

Лекция 3.1

Аритметични операции и въведение в управление на логиката на изпълнение. Графичен вход и изход на данни с диалогов прозорец. Форматиране на String.

- Аритметични операции
- Въведение в управление на логиката на изпълнение – оператор if и оператори за сравнение
- Вход и изход на данни в графичен режим
- Преобразуване на низ от цифри в числа
- Форматиране на низ
- Задачи

Аритметични операции

-Означения, наредени по приоритет на изпълнение

1. ***** умножение
2. **/** деление
3. **%** деление по модул
4. **+, -** събиране и изваждане

-Важно : целочислено деление не отчита остатък от деление

$7 / 5$ се пресмята равно на 1

Обаче, при деление с плаваща запетая

$7.0 / 4$ се пресмята равно на 1.4

- Остатък от деление % се получава като деление по модул

$7 \% 5$ се пресмята равно на 2

- **Приоритет на операциите**

-Някои операции се изпълняват преди другите(т.е. умножение преди събиране)

- ИЗПОЛЗВАЙТЕ КРЪГЛИ СКОБИ

- Пример : Намерете средната стойност на a, b и c

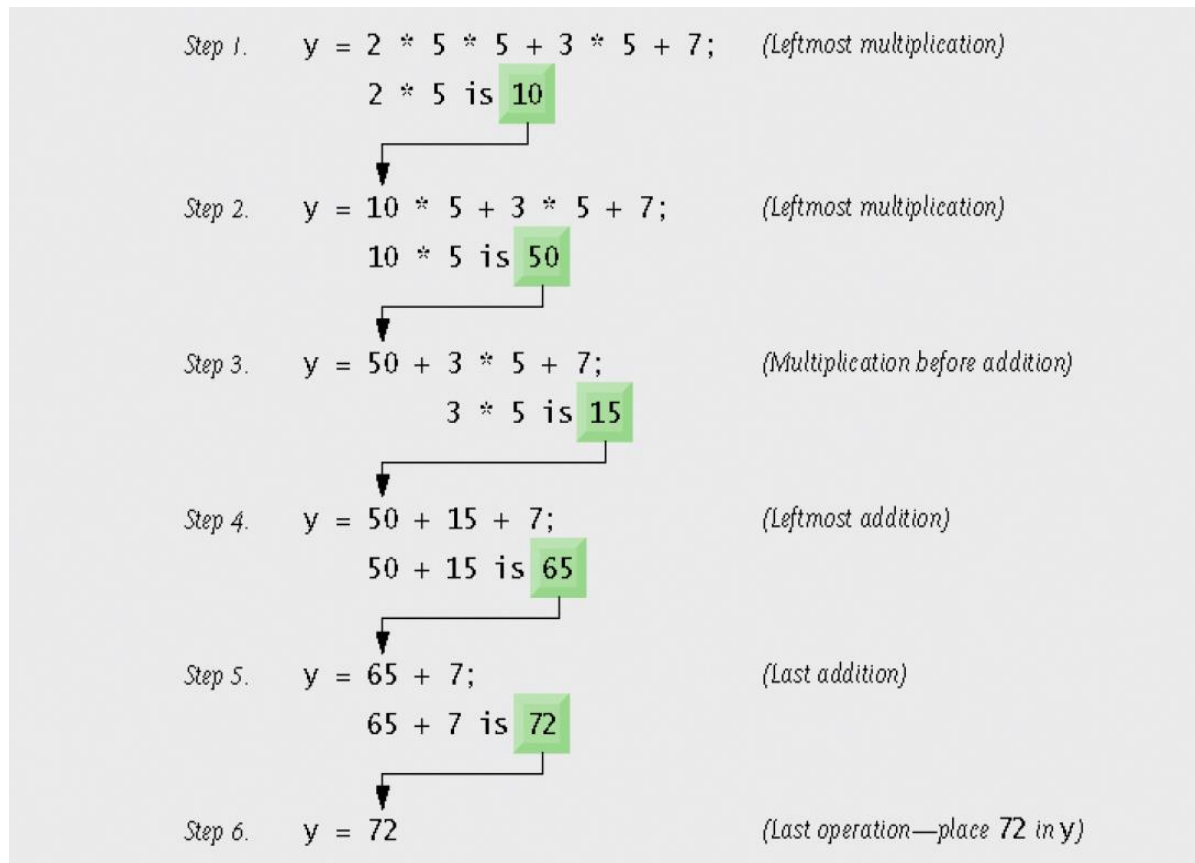
Грешно : $a + b + c / 3$

Правилно : $(a + b + c) / 3$

Правило за добро програмиране :

