DEGREE & BRANCH : BTech(CSE)

19CSE204 **Object Oriented Paradigm**
Lab # 2
Date : 10-Aug-2020

Instructions if any for students:

(1) **Students have to try the questions and submit back in a word doc with control statements, looping statements,reading input, math functions,string functions,Arrays**
(2) **Practice Questions also has to be completed and submitted by 18-Aug-2020**
(3) **Students have to try this with HPoJ Tool and keep the ouput**

Lab #2

- **At the end of the lab students will be able to write java programs with control structure, looping statement, string and math functions**
- **One Dimensional Array and Two Dimensional Array**
- **Multidimensional Array**
- **Students will have to submit the programs with output**

Topics to be covered

a. **Expression**
b. **Control statement**
c. **Looping Statement**
d. **Math Function**
e. **String Function**

1.Expression

An *expression* is a construct made up of variables, operators, and method invocations, which are constructed according to the syntax of the language, that evaluates to a single value.

Example:

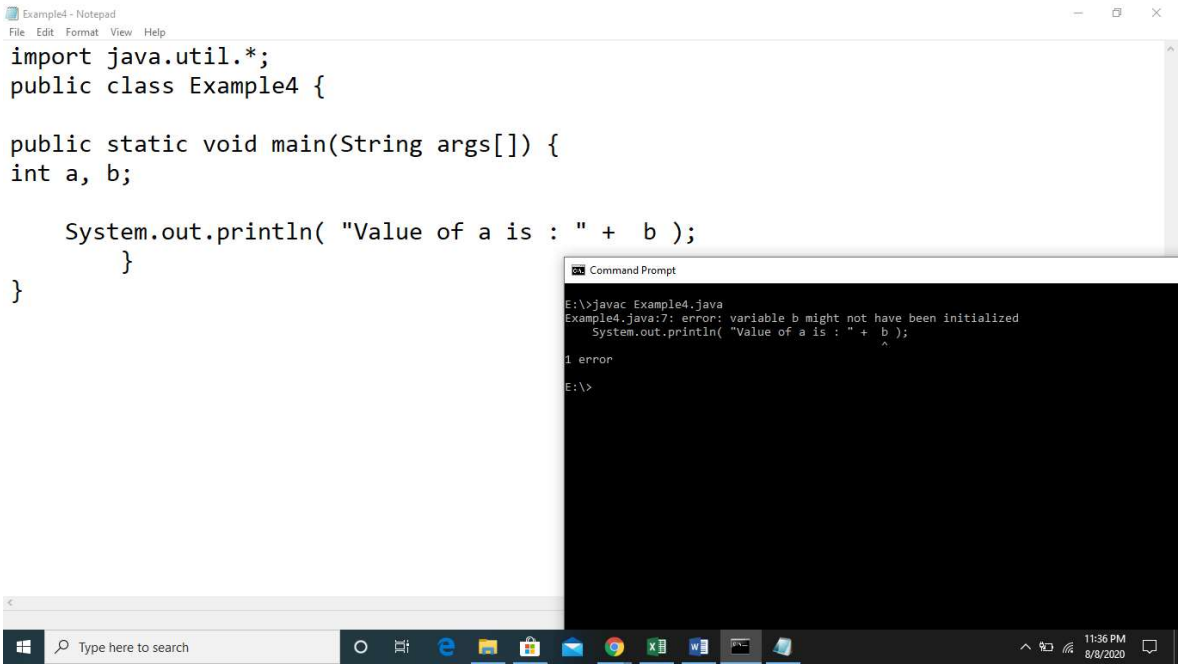
C=(a+b);

Example :

int a;

System.out.println(a);

Example :



Note : All variables in java need to be initialized

Statements are roughly equivalent to sentences in natural languages. A *statement* forms a complete unit of execution. The following types of expressions can be made into a statement by terminating the expression with a semicolon (;).

- Assignment expressions
- Any use of ++ or --
- Method invocations
- Object creation expressions

Such statements are called *expression statements*. Here are some examples of expression statements.

```
// assignment statement
aValue = 8933.234;
// increment statement
aValue++;
```

```
// method invocation statement
System.out.println("Hello World!");

// object creation statement
ClassName Object=new ClassName();
Bicycle myBike = new Bicycle();
```

In addition to expression statements, there are two other kinds of statements: *declaration statements* and *control flow statements*. A *declaration statement* declares a variable. You've seen many examples of declaration statements already:

```
// declaration statement
double aValue = 8933.234;
```

2.Blocks

A *block* is a group of zero or more statements between balanced braces and can be used anywhere a single statement is allowed. The following example, BlockDemo, illustrates the use of blocks:

```
class BlockDemo {
    public static void main(String[] args) {
        boolean condition = true;
        if (condition) { // begin block 1
            System.out.println("Condition is true.");
        } // end block one
        else { // begin block 2
            System.out.println("Condition is false.");
        } // end block 2
    }
}
```

<u>Example</u>

- aValue = 8933.234;
- aValue++;
- System.out.println("Hello World!");
- Bicycle myBike = new Bicycle();

3.Conditional Statements

a. If statement

The Java *if statement* is used to test the condition. It checks boolean condition: *true* or *false*. There are various types of if statement in Java.

- if statement

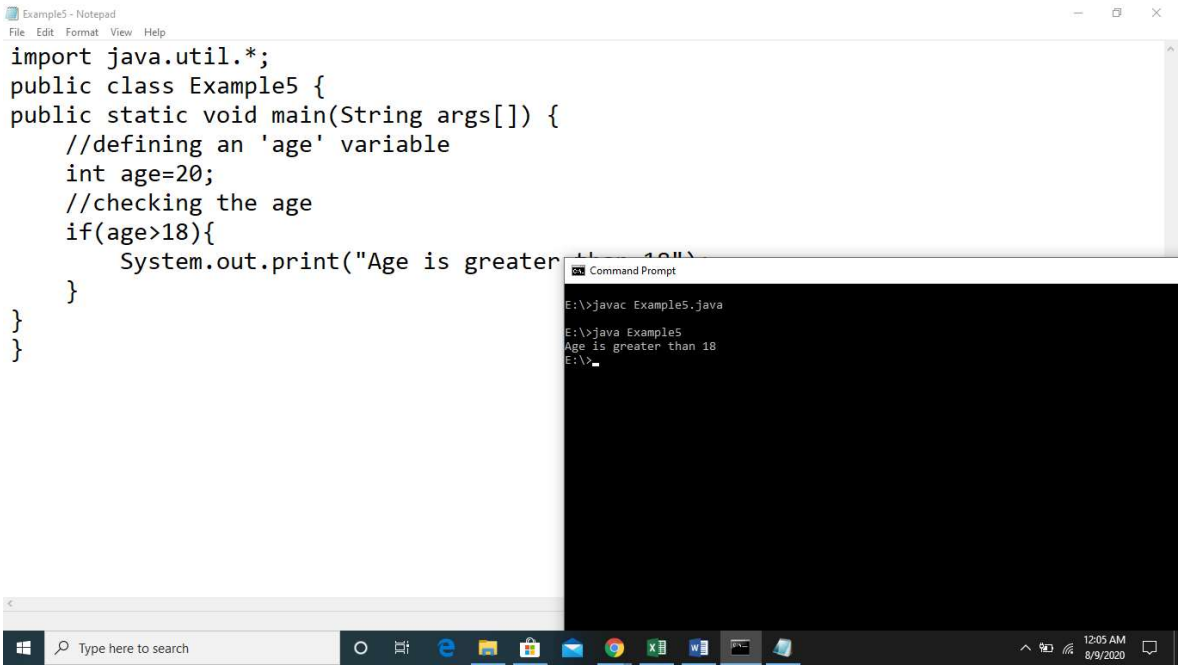
- if-else statement
- if-else-if ladder
- nested if statement

if statement

```
if(condition){
//code to be executed
}
```

Example:

```
import java.util.*;
public class Example5 {
public static void main(String args[]) {
   //defining an 'age' variable
   int age=20;
   //checking the age
   if(age>18){
      System.out.print("Age is greater than 18");
   }
}
}
```

