NOTE.1

101) Data Types

1. Java is a statically typed language.
2. A language is statically typed, if the data type of a variable is known at compile time.
3. This means that you must specify the type of the variable (Declare the variable) before you can use it.
4. Data type defines the values that a variable can take, for example if a variable has int data type, it can only take integer values.
5. In java we have two categories of data type:
    a. Primitive data types.
    b. Non-primitive data types – e.g. Arrays and Strings.

1) Primitive data types

In Java, we have eight primitive data types: boolean, char, byte, short, int, long, float and double.

No	Data Type	Valid Data	Size	Value Ranges
1	byte	Whole numbers	1 byte	from -128 to 127
2	short	Whole numbers	2 bytes	from -32,768 to 32,767
3	int	Whole numbers	4 bytes	from -2,147,483,648 to 2,147,483,647
4	long	Whole numbers	8 bytes	from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
5	float	Fractional numbers	4 bytes	6 to 7 decimal digits

6	double	Fractional numbers	8 bytes	15 decimal digits
7	char	Fractional numbers	2 bytes	Stores a single character/letter or ASCII values
8	boolean	True or False	1 bit	TRUE or FALSe value

https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html

https://onecompiler.com/java/

1.BYTE

```
public class Main
{
  public static void main (String[]args)
  {
    byte num;
    num = 113;
    System.out.println (num);
  }
}
```

If we put num=130, we will get an error.

2.SHORT

```
public class Main
{
```

```
  public static void main (String[]args)
  {
    short num;
      num = 150;
      System.out.println (num);
  }
}
```

If we put num=33000, we will get an error.

3.INTEGER

```
public class Main
{
  public static void main (String[]args)
  {
    int num;
      num = 32767;
      System.out.println (num);
  }
}
```

If we put num=2200000000, we will get an error.

4.LONG

```
public class Main
{
  public static void main (String[]args)
  {
    long num;
      num = 1649610318000L;
      System.out.println (num);
```

```
  }
}
```

- Byte type variables are especially useful when you are working with a stream of data from a network or a file. (https://docs.oracle.com/javase/tutorial/essential/io/bytestreams.html)

- The following is an example program which uses byte streams to copy xanadu.txt, one byte at a time.

Try run the below codes at https://onlinegdb.com

Filename: Main.java

```
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;

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

        FileInputStream in = null;
        FileOutputStream out = null;

        try {
            in = new FileInputStream("xanadu.txt");
            out = new FileOutputStream("outagain.txt");
            int c;

            while ((c = in.read()) != -1) {
                out.write(c);
            }
        } finally {
            if (in != null) {
                in.close();
            }
            if (out != null) {
```

```
                out.close();
            }
        }
    }
}
```

Code example:

5. FLOAT

```

public class Main
{
  public static void main (String[]args)
  {
    float num = 19.98f;
    System.out.println (num);
  }
}
```

6. DOUBLE

```

public class Main
{
  public static void main (String[]args)
  {
    double num = -42937737.9d;
    System.out.println (num);
  }
}
```

QUIZ

https://onecompiler.com/java

Add 24 and 36 using integer data type

```
public class Main {
  public static void main (String[]args){
    int a;
    int b;
    int c;
    a = 24;
    b = 36;
    c = a+b;
    System.out.println (c);
  }
}
```

CAREFUL WHEN WORKING WITH DATA TYPES

We have to be careful when working with data types.
Wrong selection of data types may lead to runtime error.

Add 24 and 36 using byte data type

```
-asyraf-
public class Main
{
  public static void main (String[]args)
  {
    byte a;
    byte b;
```

```
    a = 24;
    b = 36;
    System.out.println (a+b);
  }
}
```

NO ERROR

Add 24 and 36 using byte data type

```
public class Main
{
  public static void main (String[]args)
  {
    byte a;
    byte b;
    byte c;
    a = 24;
    b = 36;
    c=a+b;
    System.out.println (c);
  }
}
```

ERROR
Output:

```
Main.java:10: error: incompatible types: possible lossy conversion
from int to byte
    c=a+b;
       ^
1 error
error: compilation failed
```