Използването на скоби при сложни аритметични изрази, прави тези изрази лесни за разчитане и проверка.

Ред на смятане, в който се пресмята даден пример :



Обичайна грешка при програмиране :

Когато двата операнда на аритметична операция са от целочислен тип данни, то и резултатът от аритметичната операция е от целочислен тип.

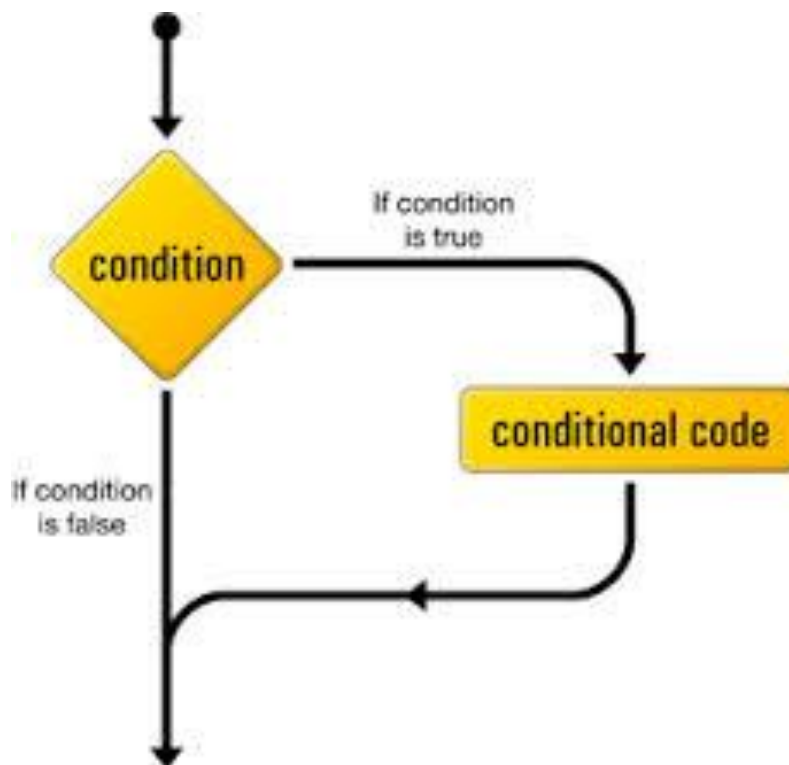
Когато поне един от двата операнда на аритметичната операция е от тип данни с плаваща запетая, то и резултатът от аритметичната операция е от плаваща запетая. **Пример :**

```
double countInt = 3 / 4 ;           // countInt is 0.0
```

```
double countDbl = 3.0 / 4;         // countDbl is 0.75
```

Управление на логиката при изпълнение на команди

- Условие за преход
 - Включва пресмятане на израз ,чийто резултат е **true** или **false**
- if инструкция
 - Тук е разгледана в най-прост вариант
 - Ако условието за преход е **true**,тогава се изпълнява тялото на **if** инструкцията
 - Управлението на програмата винаги продължава в края на **if** инструкцията
 - Условията за преход **if** инструкциите се формират от операции за сравнения или логически изрази



Оператори за равенство и сравнения :

Standard algebraic equality or relational operator	Java equality or relational operator	Sample Java condition	Meaning of Java condition
<i>Equality operators</i>			
=	==	x == y	x е равно на y
≠	!=	x != y	x не е равно на y
<i>Relational operators</i>			
>	>	x > y	x е по-голямо от y
<	<	x < y	x е по-малко от y
≥	>=	x >= y	x е по-голямо или равно на y
≤	<=	x <= y	x е по-малко или равно на y

