Day 4

toString() method

- It is non final and non native method of java.lang.Object class.
- Syntax:

> public String toString( );

- If we want to return state of current instance in String format then we should use toString() method.
- If we do not define toString() method inside class then it's super class's toString() method gets called.
- "toString" method of java.lang.Object class returns String in following format:

> F.Q.TypeName@HashCode

```
 getClass().getName()+'@'+ Integer.toHexString(hashCode())
```

- According to clients requirement, if implementation of super class method is logically incomplete / partialy complete then we should override that method in sub class.
- The result in toString() method should be a concise but informative that is easy for a person to read.

```
public String toString( )
{
    //return this.name+" "+this.empid+" "+this.salary;
    return String.format("%-15s%-5d%-10.2f", this.name, this.empid,
this.salary);
}
```

- Note : Every sub class should override toString method.

Enum

- Consider enum in C language.

```
enum ShapeType  //ShapeType : Enum Name
{
    LINE, RECT, OVAL    //Enumerators
    //LINE=0, RECT=1, OVAL=2
};
```

- Instantiation:

```
enum ShapeType sh = RECT;
```

```
enum A  //OK
{
    int num1 = 10   //OK
};
enum B  //OK
{
    int num1 = 10   //Not OK
};
```

- If we want to assign name to the Integral literal/constant then we should use enum.
- enum do not improve efficiency or reduce code size rather it help us to improve readability of source code.
- enum is keyword in java.
- Synatx:

```
enum Color  //Enum Name
{
    RED, GREEN, BLUE    //Enum Constants
    //RED=0, GREEN=1, BLUE=2 //0,1,2 : Ordinals
}
```

- Implict value of enum constant is called ordinal. We can not change ordinal of enum constant.
- In java, using name, we can assign name to the any literal.
- Enum is a reference type in java.
- java.lang.Enum is abstract class introduced in jdk1.5. It is super class of all enums in java language.
- Methods of java.lang.Enum class:
    1. public final Class getDeclaringClass();
    2. public final String name();
    3. public final int ordinal();
    4. public static <T extends Enum > T valueOf(Class enumType, String name);

```
enum Color
{
    RED, GREEN, BLUE
}
```

- Above code is internally converted as follows:

```
final class Color extends Enum<Color>
{
  public static final Color RED;
  public static final Color GREEN;
  public static final Color BLUE;
  public static Color[] values();
```

```
    public static Color valueOf( String name );
}
```

- Since enum is implictly final class we can not extend it.
- If we pass illegal or inappropriate argument to the method then method throws IllegalArgumentException.
- If we want to assign name to the literals then it is mandatory to define constructor inside enum.

Package

- It is a java language feature which is used:

1. To avoid name clashing / collision / ambiguity
2. To group functionally related/equivalent types together

- Package can contain:

1. Sub Package
2. Interface
3. Class
4. Enum
5. Exception
6. Error
7. Annotation Types

- package is a keyword in java.

```
package p1;
class Complex
{
    //TODO : Member declaration
}
```

- If we want to define any type inside package then we must write package declaration statement in .java file.

```
package p1; //OK
//package p2; //Not OK
class Complex
{
    //TODO : Member declaration
}
//package p2; //Not OK
```

- package declaration statement must be first statement inside ".java" file.
- If we define any type inside package then it is called packaged type otherwise it is called unpackaged type.
- Package name is physically mapped to folder.

- If we want to use any type inside different package then we must use either F.Q. Type name or import statement.
- Default access modifier of any type in java is package level private.

```
package p1;
??? class CComplex  //??? is package level private
{   }
```

- If we want to use/access type in different package then access modifier of type must be public.
- Access modifier of type can be either package level private or public only.
- According to java language specification, name of public type and ".java" file must be same.
- We can use packaged class/type inside unpackaged class/type.
- If we define any type without package then it is considered as member of default package.
- We can import default package hence it is impossible to use unpackaged type inside packaged type.
- We can define types in different packages.
- We can define multiple types inside same package.
- Package name convention
    1. package name should be in lower case.
    2. "com.company_name.project_name"
    3. org.sunbeam.kdac
    4. com.mysql.cj.jdbc
- java.lang.Math is a final class which contains all static members for performing basic numeric operations.
- Without type name, if we want to access static members then we should use static import.

```
import static java.lang.System.out;
import static java.lang.Math.PI;
import static java.lang.Math.pow;
class Program
{
    public static void main(String[] args)
    {
        float radius = 10;
        float area = ( float )( PI * pow(radius, 2) );
        out.println("Area    :   "+area);
    }
}
```