Add 24 and 36 using byte data type

```
public class Main
{
  public static void main (String[]args)
  {
        byte a = 3;
```

```
        byte b = 8;
        byte c;
        c = (byte) (a + b); //addition typecasted

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

NO ERROR

Add 24 and 36 using byte data type

```
public class Main
{
  public static void main (String[]args)
  {
        byte a = 100;
        byte b = 110;
        byte c;
        c = (byte) (a + b); //addition typecasted
        System.out.println(c);
  }
}
```

NO ERROR BUT OUTPUT IS

```
-46
```

Read further:
https://www.educative.io/edpresso/addition-of-byte-values-in-java

7. CHARACTER

https://onecompiler.com/java

```
public class CharExample1 {

    public static void main(String[] args) {

        char char1='a';
        char char2='A';


        System.out.println("char1: "+char1);
        System.out.println("char2: "+char2);
    }
}
```

Declare character and assign a letter.

```
public class CharExample2 {

    public static void main(String[] args) {

        char char1=65;
        char char2=97;

        System.out.println("char1: "+char1);
        System.out.println("char2: "+char2);

    }
}
```

Declare a character and assign Ascii numeric value.

```
public class CharExample3 {

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

```
        int num1=97;
        char char1=(char)num1;

        int num2=65;
        char char2=(char)num2;

        System.out.println("char1: "+char1);
        System.out.println("char2: "+char2);

    }

}
```

Declare a character and assign an integer value using casting. E.g `(char)num1;`

```
public class CharExample4 {

    public static void main(String[] args) {

        char char1='\u0061';
        char char2='\u0041';


        System.out.println("char1: "+char1);
        System.out.println("char2: "+char2);


    }

}
```

Declare a character and assign a Unicode character code.
E.g https://unicode-table.com/en/

```
public class CharExample5 {

    public static void main(String[] args) {

        char char1='A';
        char1=(char)(char1+1);

        System.out.println("char: "+char1);

    }
}
```

Character data operation (addition)

```
import java.util.Arrays;

public class CharExample6 {

    public static void main(String[] args) {

        String str="myclass";
        char[] ch=str.toCharArray();

        System.out.println("String: "+str);
        System.out.println("char: "+Arrays.toString(ch));

    }

}
```

String is converted into an array of characters.

```
public class CharExample7 {

    public static char display()
```

```
    {
        return 'a';
    }

    public static void main(String[] args) {

        CharExample7 c=new CharExample7();
        System.out.println(c.display());

    }
}
```

c.display() - a new object name c, having a method “display()”. Method display returns ‘a’.

public static char display

display = name of a method
char = value return by display method
public = can be called by other methods
static = value stays, doesn’t change.

What is the error of the below codes

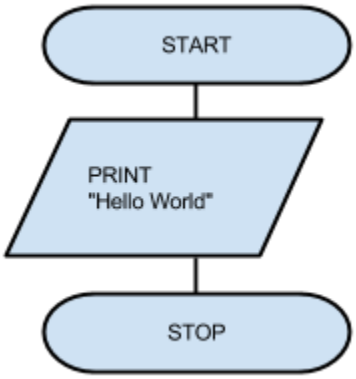
```
public class Main {

    public static void main(String[] args) {
       int myNum = 9;
       float myFloatNum = 8.99f;
       char myLetter = 'A';
       boolean myBool = false;
       char myText = "Hello World";
    }
}
```

Output:

Main.java:10: error: incompatible types: String cannot be converted to char

```
char myText = "Hello World";
              ^
1 error
error: compilation failed
```

```
int myNum = 9;
float myFloatNum = 8.99f;
char myLetter = 'A';
boolean myBool = false;
string myText = "Hello World";
```

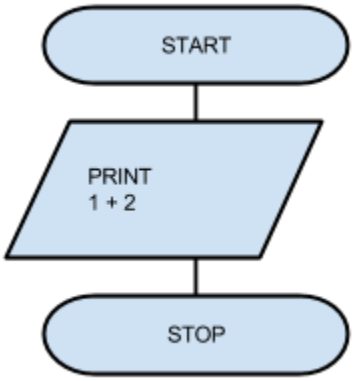


Write Java codes to implement the above design.

```
public class Main {
    public static void main(String[] args) {
 System.out.println("Hello World!");
    }
```

```
}
```

