

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]);

        }

    }

}
```

```
1 package day3;
2
3 public class ReverseArray {
4
5     public static void main(String[] args) {
6         int[] myArray = new int[10];
7         for (int i = 0; i < myArray.length; i++) {
8             myArray[i] = i;
9         }
10        System.out.println("Array in reverse order");
11        for (int i = myArray.length - 1; i >= 0; i--) {
12            System.out.println(myArray[i]);
13        }
14    }
15 }
16 }
17 }
```

Console x

<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.

```
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());

}

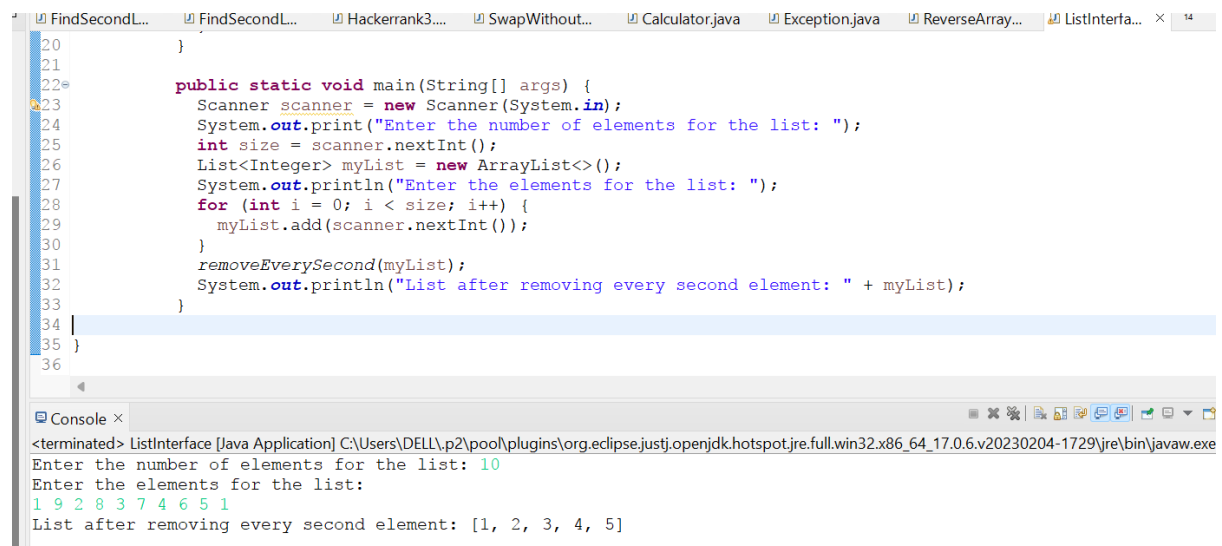
removeEverySecond(myList);

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

}

}

```



The screenshot shows an IDE with a Java file named 'ListInterface.java'. The code is the same as shown in the previous block. The console output shows the program running successfully. The user entered 10 for the number of elements, and then entered the elements 1, 9, 2, 8, 3, 7, 4, 6, 5, 1. The program then prints the list after removing every second element: [1, 2, 3, 4, 5].

```

20     }
21
22     public static void main(String[] args) {
23         Scanner scanner = new Scanner(System.in);
24         System.out.print("Enter the number of elements for the list: ");
25         int size = scanner.nextInt();
26         List<Integer> myList = new ArrayList<>();
27         System.out.println("Enter the elements for the list: ");
28         for (int i = 0; i < size; i++) {
29             myList.add(scanner.nextInt());
30         }
31         removeEverySecond(myList);
32         System.out.println("List after removing every second element: " + myList);
33     }
34 }
35 }
36

```

```

<terminated> ListInterface [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\javaw.exe
Enter the number of elements for the list: 10
Enter the elements for the list:
1 9 2 8 3 7 4 6 5 1
List after removing every second element: [1, 2, 3, 4, 5]

```

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;

