Day 6

```
class Complex
{
    void printRecord( )
    {
        System.out.println("Complex.printRecord");
    }
}
class Program
{
    public static void main(String[] args)
    {
        Complex c1;
        c1.printRecord( );//Error
    }
}
```

- If we want to use any type of local variable then it is necessary to store value inside it. In above code, c1 do not contain any value hence line no 15 is giving error.
- Consider few constants/literals

```
boolean v1 = false;
char v2 = 'A';
int v3 = 10;
double v4 = 3.14;
String v5 = "Sandeep";
Complex v6 = new Complex();
Complex v7 = null;
```

- null is literal/constant in java, which is used to initialize reference variable.
- If reference contains null value then such variable is called null refeference variable / null object.

```
int number = null;  //Not OK
Complex c1 = null;  //OK : c1-> null reference variable.
```

NULL(C/C++) versus null(Java)

- In C/C++, NULL is a macro(#define NULL 0).
- In java, null is a literal.
- In C/C+, NULL is used to initialize pointer.
  - Example : int *ptr = NULL; //ptr is NullPointer
- In Java, null is used to initialize reference variable.

  - Example : Employee emp = null; //emp is null reference variable.

NullPointerException

- Field of the class which get space inside instance is called instance variable. In other words, non static field declared inside class is called instance variable.
- A method of a class, which is designed to call on instanace/object is called instance method. In other words, non statoc method of a class is called instance method.
- Instance Members = Instance Variable + Instance Method.
- A class from which we can create instance is called concrete class. In other words, we can instantiate concrete class.
- A method of a class, which is having a body is called concrete method.

```
public static void main(String[] args)
{
    Complex c1 = null; //c1 is null object
    c1.printRecord( );//NullPointerException
}
```

- Using null object, if we try to access any member the class then JVM throws NullPointerException.
- How to solve null pointer exception:

```
public static void main(String[] args)
{
    Complex c1 = null; //c1 is null object
    c1 = new Complex();
    c1.printRecord( );//NullPointerException
}
```

Value Type Versus Reference Type

- Primitive type is also called as Value Type and non primitive type is also called as reference type.
- There are 8 value types(boolean, byte, char, short, int, long, float, double) and 4 reference types(interface, class, enum, array)
- Instance of value type get space on stack section whereas instance of reference type get space on Heap section
- We can not create instance of value type using new operator But to create instance of reference type it is mandatory to use new operator.

```
int x = 10;
int x = new int(10);    //NOt OK

Complex c1(10,20);  //Not OK
Complex c1 = new Complex(10,20);  //Not OK
```

- Variable of value type contain value and variable of reference type contain reference of instance.
- If we assign variable of value type to the another variable of value type then value gets copied. If we assign variable of reference type to the another variable of reference type then reference gets copied.

```
int x = 10;
int y = x;  //x = 10, y = 10

Complex c1 = new Complex( 10,20);
Complex c2 = c1; //Shallow Copy Of reference.
```

- Variable value type by default contain 0 and variable of reference type by default contain null.
- Variable of value type do not contain null value. But variable of reference type can contain null value.

```
int number = null;  //Not OK
Complex c1 = null;  //OK
```

Abstraction

- It is major pillar of oops.
- Process of getting essential things from instance/object/System is called abstraction.
- It describes outer behavior of system/object.
- Abstraction focuses on essential characteristics of some object relative to the percpective of viewer.
- Using abstraction, we can achive simplictiy.
- Abstraction in java:

```
public static void main(String[] args)
{
    Scanner sc = new Scanner();
    String name = sc.nextLine();
    int empid = sc.nextInt();
    float salary = sc.nextFloat();
}
```

```
public static void main(String[] args)
{
    Complex c1 = new Complex();
    c1.acceptRecord();
    c1.printRecord();
}
```

Encapsulation and Data Security

- Encapsulation is major pillar of oops.

- Implentation of abstraction is called encapsulation. In other words, binding of data and code together is called encapsulation.
- Hiding repesents encapsulation.
- Encapsulation describes internal behavior of instance.
- Abstraction and encapsulation are complementary concepts: Abstraction focuses on the observable behavior of an object, whereas encapsulation focuses on the implementation.
- Advnatage:
    1. We can achive abstraction
    2. We can achive data hiding/data encapsulation.
- Encapsulation in Java

```
class Complex
{
    //Field -> Data
    int real, imag;
    //Methods -> Code
    void acceptRecord(){    }
    void printRecord(){    }
}
```

- If we declare any field private then it its called data hiding/data encapsulatiion.
- Process of giving controlled access to the data is called data security.

```
public void setBalance(float balance)
{
    if( balance >= 0 && balance < 100000 )
        this.balance = balance;
    else
        throw new RuntimeException("Invalid Balance ");
}
```

Access Modifier

- If we want to control visibility of members of the class then we should use access modifier.
- There are 4 access modifiers in Java:
    1. private
    2. Packge Level Private
    3. protetced
    4. public
-

Final

Static In Java

Path and Classpath