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```



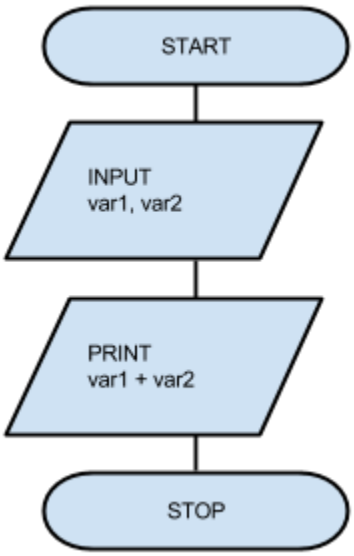
Write Java codes to implement the above design.

```
public class Main
{
  public static void main (String[]args)
  {
        byte a = 1;
        byte b = 2;
        byte c;
        c = (byte) (a + b); //addition typecasted

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

Tidak selamat jika untuk nilai yang lebih besar

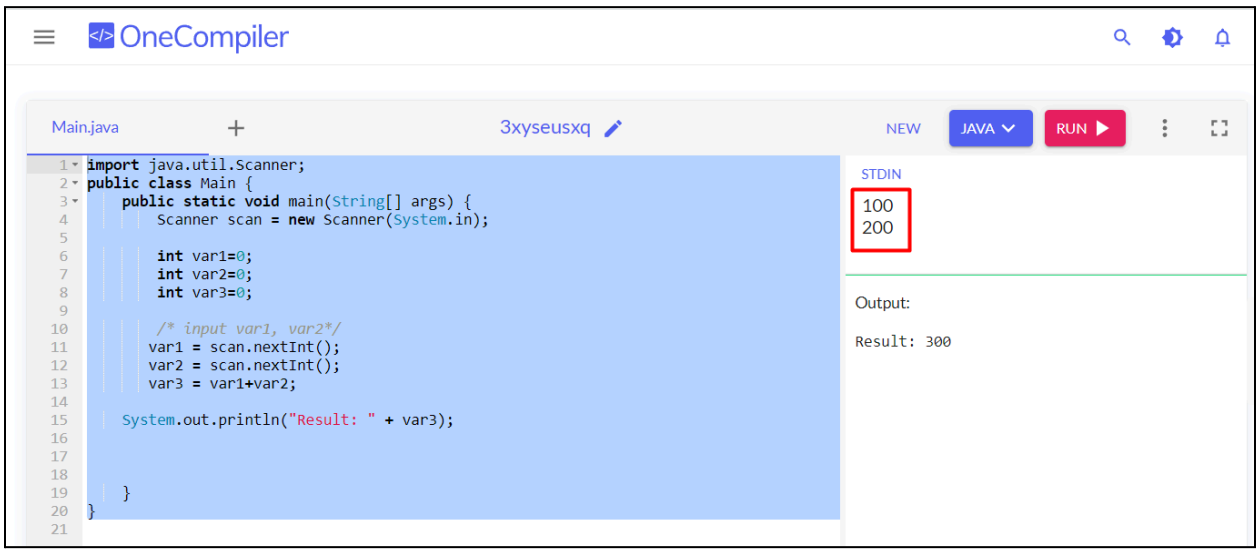
```
public class Main {

    public static void main(String[] args) {


int a = 1;
int b = 2;
int c = a+b;


System.out.println(c);

    }
}
```

DATA INPUT



Write Java codes to implement the above design.

```
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        int var1=0;
        int var2=0;
        int var3=0;

        /* input var1, var2*/
       var1 = scan.nextInt();
       var2 = scan.nextInt();
       var3 = var1+var2;
    System.out.println("Result: " + var3);
    }
}
```



Type 100 and 200 in the STDIN field.

1. Download code from OneCompiler

2. Save source code in c:\test.
3. Run command: `"C:\jdk1.8.0\bin\javac" Main.java`
4. Run command: `"C:\jdk1.8.0\bin\java" -cp . Main`
5. Output:

   When the user runs this program, the user has to type 100, press ENTER, type 200. Then the program will output 300.

