Day2

- If .java file do not contain any type(interface/class/enum) then java compiler do not generate .class file.
- Java Compiler generates .class file per type defined in .java file.
- Name of the file and class name can be difrrent. But generally name of the class and name of the file should be same.

System.out.println

- java.lang package contains all the fundamental classes of core java.
- java.lang package is by default imported in every .java file.
- System is a final class declared in java.lang package.
- Variable declared inside function(method) is called local variable(Method Local Variable).
- variable declared inside class is called data member.
- In Java data member is called as field.
- Fields declared inside System class:
    1. public static final InputStream in; //ref
    2. public static final PrintStream out;//ref
    3. public static final PrintStream err;//ref

```
package java.lang;
public final class System extends Object{
    //Fields
    public static final InputStream in; //ref
    public static final PrintStream out;//ref
    public static final PrintStream err;//ref
}
```

- How to access fields of System class?

1. System.in; //represents keyboard
2. System.out; //represents monitor
3. System.err; //Error Stream - represents monitor

- stdin, stdout, stderr are standard stream objects of C language.

```
printf("Hello World");
fprintf(stdout,"Hello World");

int number;
scanf("%d", &number );
fscanf(stdin, "%d", &number );

if( num2 == 0 )
    fprintf(stderr,"/ by zero");
```

```
else
{
    int result = num1 / num2;
    fprintf(stdout,"Result  :   %d\n", result);
}
```

- System.in, System.out and System.err are standard stream objects of Java.
- PrintStream is a class declared in java.io package
- print, printf and println are non static, overloaded methods of java.io.PrintStream class.

Entry Point Method

- In Java, function is also called as method.
- Set or rules and guidelines are called as specification/standard.
- In Java Specification = Abstract Classes + Interfaces.
- Java Language Specification(PDF:Oracle->JCP)
  - It is a specification for Java Language.
- Java Virtual Machine Specification(PDF:Oracle->JCP)
  - It is a specification for JVM implementation.
- According to JVM specification, "main" method must be entry point method.
- Syntax: public static void main(String[] args);
- We can overload main method in java.
- We can define main method per class but only main can be considered as entry point method in Java.
- If sutaible main method is not available inside class then compiler do not generate error but JVM generates error.

Data Type

- Data type of any variable/instance describes 4 things:
  1. Memory : How much memory is required to store the data.
  2. Nature : Which type of data is allowed to store inside memory
  3. Operation : Which operations(functionality) are allowed to perform on data.
  4. Range : Set of values which are allowed to store inside memory.
- Types of data type in Java:
  1. Primitive Data Types
  2. Non Primitive Data Types

Primitive Data Types

- It is also called as Value Type.
- There are 8 primitive/value types in Java. Sr.No Primitive Type Size Field's Default value Wrapper Class

1.
```
 boolean         :   Not Mentioned   :       false           :
Boolean
```

2.
```
 byte          :   1 byte          :       0              :   Byte
```

3.
```
 char          :   2 byte          :        \u0000        :
Character
```

4.
```
 short         :   2 bytes         :       0              :   Short
```

5.
```
 int           :   4 bytes         :       0              :
Integer
```

6.
```
 float         :   4 bytes         :       0.0f           :   Float
```

7.
```
 double        :   8 bytes         :       0.0d           :
Double
```

8.
```
 long          :   8 bytes         :       0L             :   Long
```

- Default value of field of primitive type is generally 0.
- In Java, primitive types are not classes. But for every pritive type class is given it is called Wrapper class.
- All the Wrapper classes are declared in java.lang Package.
- Variable of primitive/value type get space on Java Stack.

```
class Program{
    int num1;   //Field
    public static void main(String[] args) {
        int num2;   //Method Local Variable
    }
}
```

Non Primitive Data Types

- It is also called as Reference Type.
- There are 4 non primitive/reference types in Java.

1. Interface
2. Class
3. Enum
4. Array

- Instance of non primitive/reference type get space on Heap.

Components of JVM:

1. Class Loader Sub System
    - Classloader is responsible for loading .class file from HDD into JVM memory.
    - Types of class loader:
        1. Bootstrap Classloader
        2. Extension Classloader
        3. Application Classloader
2. Runtime Data Areas
    1. Method Area
    2. Heap
    3. Java Stack
    4. PC Register
    5. Native Method Stack
3. Execution Engine
    1. Interpreter
    2. Just In Time (JIT) Compiler

Dynamically Type Checked Language(s)

- Consider example in python

```
number = 10 #OK : Python
number = "DAC" #OK : Python
```

- In python type of variable can be decided by looking toward value

Statically Type Checked Language(s)

- Consider example in Java

```
number = 10; //Not OK
int number = 10;    //OK
```

- Java compiler do not decide type of variable by looking toward value. Rather if we want to use any variable it is mandatory to mention it type.

```
class Program{
    public static void main( String[] args ){
        int number; //OK
        System.out.println("Number  :   "+number); //error: variable
number might not have been initialized
    }
}
```

- We can not use any(primitive/non primitive) type of local variable w/o storing value inside it.

```
class Program{
    public static void main(String[] args) {
        int num1 = 10;  //OK
        int num2;   //OK
        num2 = num1;    //OK
        System.out.println("Num1  :   "+num1); //10
        System.out.println("Num2  :   "+num2); //20
    }
}
```

Initialization

```
int num1 = 10;      //Initialization
int num2 = num1;    //Initialization
```

- Initialization is a process of storing user defined value inside variable during its declaration.
- We can initialize any variable only once.

Assignment

```
int num1 = 10;      //Initialization
int num2;
num2 = num1;    //Assignment
num2 = 20;       //Assignment
```

- Assignment is a process of storing user defined value inside variable after its declaration.
- Assignment can be done multiple times.

Widening

```
class Program{
    public static void main(String[] args) {
        int num1 = 10;  //Initialization
```

```
        double num2 = (double)num1;  //Widening : OK
        double num3 = num1;  //Widening : OK
    }
}
```

- Widening is a process of converting value of variable of narrower type into wider type.
- In case of Widening explicit type casting is optional.

Narrowing

```
class Program{
    public static void main(String[] args) {
        double num1 = 10.5;  //Initialization
        int num2 = ( int )num1; //Narrowing : OK
        int num3 = num1; //Narrowing : NOT OK
    }
}
```

- Narrowing is a process of converting value of variable of wider type into narrower type.
- In case of Narrowing explicit type casting is mandatory.

Boxing

- It is the process of converting state of variable of value type into reference type.
- Example

```
int num1 = 10;
String str = Integer.toString( num1 );  //Boxing
```

```
int num1 = 10;
String str = String.valueOf( num1 );  //Boxing
```

UnBoxing

- It is the process of converting state of instance of reference type into value type.
- Example

```
String str = "125";
String str = Integer.parseInt(str );  //UnBoxing
```