# Assignment Day-3

## Core Java with DS and Algorithms

**Name : Joshnitha Rangolu**

### Task 1: Arrays - Declaration, Initialization, and Usage

**Create a program that declares an array of integers, initializes it with consecutive numbers, and prints the array in reverse order.**

```
package day3;

public class ReverseArray {

public static void main(String[] args) {

int[] myArray = new int[10];

for (int i = 0; i < myArray.length; i++) {

myArray[i] = i;

}

System.out.println("Array in reverse order");

for (int i = myArray.length - 1; i >= 0; i--) {

System.out.println(myArray[i]);

}

}

}
```

```java
package day3;

public class ReverseArray {

    public static void main(String[] args) {
        int[] myArray = new int[10];
        for (int i = 0; i < myArray.length; i++) {
          myArray[i] = i;
        }
        System.out.println("Array in reverse order");
        for (int i = myArray.length - 1; i >= 0; i--) {
          System.out.println(myArray[i]);
        }
      }

}
```

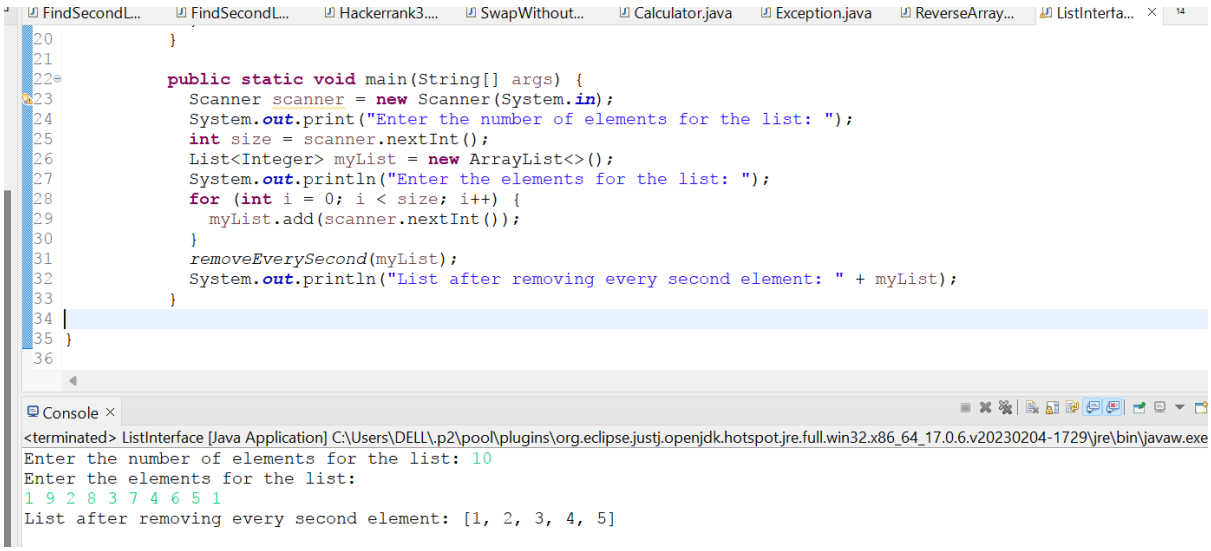
```
Console ×
<terminated> ReverseArray [Java Application] C:\Users\DELL\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v20230204-1729\jre\
Array in reverse order:
9
8
7
6
5
4
3
2
1
0
```

## Task 2: List interface

**Implement a method that takes a List as an argument and removes every second element from the list, then prints the resulting list.**

```java
package day3;

import java.util.ArrayList;

import java.util.Iterator;

import java.util.List;

import java.util.Scanner;

public class ListInterface {

public static void removeEverySecond(List<Integer> myList) {

Iterator<Integer> it = myList.iterator();

int index = 1;

while (it.hasNext()) {

it.next();

if (index % 2 == 0) {

it.remove();

}

index++;

}
```

```
}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the number of elements for the list: ");

int size = scanner.nextInt();

List<Integer> myList = new ArrayList<>();

System.out.println("Enter the elements for the list: ");

for (int i = 0; i < size; i++) {

myList.add(scanner.nextInt());

}

removeEverySecond(myList);

System.out.println("List after removing every second element: " + myList);

}

}
```