```

```

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();

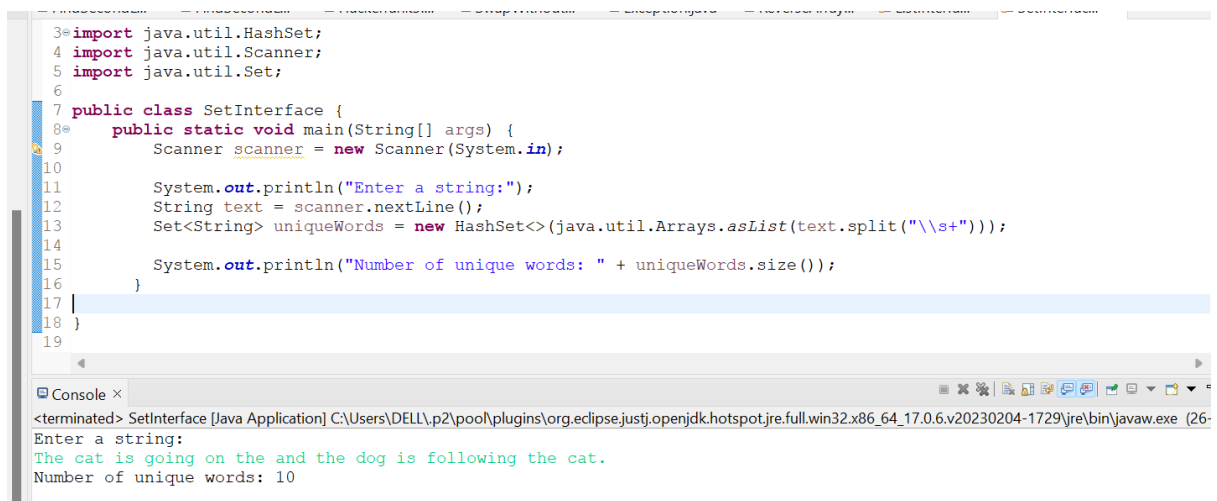
        Set<String> uniqueWords = new
        HashSet<>(java.util.Arrays.asList(text.split("\\s+")));

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

    }

}

```



The screenshot shows an IDE with a Java file named `SetInterface.java`. The code is identical to the one above. The console output at the bottom shows the program execution: it prompts for a string, receives "The cat is going on the and the dog is following the cat.", and outputs "Number of unique words: 10".

```

3=import java.util.HashSet;
4 import java.util.Scanner;
5 import java.util.Set;
6
7 public class SetInterface {
8     public static void main(String[] args) {
9         Scanner scanner = new Scanner(System.in);
10
11         System.out.println("Enter a string:");
12         String text = scanner.nextLine();
13         Set<String> uniqueWords = new HashSet<>(java.util.Arrays.asList(text.split("\\s+")));
14
15         System.out.println("Number of unique words: " + uniqueWords.size());
16     }
17 }
18
19

```

Console ×

```

<terminated> SetInterface [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\javaw.exe (26-
Enter a string:
The cat is going on the and the dog is following the cat.
Number of unique words: 10

```

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.

```

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());

}

}
```

```
FindSecondL... FindSecondL... Hackerrank3... Exception.java ReverseArray... ListInterfa... SetInterfac... *MapInterfa... × "16
13 Map<String, Integer> wordCount = new HashMap<>();
14 String[] words = text.split("\\s+");
15 for (String word : words) {
16     if (wordCount.containsKey(word)) {
17         wordCount.put(word, wordCount.get(word) + 1);
18     } else {
19         wordCount.put(word, 1);
20     }
21 }
22 System.out.println("Word Frequencies:");
23 for (Map.Entry<String, Integer> entry : wordCount.entrySet()) {
24     System.out.println(entry.getKey() + ": " + entry.getValue());
25 }
26 }
```

```
Console ×
<terminated> MapInterface [Java Application] C:\Users\DELL\p2\pool\plugins\org.eclipse.justi.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.

```
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));

```

```

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);

}

}

}

}

```



The screenshot shows the Eclipse IDE with a Java project. The editor displays the `EmployeeSort` class, which contains a `main` method and an `Employee` class. The `main` method adds five `Employee` objects to a list, sorts them using `EmployeeSalaryNameComparator`, and prints them before and after sorting. The `Employee` class has private fields for `name` and `salary`, and a `getName` method.

The console output shows the state of the program:

```

<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\j...
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}

```