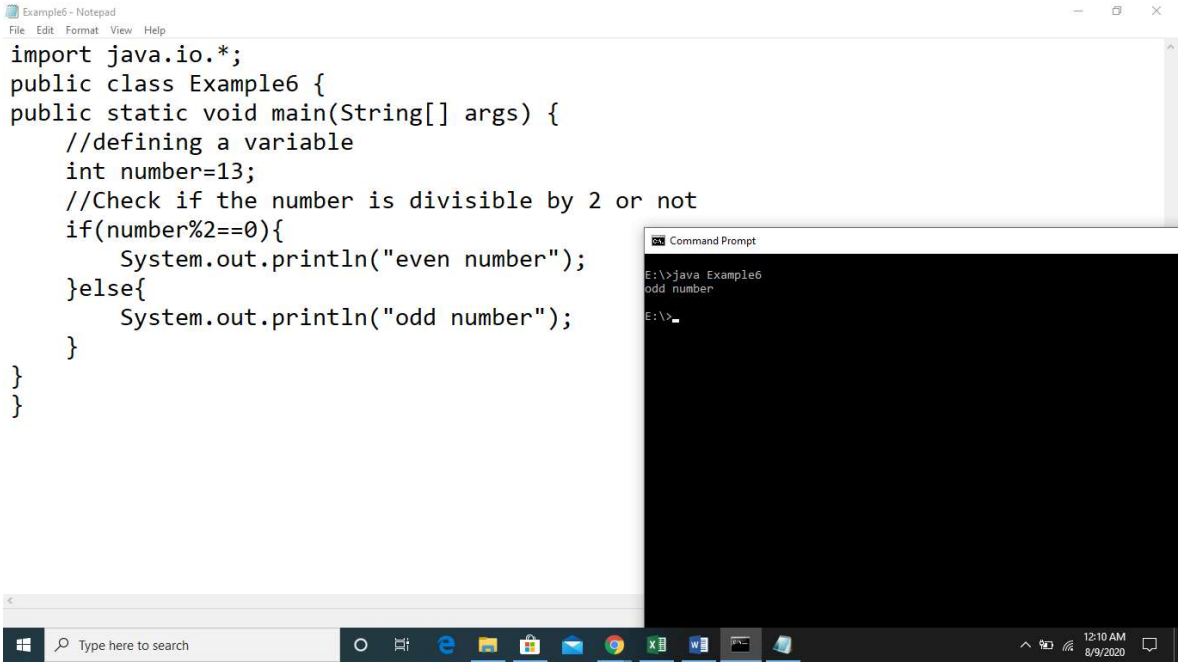


```
if(condition){
//code if condition is true
}else{
//code if condition is false
}
```

Example:

```
public class Example6 {
public static void main(String[] args) {
   //defining a variable
   int number=13;
   //Check if the number is divisible by 2 or not
   if(number%2==0){
      System.out.println("even number");
   }else{
      System.out.println("odd number");
   }
}
}
```

Output:

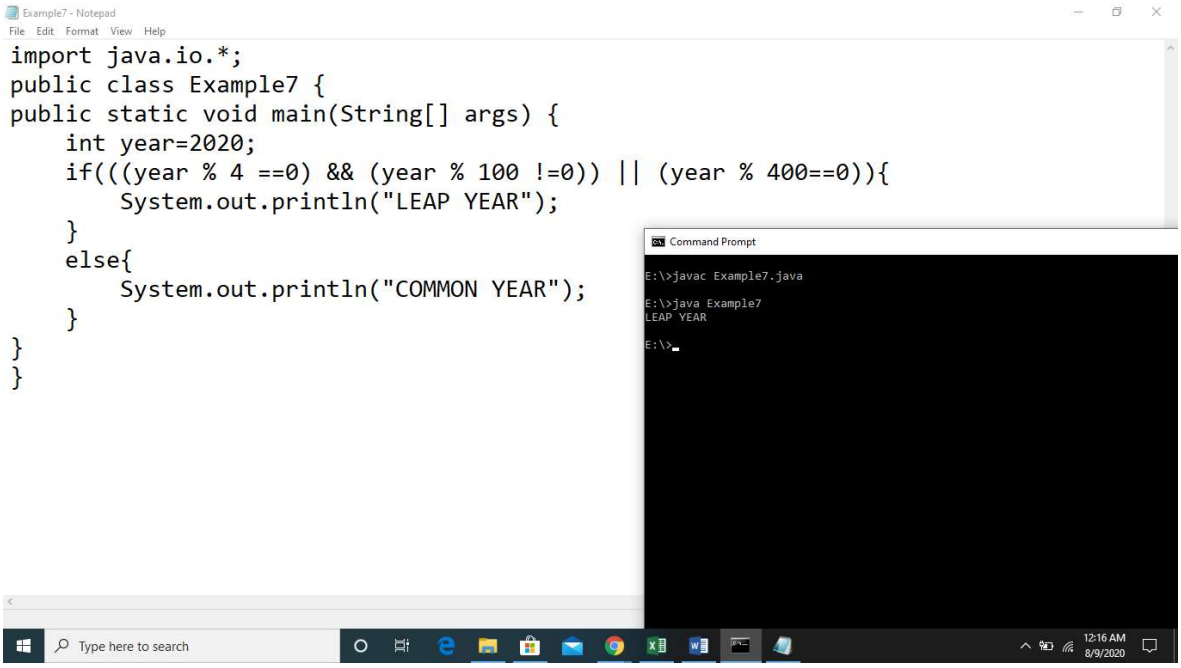


Example

```
public class LeapYearExample {
```

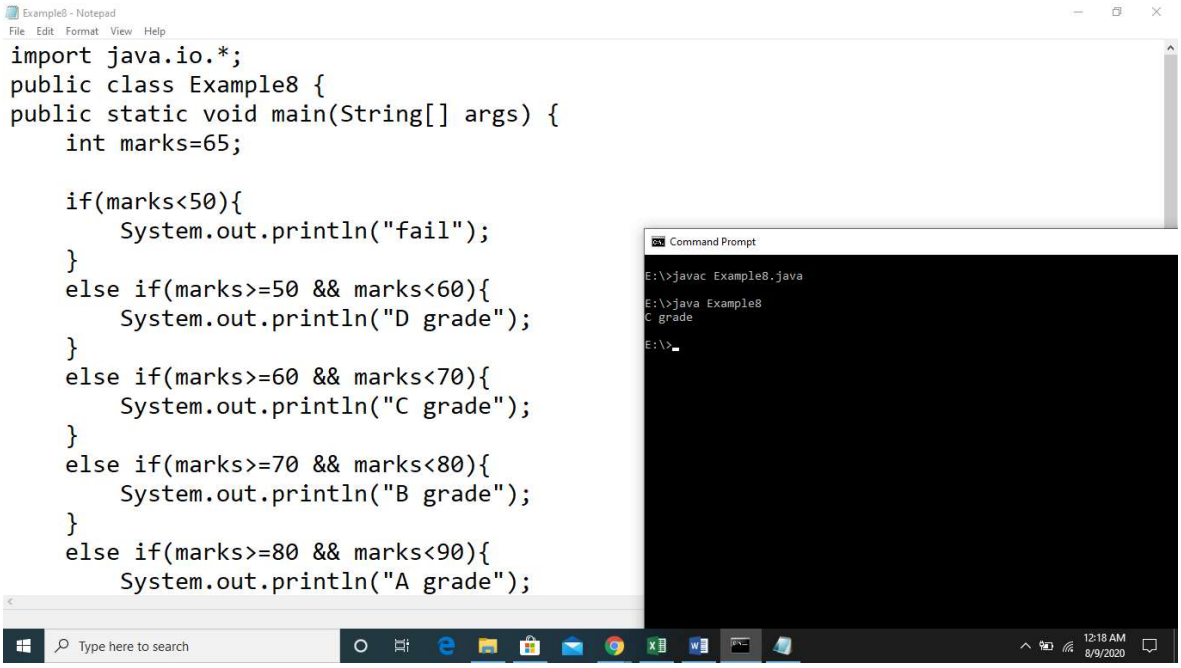
```
public static void main(String[] args) {
   int year=2020;
   if(((year % 4 ==0) && (year % 100 !=0)) || (year % 400==0)){
      System.out.println("LEAP YEAR");
   }
   else{
      System.out.println("COMMON YEAR");
   }
}
}
```