Comparison.java :

```
1 // Fig. 2.15: Comparison.java
2 // Compare integers using if statements, relational operators
3 // and equality operators.
4 import java.util.Scanner; // program uses class Scanner
5
6 public class Comparison
7 {
8     // main method begins execution of Java application
9     public static void main( String args[] )
10    {
11        // create Scanner to obtain input from command window
12        Scanner input = new Scanner( System.in );
13
14        int number1; // first number to compare
15        int number2; // second number to compare
16
17        System.out.print( "Enter first integer: " ); // prompt
18        number1 = input.nextInt(); // read first number from user
19
20        System.out.print( "Enter second integer: " ); // prompt
21        number2 = input.nextInt(); // read second number from user
22
23        if ( number1 == number2 )
24            System.out.printf( "%d == %d\n", number1, number2 );
25
26        if ( number1 != number2 )
27            System.out.printf( "%d != %d\n", number1, number2 );
28
29        if ( number1 < number2 )
30            System.out.printf( "%d < %d\n", number1, number2 );
```

Outline

Comparison.java:

(1 of 2)

1. class
Comparison

1.1 main

1.2 Declarations

1.3 Input data
(nextInt)

1.4 Compare two
inputs using if
statements

Test за равенство, извежда
резултата с printf.

%d Форматен спецификатор за
целочислени данни

Сравнява две числа с
оператор <.

```
31
32     if ( number1 > number2 )
33         system.out.printf( "%d > %d\n", number1, number2 );
34
35     if ( number1 <= number2 )
36         system.out.printf( "%d <= %d\n", number1, number2 );
37
38     if ( number1 >= number2 )
39         system.out.printf( "%d >= %d\n", number1, number2 );
40
41 } // end method main
42
43 } // end class Comparison
```

Outline

Comparison.java

(2 of 2)

Compares two numbers using relational operator >, <= and >=.

```
Enter first integer: 777
Enter second integer: 777
777 == 777
777 <= 777
777 >= 777
```

Program output

```
Enter first integer: 1000
Enter second integer: 2000
1000 != 2000
1000 < 2000
1000 <= 2000
```

- Ред 6 : започва декларацията на class **Comparison**
- Ред 12 : декларира променлива **input** реферираща **Scanner** и ѝ присвоява **Scanner** обект, който въвежда данни от Стандартен вход
- **input** е пример за променлива от **референтен** тип
- Редове 4-15 : декларират **int** променливи (прост тип данни)
- Методът **nextInt()** на **Scanner** обект служи за въвеждане на следващо цяло число от Стандартен вход.
- Редове 17-18 : подканя потребителя със съобщение(prompt) да въведе първо цяло число и присвоява въведеното число на **number1**
- Редове 20-21 : подканя потребителя със съобщение(prompt) да въведе второто цяло число и присвоява въведеното число на **number2**


```
23     if ( number1 == number2 )
24         System.out.printf( "%d == %d\n", number1, number2 );
```

- Ако променливите са равни (условието е изпълнено)
 - Ред 24 се изпълнява
- Ако променливите **не** са равни, ред 24 се пропуска(не се изпълнява)
- **Няма** точка и запетая в края на if инструкцията

- Редове 26-27, 29-30, 32-33, 35-36 и 38-39

- Сравняваме number1 и number2, съответно посредством операторите **!=, <, >, <= и >=**
Запомнете :
 - %d е форматен спецификатор за целочислени данни
 - %s е форматен спецификатор за String(текстови данни)
 - %f е форматен спецификатор за числа с плаваща запетая
 - Използват се с printf() и String.format() (ще разгледаме след малко)

Обичайна грешка при програмиране :

Пропускането на някоя от фигурните скоби за условието за преход на if инструкцията е **синтактична грешка** –тези скоби не трябва да се пропускат.

Обичайна грешка при програмиране :

Смесването на оператора за равенство, **==**, с оператора за присвояване, **=**, води до логическа и синтактична грешка.