```
C:\test>"C:\jdk1.8.0\bin\javac" Main.java

C:\test>"C:\jdk1.8.0\bin\java" -cp . Main
100
200
Result: 300

C:\test>
```

```
import java.util.Scanner;

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

        Scanner input = new Scanner(System.in);

        System.out.print("Enter an integer: ");
        int number = input.nextInt();
        System.out.println("You entered " + number);

        // closing the scanner object
        input.close();
    }
}
```

```
import java.util.Scanner;

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

        Scanner input = new Scanner(System.in);

        // Getting float input
        System.out.print("Enter float: ");
        float myFloat = input.nextFloat();
        System.out.println("Float entered = " + myFloat);

        // Getting double input
        System.out.print("Enter double: ");
        double myDouble = input.nextDouble();
        System.out.println("Double entered = " + myDouble);

        // Getting String input
        System.out.print("Enter text: ");
        String myString = input.next();
        System.out.println("Text entered = " + myString);
    }
}
```

```
100.25
44.15
Hello
```

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

```
{
  // create a BufferedReader using System.in
  BufferedReader obj = new BufferedReader(new
InputStreamReader(System.in));
   String str;

 System.out.println("Enter lines of text.");
 System.out.println("Enter 'stop' to quit.");
   do {
    str = obj.readLine();
    System.err.println(str);
  }   while(!str.equals("stop"));
}
}
```

Output:

```
Enter lines of text.
Enter 'stop' to quit.

Error: Command failed: timeout 7 java ReadLineDemo.java
```

```
selamat
hari
raya
stop
```

READING TEXT FILE FROM A GIVEN FILE PATH

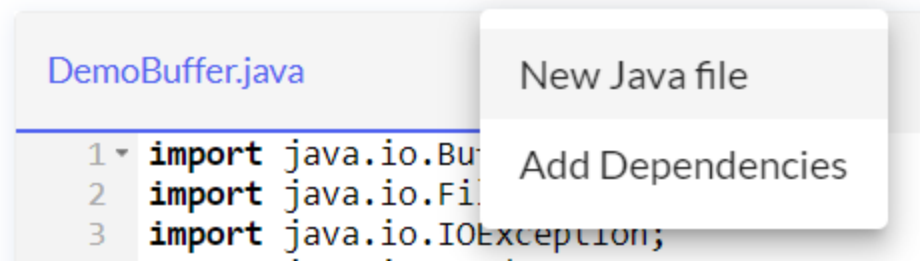
```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.io.Reader;

public class DemoBuffer
{
```

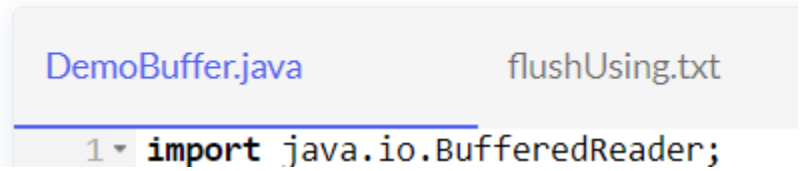
```
    public static void main(String[] args) {

        try {

            Reader reader = new
FileReader("E:\\users\\flushUsing.txt");

            try(BufferedReader bufferedReader =
                new BufferedReader(reader)){

                String line = bufferedReader.readLine();
                while(line != null) {
                    //do something with line

                    line = bufferedReader.readLine();
                    System.out.println(line);
                }