Output



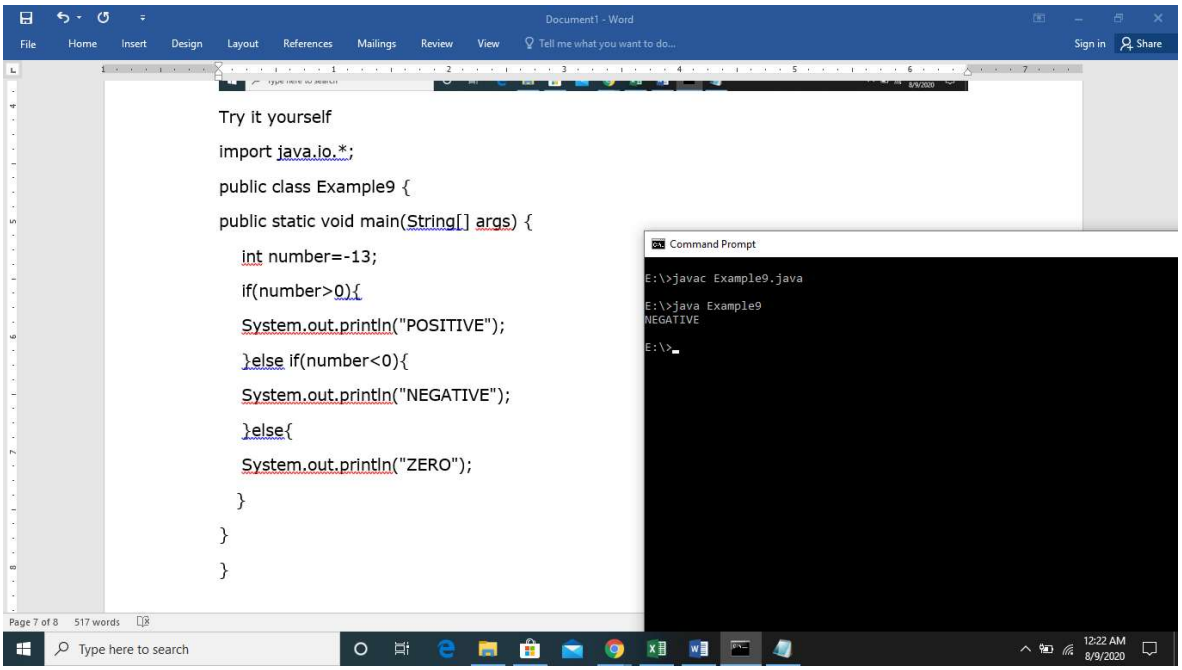
If..Else…if ladder

```
if(condition1){
//code to be executed if condition1 is true
}else if(condition2){
//code to be executed if condition2 is true
}
else if(condition3){
//code to be executed if condition3 is true
}
...
else{
//code to be executed if all the conditions are false
}
```



Try it yourself

```
import java.io.*;
public class Example9 {
public static void main(String[] args) {
   int number=-13;
   if(number>0){
   System.out.println("POSITIVE");
   }else if(number<0){
   System.out.println("NEGATIVE");
   }else{
   System.out.println("ZERO");
  }
}
}
```



Nested If statement

The nested if statement represents the *if block within another if block*. Here, the inner if block condition executes only when outer if block condition is true.

Syntax:

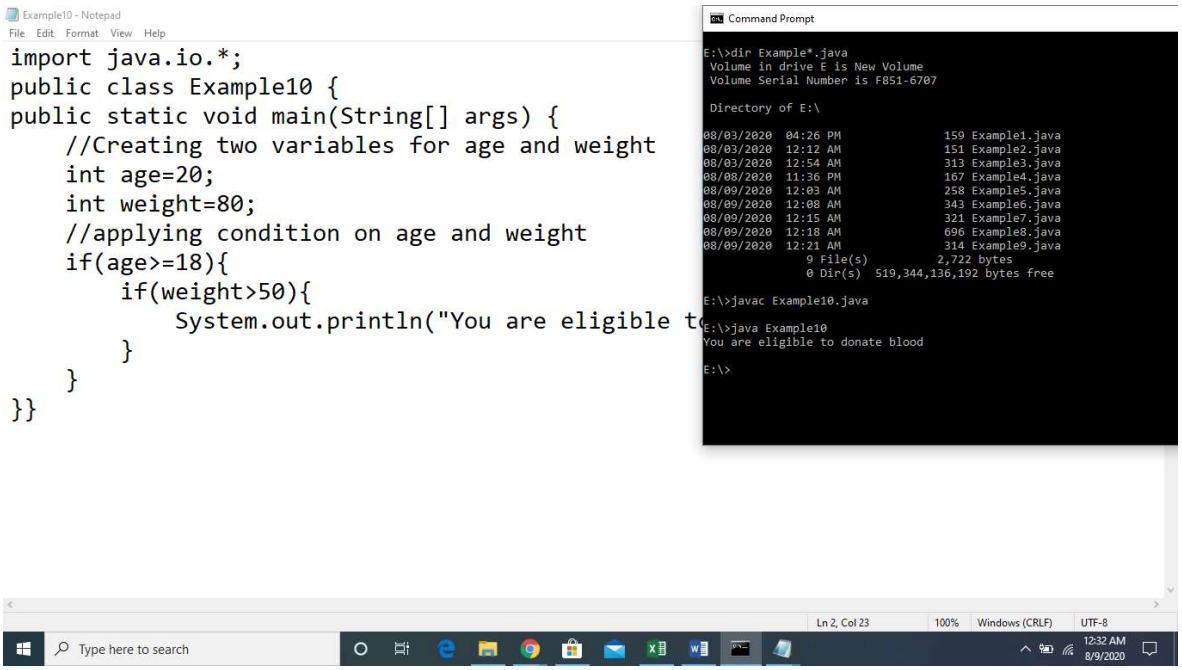
```
if(condition){
    //code to be executed
        if(condition){
          //code to be executed
   }
}
```

Example

```
public class Example10 {
public static void main(String[] args) {
   //Creating two variables for age and weight
   int age=20;
   int weight=80;
   //applying condition on age and weight
   if(age>=18){
      if(weight>50){
         System.out.println("You are eligible to donate blood");
      }
   }
```

}}

Output



Try yourself

```
public class Example11 {
public static void main(String[] args) {
   //Creating two variables for age and weight
   int age=25;
   int weight=48;
   //applying condition on age and weight
   if(age>=18){
      if(weight>50){
         System.out.println("You are eligible to donate blood");
      } else{
         System.out.println("You are not eligible to donate blood");
      }
   } else{
     System.out.println("Age must be greater than 18");
   }
} }
```

Ouput

Example12 - Notepad

File Edit Format View Help

```
import java.io.*;
public class Example11{
public static void main(String[] args) {
    //Creating two variables for age and weight
    int age=25;
    int weight=48;
    //applying condition on age and weight
    if(age>=18){
        if(weight>50){
            System.out.println("You are eligible t
        } else{
            System.out.println("You are not eligib
        }
    } else{
      System.out.println("Age must be greater than 18");
    }
}  }
```

Command Prompt

```
E:\>java Example10
You are eligible to donate blood

E:\>
```

Ln 1, Col 18 100% Windows (CRLF) UTF-8

Type here to search 12:35 AM 8/9/2020

Example

Example13 - Notepad

File Edit Format View Help

```
import java.io.*;
public class Example13{
public static void main(String args[]) {
    //Creating two variables for age and weight
    int age=25;
    int weight=48;
    //applying condition on age and weight
    if(age>=18){
        if(weight>50){
            System.out.println("You are eligible t
        } else{
            System.out.println("You are not eligib
        }
    } else{
      System.out.println("Age must be greater than 18");
    }
}  }
```

Command Prompt

```
E:\>javac Example13.java

E:\>java Example13
You are not eligible to donate blood

E:\>
```