Операторът за равенство се чете като „ е равно на ...” , докато операторът за присвояване се чете като „взима” или „получава стойността на...”

Обичайна грешка при програмиране :

Синтактична грешка е ако операторите ==, !=, >= и <= съдържат шпация(празен символ) между тях като например :
= , != , > = и < =,което е **грешно**.

Обичайна грешка при програмиране :

Пренареждането на символите в операторите !=, >= и <= , като например =! , => и =< , е **синтактична грешка**.

Правило за добро програмиране :

Подравнявайте тялото на if инструкцията, за да си личи къде започва и свършва.Това допринася съществено за проверка на програмата за логически и синтактични грешки.

Извеждане на текст в Dialog Box

Графичен изход на данни

Повечето Java Приложения използват window или dialog box за графично извеждане на данни

Текстовото извеждане – извежда на Command prompt window данни само на стандартен изход

class **JOptionPane** осигурява използване на диалогови прозорци за вход и изход на данни.

Библиотеки на Java – packages

-Съвкупност от предефинирани класове за приложения

Класове, логически обособени в групи се наричат **packages**

-Съвкупността от всички packages се наричат Java class library или Java Applications programming Interface(Java API)

JOptionPane е от javax.swing package(библиотека)

Тази библиотека съдържа класове за работа с графичен потребителски интерфейс **Graphical User Interfaces(GUIs)**

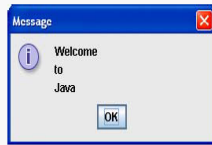
```
1 // Fig. 3.17: Dialog1.java
2 // Printing multiple lines in dialog box.
3 import javax.swing.JOptionPane; // import class JOptionPane
4
5 public class Dialog1
6 {
7     public static void main( String args[] )
8     {
9         // display a dialog with the message
10        JOptionPane.showMessageDialog( null, "Welcome\nto\nJava" );
11        System.exit( 0 ); // terminate application with window
12    } // end main
13 } // end class Dialog1
```

Outline

Dialog1.java

Import class JOptionPane

Покажи този низ в диалогов прозорец



Редове 1-2 : коментари

Две групи библиотеки(packages) на Java API

Базови библиотеки(core packages)

- Имената им започват с java. Например, **java.util**,**java.lang**

- Включени в Java 2 Software Development Kit

Разширения на базовите библиотеки(extension packages)

- Например, започват с **javax**

3-ти ред

import декларации

Използват се от компилатора, за да се означат класовете

външните за приложението класове и да се **намерят техните дефиниции** в съответната Java библиотека

В този пример, указват на компилатора да зареди **class**

JOptionPane от javax.swing библиотеката

Редове 6-11 : празен ред,започва **class Dialog1** and **main()**

10-ти ред :

JOptionPane.showMessageDialog(null, "Welcome\nto\nJava\nProgramming!");

Изпълнява метод **showMessageDialog** от **class JOptionPane**

Методът изисква минимум 2 аргумента

Аргументите се разделят със запетая

Засега,приемаме първият аргумент да е винаги **null**

Втория аргумент представя низа, който искаме да изобразим в диалогивия прозорец

showMessageDialog е **static** метод на **class JOptionPane**

- static** методи се изпълняват като използваме class името,точка(.)

следвана от името на метода

11-ти ред : **System.exit(0) ;**

Изпълнява **static** метода **exit()** на **class System** – **спира изпълнението на приложението.**

Редове 12-13 : Скоби за край на **main()** и **class Dialog1**

Въвеждане на текст в Dialog Box

Диалогов прозорец за въвеждане на текст

Позволява на потребителя да въведе данни по графичен начин

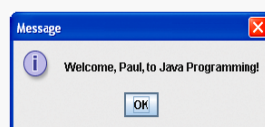
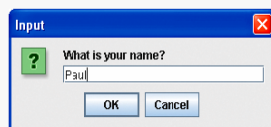
Използва метода **showInputDialog** на **class JOptionPane**

```
2 // Basic input with a dialog box.
3 import javax.swing.JOptionPane;
4
5 public class NameDialog
6 {
7     public static void main( String args[] )
8     {
9         // prompt user to enter name
10        String name =
11            JOptionPane.showInputDialog( "what is your name?" );
12
13        // create the message
14        String message =
15            String.format( "Welcome, %, to Java Programming!", name );
16        // display the message to welcome the user by name
17        JOptionPane.showMessageDialog( null, message );
18        System.exit(0); // terminate the application
19    } // end main
20 } // end class NameDialog
```

