Agenda

- Revision
- Array
- Variable Arity Method
- method overloading
- Method Parameters
    - Pass by(value & reference)
- Static (static fields/method/block)
- static import
- ~~Singleton Design Pattern~~

Array (Demo01 & Demo02)

- It is a collection that stores similar type of data in contigious manner.
- The elements stored inside an array are accessed using index.
- In java, array is a reference type.
- Arrays are of 3 types in java
    1. Single Dimension Array
    2. MultiDimension Array
        - Array with more than 1 dimension is called as multidimensional array.
        - For multidimensional array the size of both the dimensions are fixed.
    3. Ragged Array
        - Array with more than 1 dimension is called as multidimensional array.
        - For multidimensional array if the size of second dimension is not fixed or vary then such an array is called as ragged array.
- Array can be of primitive type or it can be of non primitive type
- Primitive types

```
// single Dimension Array
int arr1[] = new int[5];

//MultiDimension
int arr2[][] = new int[2][3];

// Ragged Array
int arr3[][] = new int[2][];
arr3[0] = new int[2];
arr3[1] = new int[3];
```

- NonPrimitive Type

```
// Single Dimension Array
Point arr1[] = new Point[5];

//MultiDimensional Array
Point arr2[][] = new Point[2][3];

// Ragged Array
Point arr3[][] = new Point[2][];
arr3[0] = new Point[2];
arr3[1] = new Point[3];
```

Assignemnt Question - To-Do Compulsary

- Create an single Dimension Array of Point.
- Write accept method to take input for x and y axis inside point class.
- for the array use tradational for loop and for-each loop for creating and object and calling the accept method
- find out the difference between working of for-loop and for-each loop

Variable Arity/Argument Method (Demo 03)

- If we want to pass multiple/variable no of arguments of same type to the method then we can define an varaiable arity method
- In this method the syntax is defined as below

```
void add(int ... args){

}
```

- The parameter args is internally working as an array.
- the arguments passed to this method at the time of method call will be converted in an array and will be passed to it.
- we can use a for-loop or for-each loop to iterate over this array.

Method overloading

- Writing multiple methods with same name but with differnt signature is called as method overloading
- In method overloading return type is not considered.
- Method overloading can be done in below ways
    - 1. By changing the no of parameters

```
void multiply(int num1,int num2){
  System.out.println("Multiplication = "+ (num1*num2));
```

```
}

void multiply(int num1,int num2,int num3){
  System.out.println("Multiplication = "+ (num1*num2*num3));
}
```

- 2. By changing type of parameters

```
void square(int num){
  System.out.println("Square of int = "+ (num*num));
}


void square(double num){
  System.out.println("Square of double = "+ (num*num));
}
```

- 3. By Changing order of parameters

```
void division(int num1,double num2){
  System.out.println("Division = "+ (num1/num2));
}
 void division(double num1,int num2){
  System.out.println("Division = "+ (num1/num2));
}
```

- method overloading is an example of compile time polymorphism

Method Parameters (Demo04)

- We can pass arguments to the method that we call.
- For primitive types the arguments are always passed by value
- For non primitive types the arguments are always passed by reference
- In the pass by reference, the references gets copied.
- when the references are copied then both the references point at the same object.
- if changes in the object is done by using any reference then it changes the original object values.
- so using any refenrece that point at that object, if we try to read/display the state, we will always get the updated state of the object.

Static (static fields/method/block) (Demo05)

- static means shared varaibles.
- The static fields are designed to shared across multiple objects.
- In java we cannot make local variables as static
- In java , we can make

    - 1. Field as static
    - 2. method as static
    - 3. block as static
    - 4. import static

1. Static Field

- We can declare fields inside the class as static
- Static fields get the memory on the method area
- memory is allocated at the time of class loading.
- static fields are designed to be shared across multiple objects.
- Static fields can be initialized using
    - 1. static field initializer
    - 2. static block
- static fields are designed to be accessed on classname using . operator.
- The static fields can also be accessed on object of the class using . operator.
- It is however dicouraged to use object of the class to access as it creates the confusion between static and non fields of the class.

2. Static Block

- These blocks are used to initialize the static fields of the class
- It is a block simiar to the object initaizer block but with static keyword
- It gets executed only once at the time of class loading
- If multiple static blocks are present then they will be executed in the same sequence in which they are declared.

3. Static Method

- These methods are desigend to be accessed on classname using . operator
- The static methods can also be accessed on object of the class using . operator.
- However it is discouraged to access them on class object as it creates confusion between static and non static methods.
- static methods cannot accsess non static fields of the class, as it do not get this reference.
- static methods can access only static fields of the class.
- static methods are generally used to desing factory methods.

4. Static import

- In java we can access all the static methods directly inside another static methods of the same class.
- if we want to acess the static methods of differnt class directly without using the classname and . operator then use static import.

```
//import static com.sunbeam.BankAccount.changeRoi;
//OR
import static com.sunbeam.BankAccount.*;
```

Lab Work

- Arrays