Ln 2, Col 23 100% Windows (CRLF) UTF-8

Type here to search 12:38 AM 8/9/2020

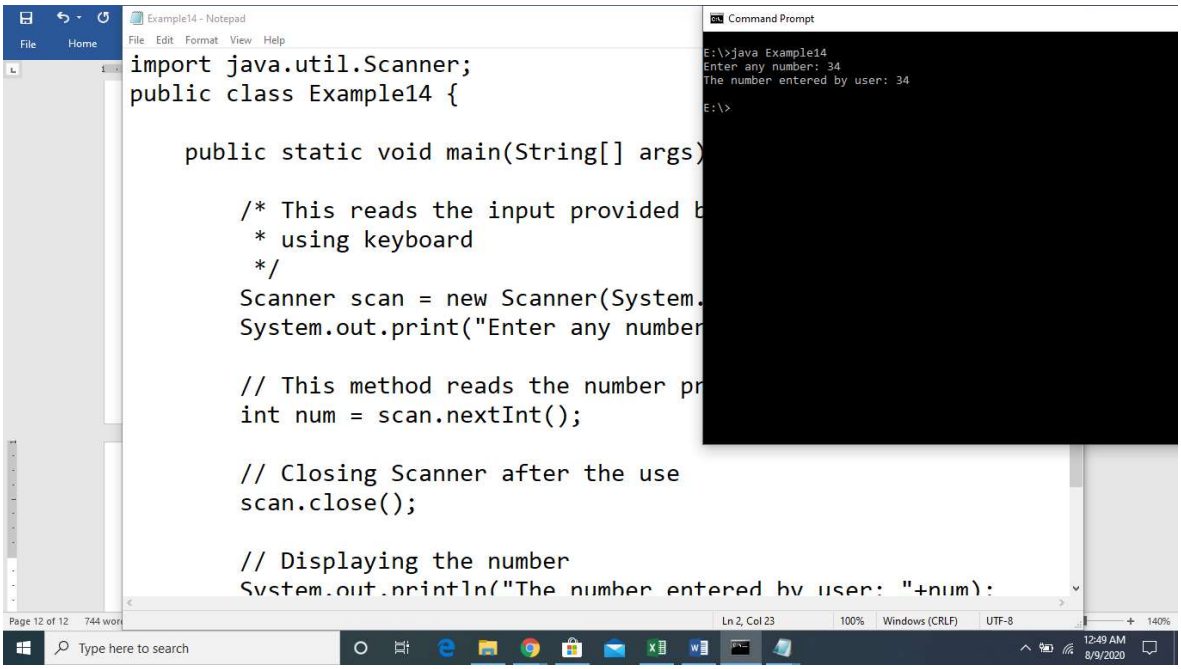
Just check the main parameters. This is also allowed in java

Reading Input from User

Package : java.util.Scanner

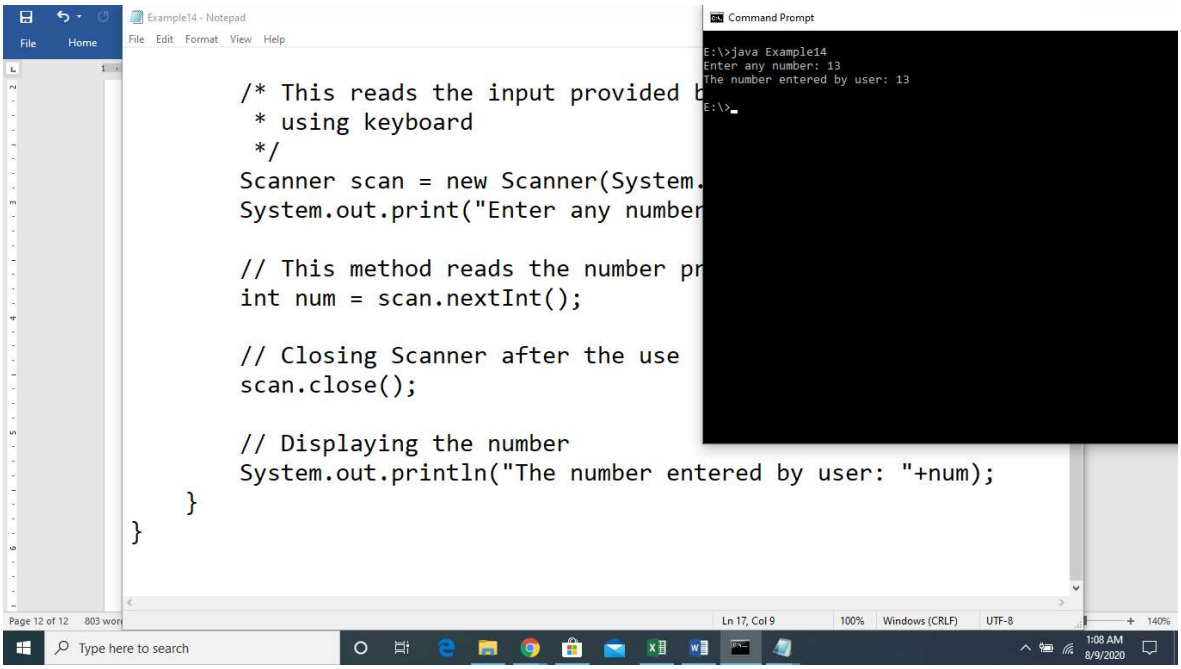
Example

```
import java.util.Scanner;

public class Example14 {

   public static void main(String[] args) {

      /* This reads the input provided by user
       * using keyboard
       */
      Scanner scan = new Scanner(System.in);
      System.out.print("Enter any number: ");

      // This method reads the number provided using keyboard
      int num = scan.nextInt();

      // Closing Scanner after the use
      scan.close();

      // Displaying the number
      System.out.println("The number entered by user: "+num);
   }
}
```



Note : The Java Scanner class provides nextXXX() methods to return the type of value such as nextInt(), nextByte(), nextShort(), next(), nextLine(), nextDouble(), nextFloat(), nextBoolean(),

To get the instance of Java Scanner which reads input from the user, we need to pass the input stream (System.in) in the constructor of Scanner class. For Example:



4.<u>Looping Statements</u>

a. while
b. do..while
c. for statement

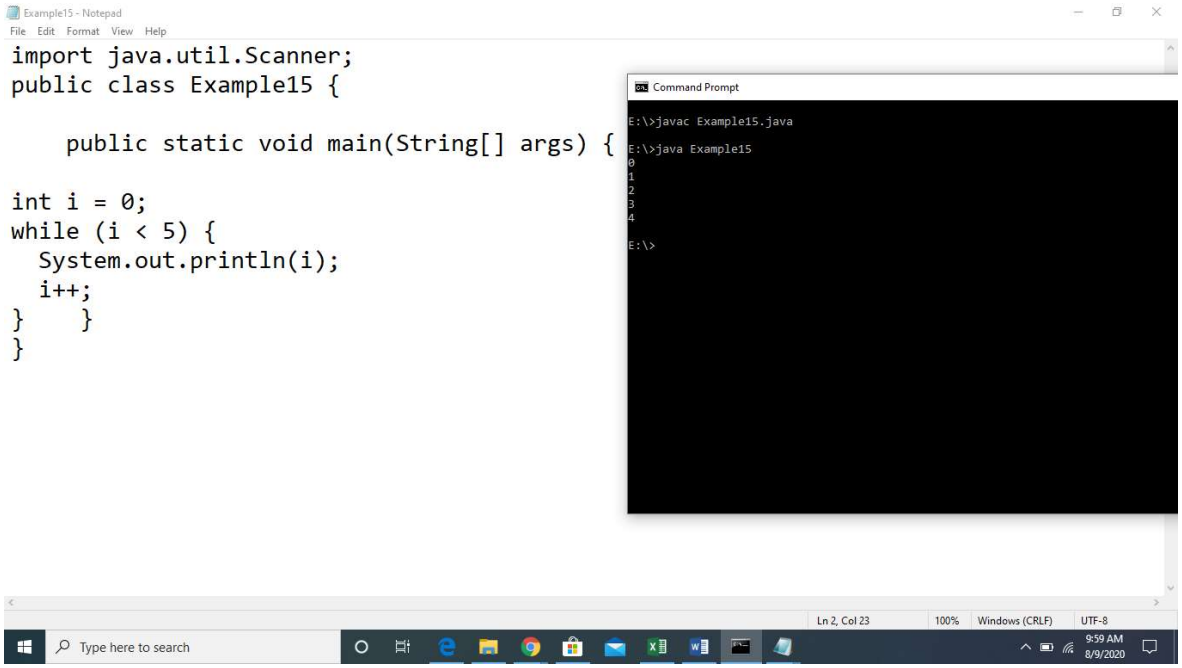
a. While loop

The while loop loops through a block of code as long as a specified condition is true:

Syntax:

```
while (condition) {
  // code block to be executed
}
```

<u>Output</u>



Try it yourself:

(i) Change as i>5 and check the output

(ii)
```
int i=5;
while(i>=5)
{
System.out.println(i);
i--;
}
```

(iii) Declare two variables i,j

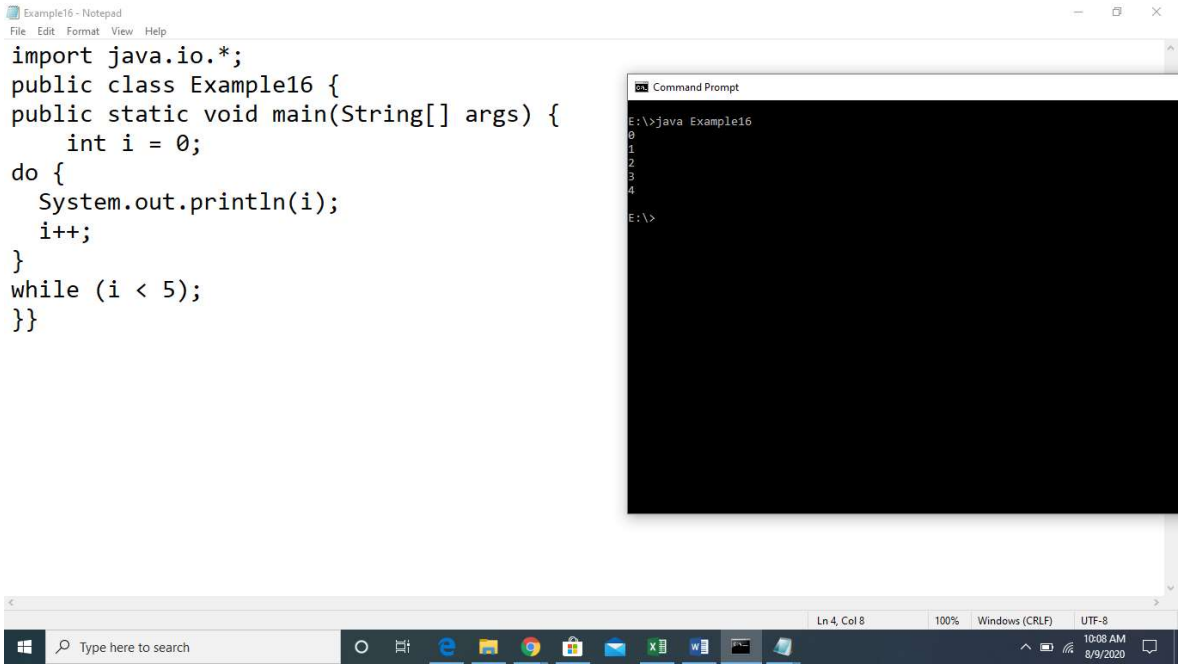
```
int i=10;
int j=10;
while(i<20)
{
while(j<20)
{
System.out.println(i);
I++;
}
System.out.println(j);
j=j+1;
}
```

Do while loop

The do/while loop is a variant of the while loop. This loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true.

```
do {
  // code block to be executed
}
while (condition);
```

Output



Try it yourself

(i)

```
import java.io.*;
public class Example16 {
public static void main(String[] args) {
   int i = 0;
do {
  System.out.println(i);
  i++;
}
while (i > 5);
}}
```

(ii)

```
import java.util.Scanner;
public class Example15 {
   public static void main(String[] args) {
int i = 0;
while (i > 5) {
  System.out.println(i);
  i++;
}    }
}
```

(ii) Read the value of i from the user and execute the while loop

for Loop

```
for (statement 1; statement 2; statement 3) {
  // code block to be executed
 }
```

Statement 1 is executed (one time) before the execution of the code block.

Statement 2 defines the condition for executing the code block.

Statement 3 is executed (every time) after the code block has been executed.

Example17 - Notepad

File Edit Format View Help

```
import java.io.*;
public class Example17 {
public static void main(String[] args) {
    int i = 0;
for (i = 0; i < 5; i++) {
  System.out.println(i);
}}
}
```

Command Prompt

```
E:\>javac Example17.java

E:\>java Example17
0
1
2
3
4

E:\>
```

Ln 5, Col 6 100% Windows (CRLF) UTF-8

Type here to search 10:17 AM 8/9/2020

Example

Example18 - Notepad

File Edit Format View Help

```
import java.io.*;
public class Example18 {
public static void main(String[] args) {
    int i,j = 0;
for (i = 0; i < 5; i++) {
for(j=0;j<5;j++)
{
  System.out.println(i);
}
 System.out.println(j);
}}
}
```

Command Prompt

```
0
0
0
5
1
1
1
1
1
5
2
2
2
2
2
5
3
3
3
3
3
5
4
4
4
4
4
5

E:\>\
```

Ln 8, Col 2 100% Windows (CRLF) UTF-8

Type here to search 10:26 AM 8/9/2020

<u>Try it yourself</u>

(i) Modify the program to generate output as below:

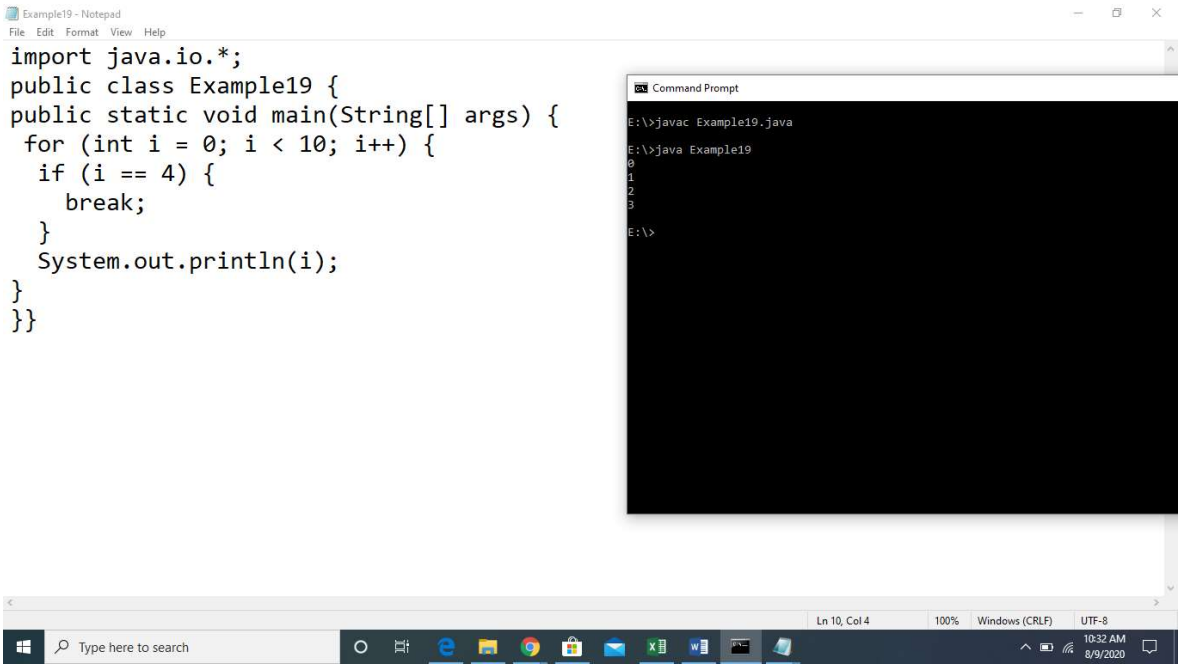
1	2
2	3
3	4
4	5
5	6
6	7

5.Break Statement

(i) The break statement can also be used to jump out of a loop.
(ii) The break statement allows for exits out the loop **unconditionally**

```
import java.io.*;

public class Example19 {

public static void main(String[] args) {

 for (int i = 0; i < 10; i++) {

  if (i == 4) {

   break;

  }

  System.out.println(i);

}

}}
```