Покажи диалогов прозорец за вход и
съобщение до потребителя

NameDialog.java

Форматиране на String
преди показването му в
диалоговия прозорец



Редове 10-11 :

Иползват **showInputDialog** да покажат съобщение и текстово поле, където се въвеждат данните

Метод **showInputDialog** връща String, съдържащ въведеното от потребителя при натискане на OK

Метод **showInputDialog** връща null при натискане на cancel

Редове 14-15

Изпълнява static метода **format()** на **class String**

Синтаксисът на метода **String.format()** е идентичен на **System.out.printf()**

За разлика от **System.out.printf** методът **format()** **връща форматиран String** вместо да го отпечата на Стандартния изход

Въвеждане на цифрови данни с графичен интерфейс

Програма за събиране на числа

- Използва диалогов прозорец за въвеждане на числа
- Използва **преобразуване на текст** в **цифрови данни**
- Използва диалогов прозорец да изобрази сумата на две числа

```
1 // Fig. 2.9: Addition.java
2 // Addition program that displays the sum of two numbers.
3
4 // Java packages
5 import javax.swing.JOptionPane; // program uses JOptionPane
6
7 public class Addition {
8
9     // main method begins execution
10    public static void main( String[] args )
11    {
12        String firstNumber; // first string entered by user
13        String secondNumber; // second string entered by user
14
15        int number1; // first number to add
16        int number2; // second number to add
17        int sum; // sum of the two numbers
18
19        // read in first number from user as a String
20        firstNumber = JOptionPane.showInputDialog( "Enter first integer" );
21
22        // read in second number from user as a String
23        secondNumber =
24            JOptionPane.showInputDialog( "Enter second integer" );
25
26        // convert numbers from type String to type int
27        number1 = Integer.parseInt( firstNumber );
28        number2 = Integer.parseInt( secondNumber );
29
30        // add numbers
31        sum = number1 + number2;
32    }
```

Declare variables: name and type.

Input first integer as a String, assign to firstNumber.

Convert strings to integers.

Add, place result in sum.


```
33      // display result
34      JOptionPane.showMessageDialog( null, "The sum is " + sum,
35      "Results", JOptionPane.PLAIN_MESSAGE );
36
37      System.exit( 0 ); // terminate application with window
38
39  } // end method main
40
41  } // end class Addition
```

```
5  import javax.swing.JOptionPane;
```

- Указва в коя библиотека е class JOptionPane

```
7  public class Addition {
```

- Декларация на public class Addition → файлът с тази програма трябва да бъде **Addition.java**

-Редове 10-11 : main()

```
12      string firstNumber; // first string entered by user
13      string secondNumber; // second string entered by user
```

- Декларации на променливи

-firstNumber и secondNumber са променливи от тип String

- Променливи

Място в паметта, в което съхранява стойност от даден тип

-Трябва да се декларира по име и типм, преди да се използва

-firstNumber и secondNumber са от тип String(package java.lang)

-Тези променливи служат за съхранение на String(низове)

Името на променливата : произволен и **логически обособен** идентификатор

Декларациите завършват с ;

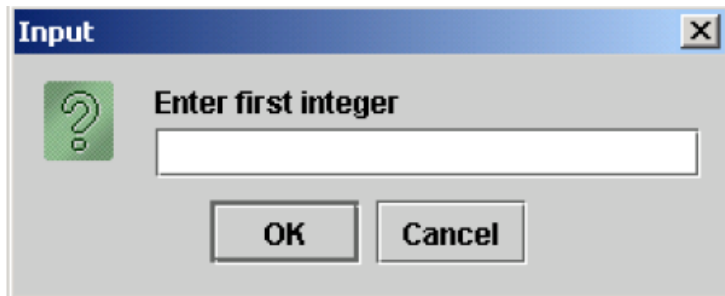
```
15      int number1;           // first number to add
16      int number2;           // second number to add
17      int sum;                // sum of number1 and number2
```