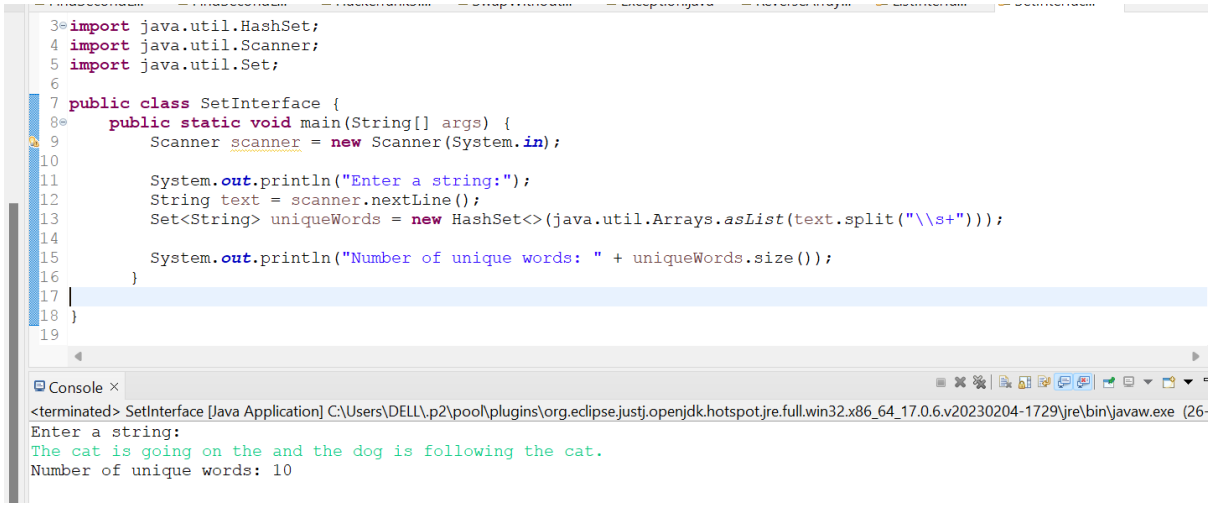



## Task 3: Set interface

**Write a program that reads words from a String variable into a Set and prints out the number of unique words, demonstrating the unique property of sets.**

```
package day3;

import java.util.HashSet;
```

```java
import java.util.Scanner;

import java.util.Set;

public class SetInterface {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter a string:");

String text = scanner.nextLine();

Set<String> uniqueWords = new
HashSet<>(java.util.Arrays.asList(text.split("\\s+")));

System.out.println("Number of unique words: " + uniqueWords.size());

}

}
```




## Task 4: Map interface

**Create a Java class that uses a Map to store the frequency of each word that appears in a given string.**

```java
package day3;

import java.util.HashMap;

import java.util.Map;

import java.util.Scanner;

public class MapInterface {

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

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

System.out.println("Enter a string:");

String text = scanner.nextLine();

Map<String, Integer> wordCount = new HashMap<>();

String[] words = text.split("\\s+");

for (String word : words) {

if (wordCount.containsKey(word)) {

wordCount.put(word, wordCount.get(word) + 1);

} else {

wordCount.put(word, 1);

}

}

System.out.println("Word Frequencies:");

for (Map.Entry<String, Integer> entry : wordCount.entrySet()) {

System.out.println(entry.getKey() + ": " + entry.getValue());

}

}

}
```

```java
        Map<String, Integer> wordCount = new HashMap<>();
        String[] words = text.split("\\s+");
        for (String word : words) {
          if (wordCount.containsKey(word)) {
            wordCount.put(word, wordCount.get(word) + 1);
          } else {
            wordCount.put(word, 1);
          }
        }
        System.out.println("Word Frequencies:");
        for (Map.Entry<String, Integer> entry : wordCount.entrySet()) {
          System.out.println(entry.getKey() + ": " + entry.getValue());
        }
```

```
<terminated> MapInterface [Java Application] C:\Users\DELL\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v20230204-1729\jre\bin\java
Enter a string:
The cat is going on the road and dog is following the cat.
Word Frequencies:
The: 1
the: 2
going: 1
road: 1
and: 1
cat.: 1
cat: 1
following: 1
is: 2
dog: 1
on: 1
```

## Task 5: Iterators and Comparators

**Write a custom Comparator to sort a list of Employee objects by their salary and then by name if the salary is the same.**

```java
package day3;

import java.util.ArrayList;

import java.util.Collections;

import java.util.Comparator;

import java.util.List;

public class EmployeeSort {

public static void main(String[] args) {

class Employee {

private String name;

private double salary;

public Employee(String name, double salary) {

this.name = name;

this.salary = salary;
```

```
}
public String getName() {
return name;
}
public double getSalary() {
return salary;
}
@Override
public String toString() {
return "Employee{name='" + name + "', salary=" + salary + "}";
}
}
class EmployeeSalaryNameComparator implements Comparator<Employee> {
@Override
public int compare(Employee e1, Employee e2) {
int salaryComparison = Double.compare(e1.getSalary(), e2.getSalary());
if (salaryComparison != 0) {
return salaryComparison;
} else {
return e1.getName().compareTo(e2.getName());
}
}
}
List<Employee> employees = new ArrayList<>();
employees.add(new Employee("John", 50000));
employees.add(new Employee("Alice", 75000));
employees.add(new Employee("Bob", 50000));
employees.add(new Employee("David", 60000));
```

```java
employees.add(new Employee("Carol", 75000));

System.out.println("Before Sorting:");

for (Employee e : employees) {

System.out.println(e);

}

Collections.sort(employees, new EmployeeSalaryNameComparator());

System.out.println("\nAfter Sorting:");

for (Employee e : employees) {

System.out.println(e);

}

}

}
```

```java
public class EmployeeSort {
    public static void main(String[] args) {
        class Employee {
            private String name;
            private double salary;

            public Employee(String name, double salary) {
                this.name = name;
                this.salary = salary;
            }

            public String getName() {
                return name;
            }
```

Console

```
<terminated> EmployeeSort [Java Application] C:\Users\DELL\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v20230204-1729\jre\bin\ja
Before Sorting:
Employee{name='John', salary=50000.0}
Employee{name='Alice', salary=75000.0}
Employee{name='Bob', salary=50000.0}
Employee{name='David', salary=60000.0}
Employee{name='Carol', salary=75000.0}

After Sorting:
Employee{name='Bob', salary=50000.0}
Employee{name='John', salary=50000.0}
Employee{name='David', salary=60000.0}
Employee{name='Alice', salary=75000.0}
Employee{name='Carol', salary=75000.0}
```