Ouput

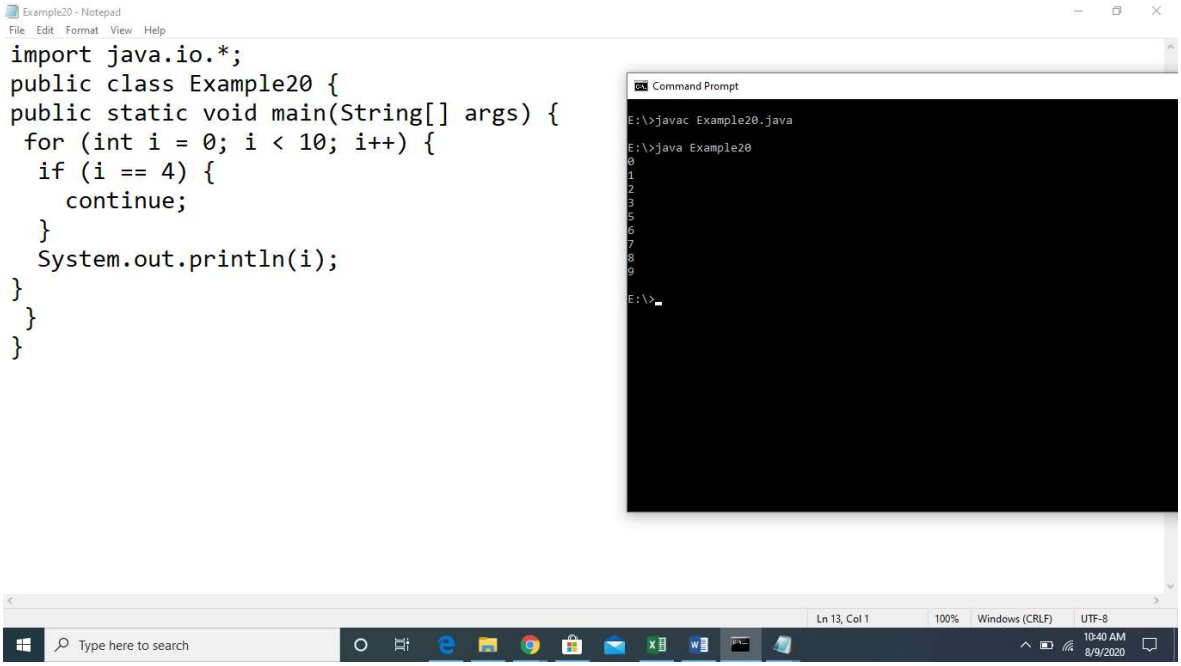


6.Continue statement

The continue statement breaks one iteration (in the loop), if a specified condition occurs, and continues with the next iteration in the loop.

```
import java.io.*;

public class Example20 {

public static void main(String[] args) {

 for (int i = 0; i < 10; i++) {

  if (i == 4) {

    continue;

  }

  System.out.println(i);

}

 }

}
```

<u>Output</u>



Try it yourself

1. Have two break statements and check the output
   Hint:
   break;
   break;
2. Have two continue statements and check the output

   Hint:

   continue;

   continue;

3. Try using a break with continue
   Break;
   Continue;
4. Try using continue with break statement

   continue;

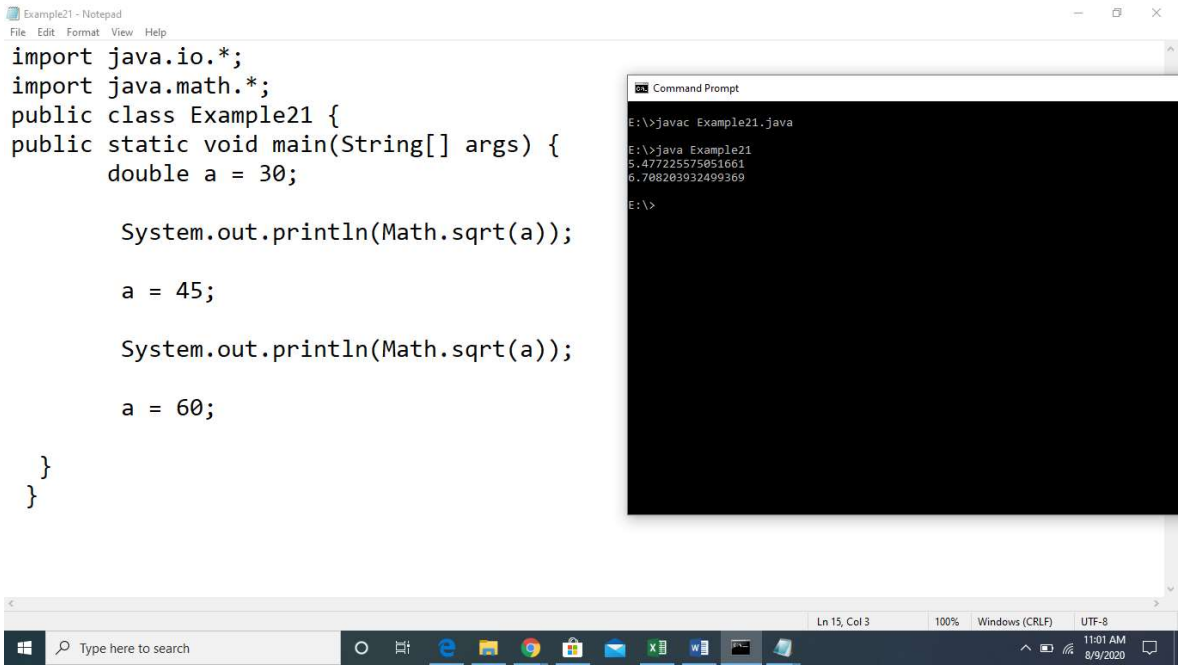
   break;

7.Java.Math package

It is a package that has methods for handling numeric functions.

```
import java.io.*;

import java.math.*;

public class Example21 {

public static void main(String[] args) {

      double a = 30;

      System.out.println(Math.sqrt(a));

       a = 45;

       System.out.println(Math.sqrt(a));

       a = 60;

  }

 }
```

Output



Method Name	Purpose	Example	Output
Abs	Returns the absolute value of the argument Arguments: Double, float, int, long	Math.abs(10)	
Round	Returns the closed int or long (as per the argument) Arguments: double or float	Math. Round(9.65)	
Ceil	Returns the smallest integer that is greater than	Math.Ceil(0.45)	

	or equal to the argument Arguments: double		
Floor	Returns the largest integer that is less than or equal to the argument Argument: Double	Math.floor(0.45)	
Min	Returns the smallest of the two arguments Argument: double, float, int, long	Math.min(2.3)	
Max	Returns the largest of the two arguments Argument: double, float, int, long double, float, int, long		
Sqrt	Returns the square root of a given number Arguments: int		

Note : Argument indicates the datatype of parameters that the method can take

Find it Yourself

Check whether the following Methods exist with its purpose

a. Max
b. Sin
c. Cos
d. Tan
e. Atan2
f. ToDegrees
g. Toradians
h. Random

8.Java Strings

A String is a collection of characters

Reading a string from the user

Syntax:

String name="India";

(i) Using Buffered Reader Class

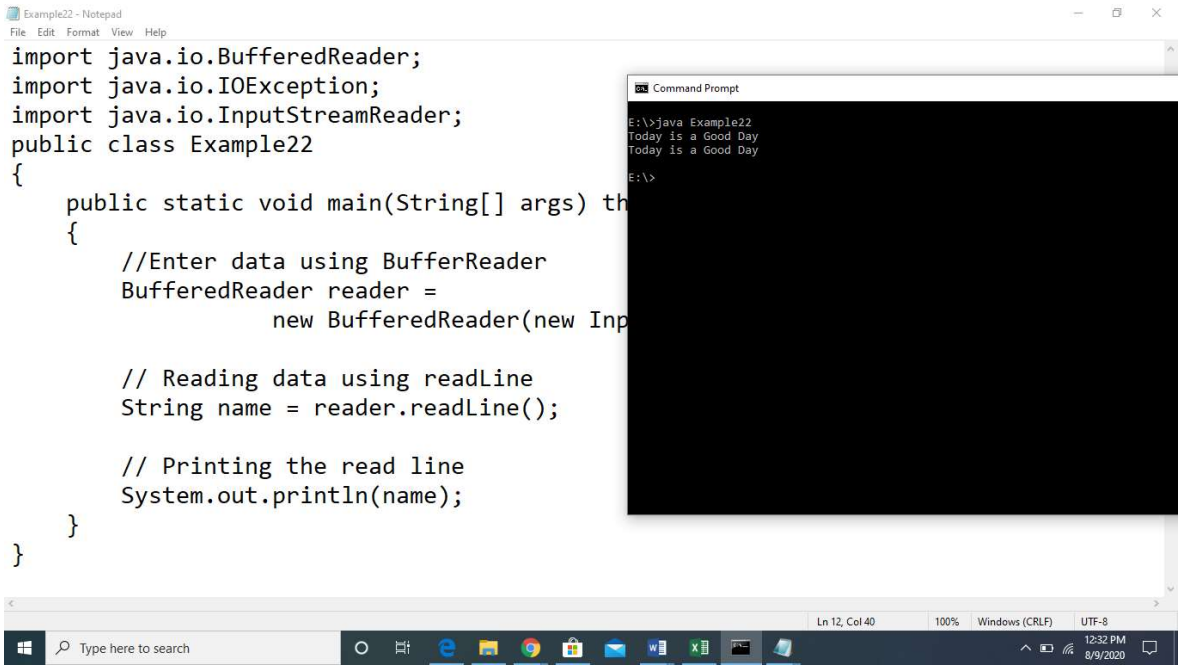
- **BufferedReader is a class for handling strings**
- **This method is used by wrapping the System.in (standard input stream) in an InputStreamReader which is wrapped in a BufferedReader, we can read input from the user in the command line.**

Example

```
import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class Example22

{

   public static void main(String[] args) throws IOException

   {

      //Enter data using BufferReader

      BufferedReader reader =

            new BufferedReader(new InputStreamReader(System.in));
```

```
        // Reading data using readLine
        String name = reader.readLine();


        // Printing the read line
        System.out.println(name);
    }
}
```

Output



(ii) Scanner Class

The main purpose of the Scanner class is to parse primitive types and strings using regular expressions, however it is also can be used to read input from the user in the command line.