- Декларира се променливи number1,number2, и съм от тип инт (цяло число 32 бита)
int служи за съхраняване на целочислени стойности : 9, 5, -4, 95

```
20      firstNumber = JOptionPane.showInputDialog( "Enter first integer" );
```

- Прочита String,чрез метода **showInputDialog**,който извикваме със **String** указващ **съобщението до потребителя** т.е. чрез текста „Enter first integer:” подканваме потребителя да въведе число.

Метода изобразява следното :



При въвеждане на погрешен тип данни(различен от цяло число) или натискане на Cancel,възниква грешка – преобразуване на данните грешно

-След изпълнението на метода(при натискане на Cancel или OK) – резултатът от изпълнението на showInputDialog се присвоява на firstNumber с оператора = в String формат(текстов формат).

= е бинарен оператор,използва два операнда

- Изразът от дясно се изчислява и после се присвоява на променливата от ляво на оператора.

Чете се като : firstNumber получава стойността на JOptionPane.showInputDialog(“Enter first integer”)

```
23      secondNumber =
24      JOptionPane.showInputDialog( "Enter second integer" );
```

-Идентично на предишната команда,разликата е,че въвеждаме второто число.

-Инициализира secondNumber на второто въведено число в **String формат**.

```
27     number1 = Integer.parseInt( firstNumber );
28     number2 = Integer.parseInt( secondNumber );
```

27-ми ред - метод **Integer.parseInt()**

- Преобразува String аргумент в целочислена тип данна (type int)
-class Integer от библиотека java.lang
- Цялото число върнато от метода Integer.parseInt() се присвоява на променливата number1

28-ми ред – аналогично за променливата number2, в която въвеждаме второто число, което ще сумираме.

```
31     sum = number1 + number2;
```

- Команда за присвояване, пресмята сбора на number1 и number2 и присвоява резултата на променливата sum

Чете се : **sum** получава стойността на **number1 + number2**

```
34     JOptionPane.showMessageDialog( null, "The sum is " + sum,
35     "Results", JOptionPane.PLAIN_MESSAGE );
```

- Използва **showMessageDialog()** да изведе резултата **"The sum is " + sum**

- Използва оператора + да събере(добави) низа "The sum is" и **целочислената стойност** на променливата **sum**.

- Резултатът е нов низ(заделя се ново място в паметта)

- променливата sum първо се преобразува до String И после се събира с низа отляво

-Забележете празния символ в края на низа "The sum is "

Друга версия на JOptionPane.showMessageDialog :

```
JOptionPane.showMessageDialog( null, "The sum is " + sum,
    "Results", JOptionPane.PLAIN_MESSAGE );
```

- Изисква 4 аргумента(вместо 2 както по-рано)

- **Първият** аргумент е null засега

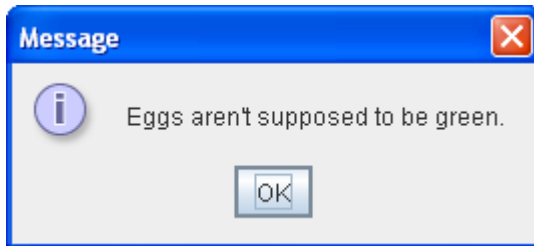
- **Вторият**: низът, който искаме да изведем графично

- **Тетият** : низът, който е в заглавието на диалоговия прозорец

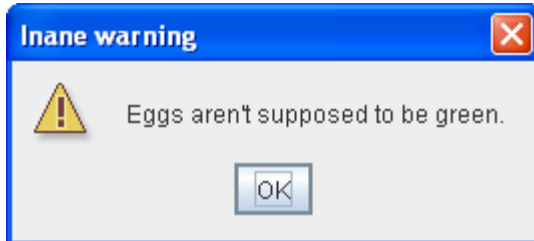
- **Четвъртият** : типът на диалоговия прозорец и иконка.

