i==2)  
        System.out.println("x-y="+ (x-y));  
    else  
        if(i==3)  
            System.out.println("x*y="+ (x*y));  
        else  
            if(i==4)  
                System.out.println("x/y="+ (x/y));  
            else  
                System.out.println("choice = 1,2,3 or 4!");  
    }  
}
```

//мишена + 2 играчи=кой е победител ?

```
import java.util.*;
public class Example_ifelse {
public static void main( String [] args) {
Scanner kbd = new Scanner(System.in);
System.out.println("Input x1:");
double x1 = kbd.nextDouble();
System.out.println("Input y1:");
double y1= kbd.nextDouble();
System.out.println("Input x2:");
double x2= kbd.nextDouble();
System.out.println("Input y2:");
double y2= kbd.nextDouble();
double d1= Math.sqrt(x1*x1+y1*y1);
double d2= Math.sqrt(x2*x2+y2*y2);
int r1=5,r2=10,r3=15; int t1=0,t2=0;
if(d1<r1) t1=20;
else
    if(d1<r2) t1=10;
    else
        if (d1<r3) t1=5;
        else t1=0;
```

```
if(d2<r1) t2=20;
else
    if(d2<r2) t2=10;
    else
        if (d2<r3) t2=5;
        else
            t2=0;

System.out.println("t1="+t1);
System.out.println("t2="+t2);
if(t1<t2)
    System.out.println("t1<t2");
else
    if(t1==t2)
        System.out.println("t1=t2");
    else
        System.out.println("t1>t2");
}}
```

Задача:

Да се напише програма, която въвежда 3 реални числа a , b и c и извежда 0, ако не съществува триъгълник със страни a , b и c . Ако такъв триъгълник съществува, да извежда 3, 2 и 1 в зависимост от това, какъв е триъгълникът:

равностранен = 3, равнобедрен = 2 или разностранен = 1.

В задачата да се включат всички необходими проверки за входната информация:

За грешно въведено отрицателно число или 0, тогава a , b или c – не са отсечки.

След направените проверки и определяне вида на триъгълника да се изчисли лицето и периметъра на фигурата, като се използват въведените данни.

Пояснение:

Условие за наличие на фигурата е: сумата от всяка двойка страни да е по-голяма от третата.

За изчисленията да се използват следните формули:

$p = \frac{1}{2}(a + b + c)$ се означава полупериметъра.

$S = \sqrt{p(p - a)(p - b)(p - c)}$ - Херонова формула

Оператор за многовариантен избор: *switch*

Оператор *switch* се използва, когато имаме целочислена променлива, която може да има определен брой стойности.

```
switch (variable_name)
{
    case value_1: Statements;break;
    case value_2: Statements;break;
    case value_3: Statements;break;
    .....
    case value_n: Statements;break;
    default :Statements;
}
```

Опция *default*:

Използва се, когато е възможно стойностите за избора да не са посочени в секциите case.

Оператор *break* :

Този оператор се използва за прекъсване на проверките при срещане на съвпадение на някой от редовете с case и за излизане от оператора switch.

Само *int* (целочислен тип) и *char* (символен тип) са допустими за управляващи променливи:

//Пример 1:

```
int x = 3, p = 5, y = -8;  
switch (x)  
{  
case 2: p++;  
case 3:  
case 4: y += (--p); break;  
case 5: y += (p++);  
}  
System.out.println(y);
```

// -4

//Пример 2

```
int z = 2, q = 0;  
switch (z)  
{  
    case 1: q++;  
    case 2: q++;  
    case 3: q++;  
    case 4: q++;  
    default: q++;  
}  
System.out.println(--q);  
  
//3
```

```
import java.util.*;
public class Choice {
public static void main( String[] args) {
// switch + break – зад. Кой ден от седмицата е въведеното число?
    Scanner kbReader = new Scanner(System.in);
    int k= kbReader.nextInt();
    switch (k)
    {
        case 1: System.out.println("k=1 Понеделник"); break;
        case 2: System.out.println("k=2 Вторник"); break;
        case 3: System.out.println("k=3 Сряда"); break;
        case 4: System.out.println("k=4 Четвъртък"); break;
        case 5: System.out.println("k=5 Петък"); break;
        case 6: System.out.println("k=6 Събота"); break;
        case 7: System.out.println("k=7 Неделя"); break;
        default: System.out.println(„Няма такъв ден , грешно въвеждане!");
    }
}
}
```

//Да се въведе число от 1 до 10 и да се отпечата дали е четно/нечетно:

```
Scanner kbi = new Scanner(System.in);  
int i = kbi.nextInt();  
switch (i)  
{  
    case 1:  
    case 3:  
    case 5:  
    case 7:  
    case 9: System.out.println("Нечетно"); break;  
    case 2:  
    case 4:  
    case 6:  
    case 8:  
    case 10: System.out.println("Четно"); break;  
    default : System.out.println("Не е от 1 до 10");  
}  
System.out.println("i="+i);
```