            }
        } catch (IOException e) {

            System.out.println(e);
        }

    }
}
```

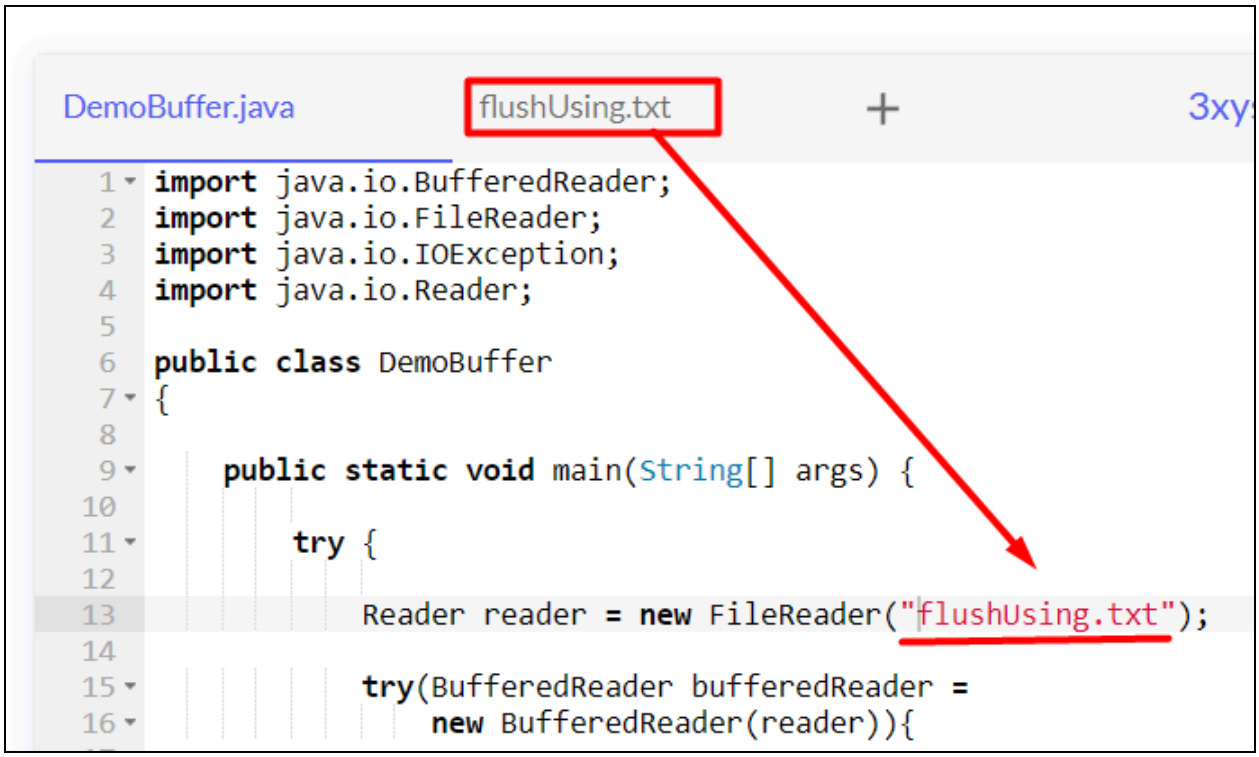
Output:

java.io.FileNotFoundException: E:\users\flushUsing.txt (No such file or directory)

Program outputs error because file is not found at a given path.







Download codes from OneCompiler.com into C:\test
Change file name to “flushUsing.txt”
Compile and Run.
We are reading text files from the same folder.

```
C:\test>"C:\jdk1.8.0\bin\javac" DemoBuffer.java

C:\test>"C:\jdk1.8.0\bin\java" -cp . DemoBuffer
hari
raya
null

C:\test>
```

Move flushUsing.txt to C:\test1\

Edit content as:

Selamat
Tahun
baru

Compile

Run

We are reading text files from the same folder.

```
C:\test>"C:\jdk1.8.0\bin\javac" DemoBuffer.java

C:\test>"C:\jdk1.8.0\bin\java" -cp . DemoBuffer
tahun
baru
null
```

c:\test\DemoBuffer.java

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

```
public class DemoBuffer
{

    public static void main(String[] args) {

        try {

            Reader reader = new
FileReader("C:\\test1\\flushUsing.txt");

            try(BufferedReader bufferedReader =
                new BufferedReader(reader)){

                String line = bufferedReader.readLine();
                while(line != null) {
                    //do something with line

                    line = bufferedReader.readLine();
                    System.out.println(line);
                }

            }
        } catch (IOException e) {

            System.out.println(e);
        }

    }
}
```

c:\test1\flushUsing.txt

```
selamat
tahun
baru
```

Kenapa baris pertama tidak dipaparkan di skrin?

Question: Why was the first line not printed on the screen?

Output:

```
hari
raya
null
```

Answer: The first line was not printed because it was overridden by the readline command in the While Loop

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

public class DemoBuffer
{

    public static void main(String[] args) {

        try {

            Reader reader = new FileReader("flushUsing.txt");

            try(BufferedReader bufferedReader =
                new BufferedReader(reader)){

                String line = bufferedReader.readLine();
                System.out.println(line);
                while(line != null) {
                    //do something with line
                    line = bufferedReader.readLine();
                    System.out.println(line);
                }

            }
        } catch (IOException e) {

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

```
    }
}
```

The solution above is less efficient because we repeat the same command twice i.e “System.out.println(line);”

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

public class DemoBuffer
{

    public static void main(String[] args) {

        try {

            Reader reader = new FileReader("flushUsing.txt");

            try(BufferedReader bufferedReader =
                new BufferedReader(reader)){

                String line = bufferedReader.readLine();

                while(line != null) {
                    System.out.println(line);
                    line = bufferedReader.readLine();

                }

            }
        } catch (IOException e) {

            System.out.println(e);
        }

    }
}
```

This code is better because there is no repetition of commands.

Output:

```
selamat
hari
raya
```


```
import java.io.Console;
class Main1
{
    public static void main(String args[])
    {
        String str;
        //Obtaining a reference to the console.
        Console con = System.console();
        // Checking If there is no console available, then exit.
        if(con == null)
        {
            System.out.print("No console available");
            return;
        }
        // Read a string and then display it.
        str = con.readLine("Enter your name: ");
        con.printf("Here is your name: %s\n", str);
   }
}
```

COMPILE:
C:\test>"C:\jdk1.8.0\bin\javac" Main1.java

RUN:
C:\test>"C:\jdk1.8.0\bin\java" -cp . Main1

```
C:\test>"C:\jdk1.8.0\bin\javac" Main1.java

C:\test>"C:\jdk1.8.0\bin\java" -cp . Main1
Enter your name: ahmad
Here is your name: ahmad
```

PRINT COMMAND

Print line
System.out.**println**(line);

Print
System.out.**print**("No console available");

Print Formatted Output
con.**printf**("Here is your name: %s\n", str);

\n= newline

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

       Console col = System.console();
      String fmt = "%1$4s";
      String alpha = null;
```

```
      if(col==null) {
          System.out.println("No console Available");
      }

      try {
         // creates a console object
       col= System.console();


          // read the line from the user input
            alpha = col.readLine(fmt, "Enter :","Alphabets: ");

            // prints
            col.format(fmt, "Alphabets entered: " + alpha);
         } catch(Exception e) {
         System.out.println(e);
      }
   }
}
```

COMPILE:

```
C:\test>"C:\jdk1.8.0\bin\javac" Readl.java
```

RUN:

```
C:\test>"C:\jdk1.8.0\bin\java" -cp . Readl
```

```
C:\test>"C:\jdk1.8.0\bin\javac" Readl.java

C:\test>"C:\jdk1.8.0\bin\java" -cp . Readl
Enter :hello
Alphabets entered: hello
```

```
import java.io.Console; // import predefine class
class FormattedOutput {
  public static void main(String[] args)  {
```

```
    Console col = System.console();

    if (col != null) {
        //String format
      String fmt = "%1$4s %2$10s %3$10s%n";
        // format
      col.format(fmt, "Student","34", "5");
      col.format(fmt, "CSS","2","50");
      col.format(fmt, "Java","3","30");
     }
 }   }
```

COMPILE:
"C:\jdk1.8.0\bin\javac" FormattedOutput.java

RUN:
C:\test>"C:\jdk1.8.0\bin\java" -cp . FormattedOutput

```
C:\test>"C:\jdk1.8.0\bin\javac" FormattedOutput.java

C:\test>"C:\jdk1.8.0\bin\java" -cp . FormattedOutput
Student         34          5
 CSS          2         50
Java          3         30
```

```
String fmt = "%1$10s %2$10s %3$10s%n";
```

```
C:\test>"C:\jdk1.8.0\bin\javac" FormattedOutput.java

C:\test>"C:\jdk1.8.0\bin\java" -cp . FormattedOutput
   Student         34          5
       CSS          2         50
      Java          3         30
```