- Новият пример не указва да се използва иконка, то се задава от константата

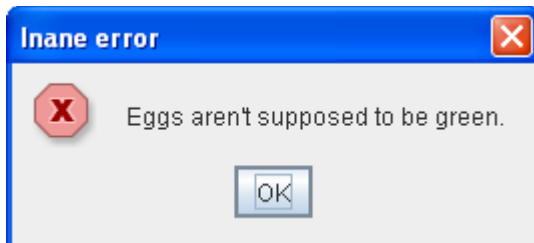
JOptionPane.PLAIN_MESSAGE



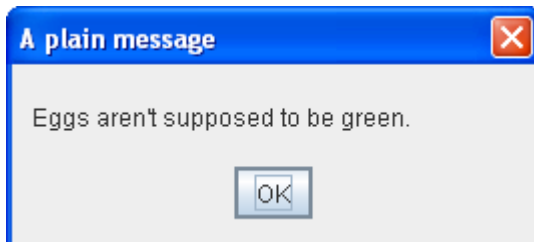
```
//default title and icon  
JOptionPane.showMessageDialog( null,  
    "Eggs are not supposed to be  
green.");
```



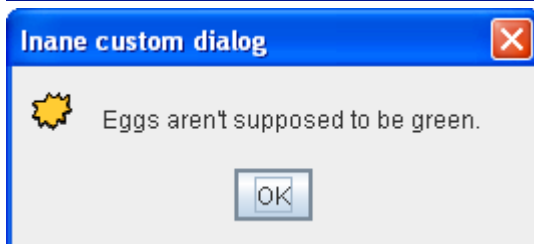
```
//custom title, warning icon  
JOptionPane.showMessageDialog( null,  
    "Eggs are not supposed to be green.",  
    "Inane warning",  
    JOptionPane.WARNING_MESSAGE);
```



```
//custom title, error icon  
JOptionPane.showMessageDialog( null,  
    "Eggs are not supposed to be green.",  
    "Inane error",  
    JOptionPane.ERROR_MESSAGE);
```







```
//custom title, no icon  
JOptionPane.showMessageDialog( null,  
    "Eggs are not supposed to be green.",  
    "A plain message",  
    JOptionPane.PLAIN_MESSAGE);
```



```
//custom title, custom icon  
JOptionPane.showMessageDialog( null,  
    "Eggs are not supposed to be green.",  
    "Inane custom dialog",  
    JOptionPane.INFORMATION_MESSAGE,  
    icon);
```

JOptionPane константи за типове диалогови прозорци :

Тип на диалогов прозорец	иконка	Описание
<code>JOptionPane.ERROR_MESSAGE</code>		Изобразява диалог за означаване на грешка на потребителя.
<code>JOptionPane.INFORMATION_MESSAGE</code>		Изобразява диалог с информация до потребителя. Потребителят може да игнорира диалога.
<code>JOptionPane.WARNING_MESSAGE</code>		Изобразява диалог предупреждаващ за евентуален проблем.
<code>JOptionPane.QUESTION_MESSAGE</code>		Изобразява диалог с въпрос към потребителя. Изисква отговор от потребителя с натискане на Yes или No бутон.
<code>JOptionPane.PLAIN_MESSAGE</code>	no icon	Изобразява диалог със съобщение, но без иконка- обикновен диалогов прозорец.

Продължение на **Comparison.java** :

```
1 // Fig. 2.20: Comparison.java
2 // Compare integers using if statements, relational operators
3 // and equality operators.
4
5 // Java packages
6 import javax.swing.JOptionPane;
7
8 public class Comparison {
9
10 // main method begins execution of Java application
11 public static void main( String args[] )
12 {
13     String firstNumber; // first string entered by user
14     String secondNumber; // second string entered by user
15     String result; // a string containing the output
16
17     int number1; // first number to compare
18     int number2; // second number to compare
19
20 // read first number from user as a string
21 firstNumber = JOptionPane.showInputDialog( "Enter first integer:" );
22
23 // read second number from user as a string
24 secondNumber =
25     JOptionPane.showInputDialog( "Enter second integer:" );
26
27 // convert numbers from type String to type int
28 number1 = Integer.parseInt( firstNumber );
29 number2 = Integer.parseInt( secondNumber );
30
31 // initialize result to empty String
32 result = "";
33
```

Променлива от референтен тип като String трябва да се инициализира, **преди** да се използва!