Advantages:

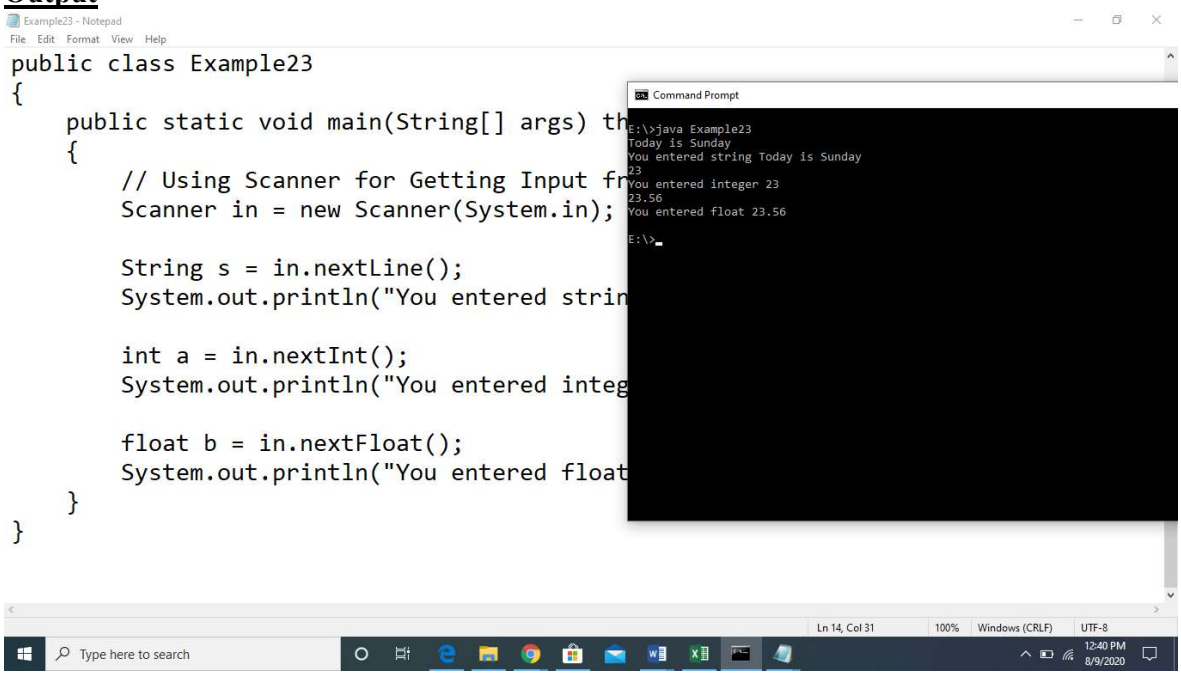
- Convenient methods for parsing primitives (nextInt(), nextFloat(), …) from the tokenized input.
- Regular expressions can be used to find tokens.

Example

```
import java.io.BufferedReader;
import java.io.IOException;
```

```
import java.util.Scanner;
public class Example23
{
   public static void main(String[] args) throws IOException
   {
      // Using Scanner for Getting Input from User
      Scanner in = new Scanner(System.in);

      String s = in.nextLine();
      System.out.println("You entered string "+s);

      int a = in.nextInt();
      System.out.println("You entered integer "+a);

      float b = in.nextFloat();
      System.out.println("You entered float "+b);
   }
}
```

Output



c. **Console Class**

It has been becoming a preferred way for reading user's input from the command line. In addition, it can be used for reading password-like input without echoing the characters entered by the user; the format string syntax can also be used (like System.out.printf()).

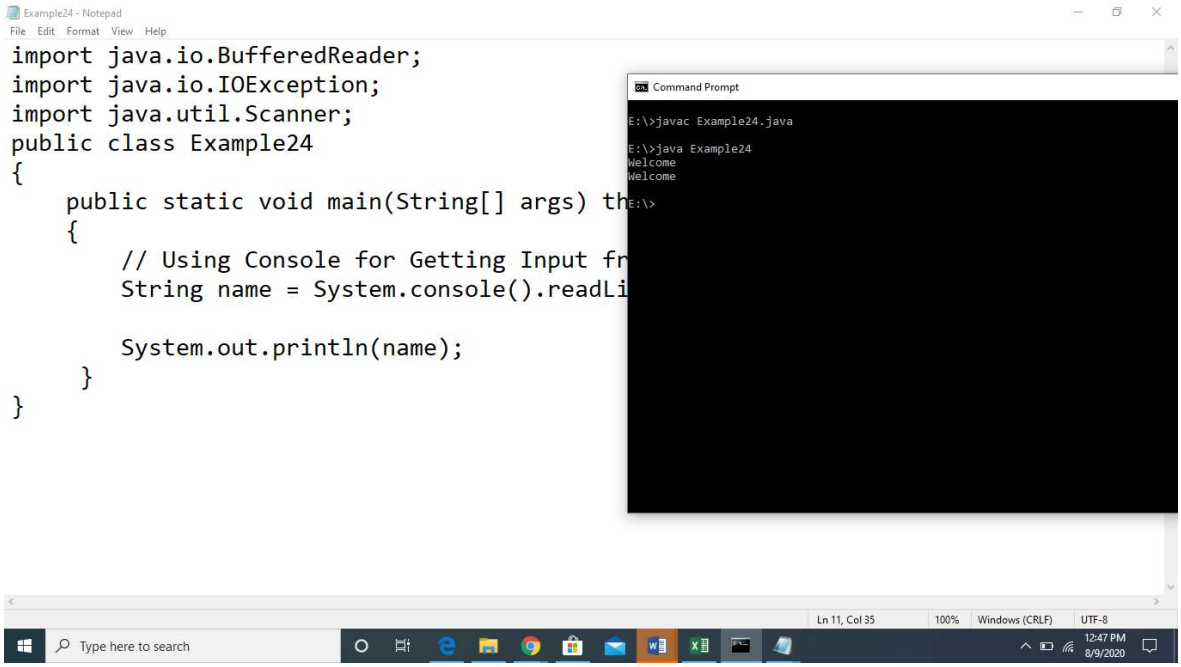
Advantages:

Reading password without echoing the entered characters.
Reading methods are synchronized.
Format string syntax can be used.

Drawback:
Does not work in non-interactive environment (such as in an IDE).

Example

```
import java.io.BufferedReader;
import java.io.IOException;
import java.util.Scanner;
public class Example24
{
   public static void main(String[] args) throws IOException
   {
      // Using Console for Getting Input from User
      String name = System.console().readLine();

      System.out.println(name);
   }
}
```



Practice Questions

1. Write a program to display ten proverbs on the screen
2. Write a program to display the even number given the 'n' from the user
3. Write a program to generate Fibonacci series
4. Write a program to convert time into minute and seconds
5. Write a program to print a Bill as below:

   Bill No: 1012

   Date :9-Aug-2020

ItemNo	Name	Quantity	Amount
1	Cinthol	12	200
2	Dettol	10	100
3	Oil	2	100
Total			400

   Use math package

6. Write a program to generate a EB receipt as below:

TamilNadu Generation And Distribution Corporation Ltd
CBE/Metro EDC

Assessment for the month of 07/2020
Service No : 03229010959
Consumer Name: S.Ramesh
Address : 3S PEELAMEDU
Tariff : LA1A Phase : 3 Assessed Units: 20