Stream

- Stream is an abstraction(instance) which is used to produce(write) and consume(read) information from source to destination.
- Standard stream objets/instances of java associated with console/terminal

1. System.in -> Keyborad
2. System.out -> Monitor
3. System.err -> Error Stream associated with Monitor

- Scanner is a final class declared in java.util package.
- Instantiation:

```
    Scanner sc = new Scanner( System.in );
```

- Methods of Scanner class
    1. public String nextLine()
    2. public int nextInt()
    3. public float nextFloat()
    4. public double nextDouble()

```
Scanner sc = new Scanner(System.in);
System.out.print("Name  :   ");
String name = sc.nextLine();
System.out.print("Empid  :   ");
int empid = sc.nextInt();
System.out.print("Salary  :   ");
float salary = sc.nextFloat();
```

Array

- Value/data stored in data structure is called Element.
- Array is linear data structure/collection which is used to store elements of same type in continous memory location.
- If we want to access elements of array then we should use subscript operator and integer index.
- Array index always begins with zero.
- In java, checking array bounds is a job of JVM.
- Array is reference type i.e to create array instance it is mandatory to use new operator.
- Types of Array

1. Single dimensional Array
2. Multi dimensional Array
3. Ragged Array

Single dimensional Array

- reference declaration

```
int arr1[] = null   //OK
int [ arr2 ] = null   //Not OK
int[ ] arr3 = null   //OK
```

- Array Instantiation

```
int[] arr = new int[ 3 ];
```

- Using negative size, if we try to create array instance then JVM throws NegativeArraySizeException.

```
int[] arr = new int[ -3 ]; //NegativeArraySizeException
```

- If we want to process elements of array then we should use methods declared in java.util.Arrays class.

```
int[] arr = new int[ 3 ];       //OK
System.out.println(arr.toString());     //[I@7852e922
System.out.println(Arrays.toString(arr));//[0, 0, 0]
```

- Array Initialization

```
//int[] arr = new int[  ] { 10, 20, 30, 40, 50 };
int[] arr = { 10, 20, 30, 40, 50 };
for( int index = 0; index < arr.length; ++ index )
    System.out.println(arr[ index ]);
```

- Using illegal index, if we try to access elements of array then JVM throws ArrayIndexOutOfBoundsException.

```
int[] arr = new int[ ] { 10, 20, 30, 40, 50 };
int element = arr[ -1 ];// ArrayIndexOutOfBoundsException
//int element = arr[ arr.length ];ArrayIndexOutOfBoundsException
```

- If we want to sort elements of array then we should use Arrays.sort() method which implicitly use Dual Pivot Quicksort algorithm

```
int[] arr = new int[ ] { 50, 10, 40, 20, 30 };
Arrays.sort(arr);//Dual-Pivot Quicksort
```

- Types of loop

1. do-while
2. while
3. for
4. foreach loop (It is also called as iterator)

- foreach loop is forward and read only loop.

```
int[] arr = new int[ ] { 10, 20, 30, 40, 50 };
for (int element : arr)
    System.out.println(element);
```

Multi dimensional Array

- Array of array whose column size is same is called multi dimensional array.
- reference declaration

```
int arr1[][] = null;    //OK
int[] arr2[] = null;    //OK
int[][] arr3 = null;    //OK
```

- Instantiation

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

Ragged Array

- Array of array whose column size is different is called ragged array.

```
int[][] arr = new int[ 3 ][  ];
arr[ 0 ] = new int[ 4 ];
arr[ 1 ] = new int[ 2 ];
arr[ 2 ] = new int[ 3 ];
```

```
class LinkedListNode
{
private:
    int data;
    LinkedListNode *next;
public:
    LinkedListNode( int data )
    {
        this->data = data;
        this->next = NULL;
    }
    friend class LinkedList;
};
class LinkedList
{
private:
    LinkedListNode *head;
```

```
public:
    LinkedList( )
    {
        this->head = NULL;
    }
};
```

```
class LinkedListNode
{
    int data;
    LinkedListNode next;
    public LinkedListNode( int data )
    {
        this.data = data;
    }
};
class LinkedList
{
    private LinkedListNode head;
    public void addFirst( int data )
    {   }
    public void addLast( int data )
    {   }
    public void removeFirst( )
    {   }
    public void removeLast( )
    {   }
    public void printList( )
    {   }
}
class Program
{
    public static void main(String[] args)
    {
        LinkedList list = new LinkedList();
        list.addFirst( 10 );
        list.addLast( 20 );
        list.addLast( 30 );
        list.addLast( 40 );
        list.addLast( 50 );
        list.printList( );
    }
}
```