[Outline](#)

```

34     if ( number1 == number2 )
35         result = result + number1 + " == " + number2;
36
37     if ( number1 != number2 )
38         result = result + number1 + " != " + number2;
39
40     if ( number1 < number2 )
41         result = result + "\n" + number1 + " < " + number2;
42
43     if ( number1 > number2 )
44         result = result + "\n" + number1 + " > " + number2;
45
46     if ( number1 <= number2 )
47         result = result + "\n" + number1 + " <= " + number2;
48
49     if ( number1 >= number2 )
50         result = result + "\n" + number1 + " >= " + number2;
51
52     // Display results
53     JOptionPane.showMessageDialog( null, result, "Comparison Results",
54         JOptionPane.INFORMATION_MESSAGE );
55
56     system.exit( 0 ); // terminate application
57
58 } // end method main
59
60 } // end class Comparison
  
```

Test for equality, create new string, assign to result.

Notice use of
JOptionPane.INFORMATION_MESSAGE

Comparison.java

 3. if statements
 4. showMessageDialog

Редове 1-12 : коментари,import,декларация на class **Comparison** и **main()**

Редове 13-15 : деклариране на променливи,разделение със запетая

```

13     String firstNumber,
14         secondNumber,
15         result;
  
```

Редове 17-18 деклариране на променливи на отделен ред.

Редове 21-30 : въвеждане на числа в текстов формат и **преобразуването им**
целочислени променливи

```

20     // read first number from user as a string
21     firstNumber = JOptionPane.showInputDialog( "Enter first integer:" );
22
23     // read second number from user as a string
24     secondNumber =
25         JOptionPane.showInputDialog( "Enter second integer:" );
26
27     // convert numbers from type String to type int
28     number1 = Integer.parseInt( firstNumber );
29     number2 = Integer.parseInt( secondNumber );
  
```

32-ри ред

-Инициализираме **result** с **празен низ** от символи

-result е променлива от референтен тип и трябва да се инициализира преди да се използва

```
34         if ( number1 == number2 )  
35             result = result + number1 + " == " + number2;
```

-if команда за проверка на **равенство**(==)

- Ако променливите са равни(резултатът от условието е **true**)
 - result се събира с оператора +
result = result + other strings
 - Дясната страна се пресмята първа ,полученият String се присвоява на **result**
 - Ако променливите не са равни,това присвояване се пропуска

- Редове 37-50 : други,аналогични if команди за проверки...

-Редове 53-54 : result се извежда в диалогов прозорец посредством
showMessageDialog()

Задачи :

1. Напишете програма(конзолно приложение), което :

- Въвежда цяло петцифрено десетично число

- Отделя всъчка от цифрите на числото и ги отпечатва на един ред,

а) **В същата последователност,разделени със запетая и празен символ**

б) **В обратната последователност,разделени със запетая и празен символ**

Предполагаме,че потребителя въвежда само петцифрено десетично число.

2. Напишете програма(конзолно приложение), което прочита последователно 5 цели числа от клавиатурата и извежда колко от тези числа са били :

а) **положителни**

б) **отрицателни**

в) **нула**

Забележка: Да се използват само средства на езика,които са предадени до този момент на лекции т.е. без цикли.

3. Напишете програма(конзолно приложение) ,което да преобразува от стойности от **km/h** скорост в **miles/h** еквивалентна стойност, като използва диалогов прозорец за вход и изход на данните

- чете от диалогов прозорец цяло число,което е километра в час (например 120,60,80 и тн) – използвайте подходящо съобщение към потребителя

- пресмята **miles/h** от зададените километри по следната формула :

1	=	0.621371
Km/hour		Miles/hour

- извежда в диалогов прозорец получения резултат