Bill date : 2020-07-20 Due date: 2020-08-10

CC Charge	Fixed Cost	MD Penalty	PF Penalty	E-Tax	Amount
10	10	10	250	0	250

Advance Amount : Rs.200
Amount to be paid : Rs.50

**This is only the current consumption charges. Arrears if any will be included in the bill and have to be paid.
This is an automatically generated email. Kindly do not reply to this mail.
Click here for making Online Payment: https://www.tnebnet.org/awp/login

Note:

a. Amount =CCCharge+Fixed Cost+PF Penalty+E-Tax
b. Amount to be paid=Advance Amount-Amount

Test case:

1. Amount as 250,Advance Amount as 100
2. Amount as 0,Advance amount as 200

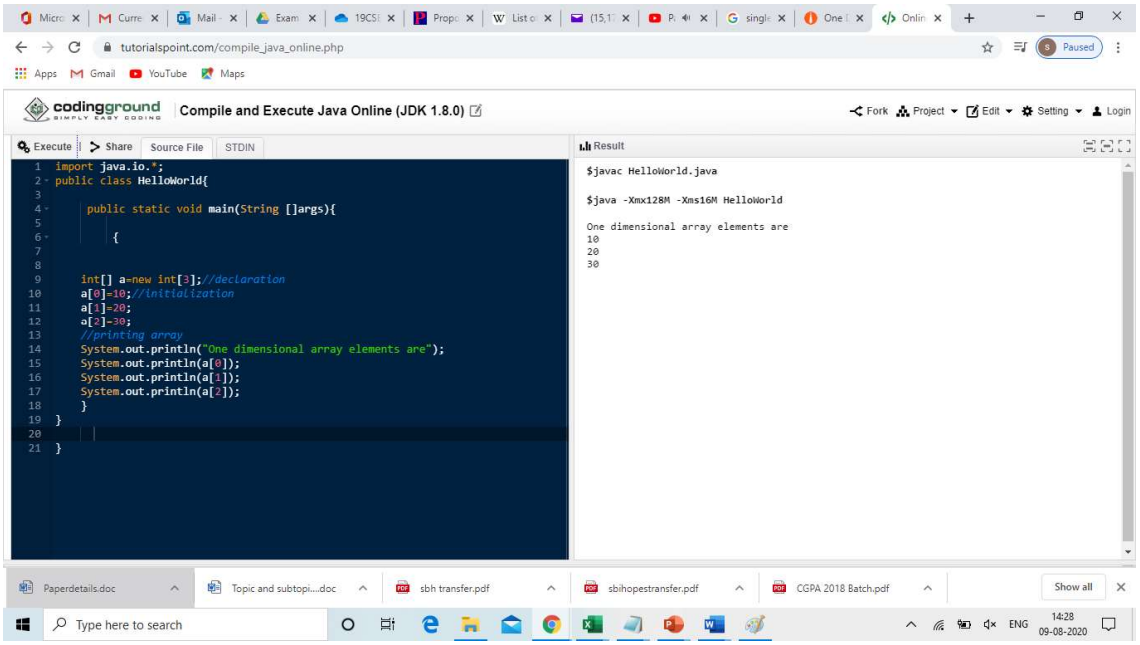
9.Array

- **An array is a collection of elements of one specific type in a horizontal fashion.**
- **The array in contention here is that of the one-dimensional array in Java programming.**

Example

```
import java.io.*;

public class HelloWorld{


    public static void main(String []args){


        {



            int[] a=new int[3];//declaration

            a[0]=10;//initialization

            a[1]=20;

            a[2]=30;

            //printing array

            System.out.println("One dimensional array elements are");

            System.out.println(a[0]);

            System.out.println(a[1]);

            System.out.println(a[2]);

            }

}
```

}



<u>Try it Yourself</u>

Read value for the array from the user

<u>A Sample program is given below:</u>

```
import java.io.*;
class OnedimensionalScanner
{
  public static void main(String args[])
  {
        int len;
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter Array length : ");
        len=sc.nextInt();
        int a[]=new int[len];//declaration
        System.out.print("Enter " + len + " Element to Store in Array :\n");
      for(int i=0; i<len; i++)
      {
```

```
        a[i] = sc.nextInt();
      }
      System.out.print("Elements in Array are :\n");
      for(int i=0; i<len; i++)
      {
        System.out.print(a[i] + "  ");
      }
  }
}
```

MultiDimensional Array

A multidimensional array is an array of arrays. Each element of a multidimensional array is an array itself. For example,

```
int[][] a = new int[3][4];
```

Here, we have created a multidimensional array named `a`. It is a 2-dimensional array, that can hold a maximum of 12 elements,

	Column 1	Column 2	Column 3	Column 4
Row 1	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 2	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 3	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Two-dimensional Array

Remember, Java uses zero-based indexing, that is, indexing of arrays in Java starts with 0 and not 1.

Let's take another example of the multidimensional array. This time we will be creating a 3-dimensional array. For example,

```
String[][][] data = new String[3][4][2];
```

Here, `data` is a 3d array that can hold a maximum of 24 (3*4*2) elements of type `String`.

Initializing a 2D Array

```
int[][] a = {
      {1, 2, 3},
      {4, 5, 6, 9},
      {7},
};
```

Each row of the multidimensional array in Java can be of different lengths.

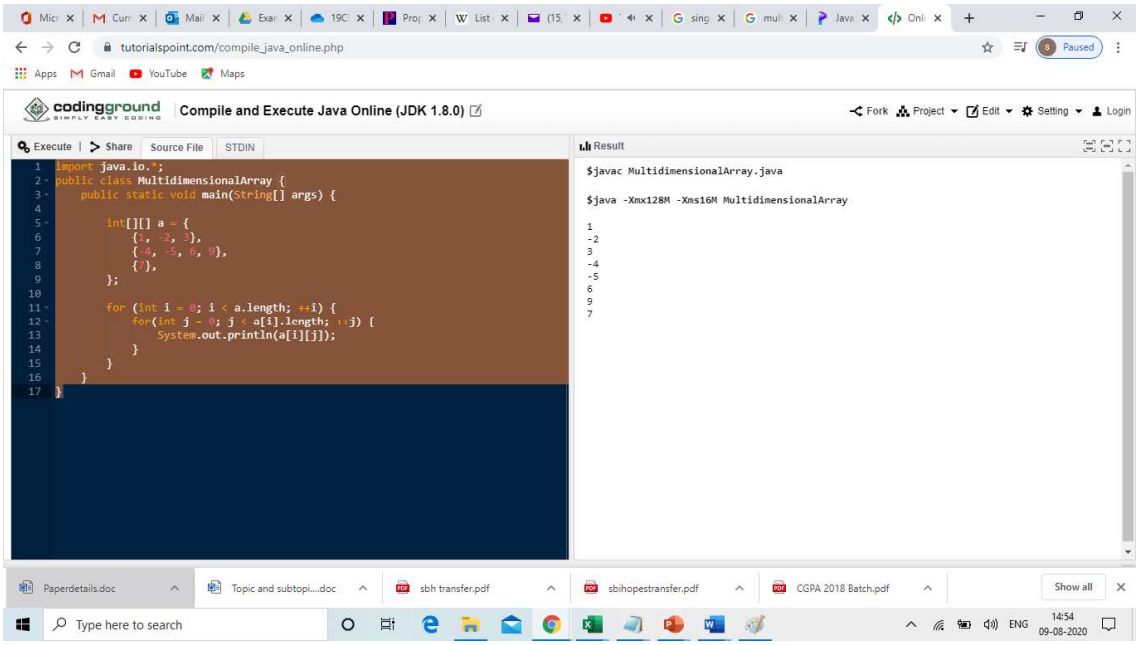
	Column 1	Column 2	Column 3	Column 4
Row 1	1 a[0][0]	2 a[0][1]	3 a[0][2]	
Row 2	4 a[1][0]	5 a[1][1]	6 a[1][2]	9 a[1][3]
Row 3	7 a[2][0]			

Initialization of Two-dimensional Array

Example: 2-dimensional Array

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


      int[][] a = {
         {1, -2, 3},
         {-4, -5, 6, 9},
         {7},
      };


      for (int i = 0; i < a.length; ++i) {
         for(int j = 0; j < a[i].length; ++j) {
            System.out.println(a[i][j]);
         }
      }
   }
}
```

Output



Self -Study

Three Dimensional Array

Array Initialization

```
// test is a 3d array
int[][][] test = {
        {
          {1, -2, 3},
          {2, 3, 4}
        },
        {
          {-4, -5, 6, 9},
          {1},
          {2, 3}
        }
};
```

Question

Write a program to initialize a 3D array and display its output