Sukkur IBA University

Data Structure Algorithm

Name: Tariq Mehmood

CMS ID: 023-23-0127

Semester: 3rd

Section: E

Submitted by: Sir Riaz Hussain

Lab No: 07

Q1

```
public class TestSorting {


  public static void main(String[] args) {
    int[] descendingArray = {9, 8, 7, 6, 5, 4, 3, 2, 1};
    int[] almostSortedArray = {1, 2, 3, 4, 6, 5, 7, 8, 9};


    System.out.println("Sorting Descending Array:");
    int[] arrayToSort = new int[descendingArray.length];


    System.arraycopy(descendingArray, 0, arrayToSort, 0, descendingArray.length);
    bubbleSort(arrayToSort);
    printArray(arrayToSort);


    System.arraycopy(descendingArray, 0, arrayToSort, 0, descendingArray.length);
    selectionSort(arrayToSort);
    printArray(arrayToSort);


    System.arraycopy(descendingArray, 0, arrayToSort, 0, descendingArray.length);
    insertionSort(arrayToSort);
    printArray(arrayToSort);


    System.out.println("\nSorting Almost Sorted Array:");
    arrayToSort = new int[almostSortedArray.length];


    System.arraycopy(almostSortedArray, 0, arrayToSort, 0, almostSortedArray.length);
    bubbleSort(arrayToSort);
    printArray(arrayToSort);


    System.arraycopy(almostSortedArray, 0, arrayToSort, 0, almostSortedArray.length);
    selectionSort(arrayToSort);
    printArray(arrayToSort);


    System.arraycopy(almostSortedArray, 0, arrayToSort, 0, almostSortedArray.length);
```

```
    insertionSort(arrayToSort);
    printArray(arrayToSort);
  }

  public static void bubbleSort(int[] array) {
    int n = array.length;
    for (int i = 0; i < n - 1; i++) {
      for (int j = 0; j < n - i - 1; j++) {
        if (array[j] > array[j + 1]) {
          int temp = array[j];
          array[j] = array[j + 1];
          array[j + 1] = temp;
        }
      }
    }
  }

  public static void selectionSort(int[] array) {
    int n = array.length;
    for (int i = 0; i < n - 1; i++) {
      int minIndex = i;
      for (int j = i + 1; j < n; j++) {
        if (array[j] < array[minIndex]) {
          minIndex = j;
        }
      }
      int temp = array[minIndex];
      array[minIndex] = array[i];
      array[i] = temp;
    }
  }

  public static void insertionSort(int[] array) {
    int n = array.length;
    for (int i = 1; i < n; i++) {
```

```
        int key = array[i];
        int j = i - 1;
        while (j >= 0 && array[j] > key) {
          array[j + 1] = array[j];
          j--;
        }
        array[j + 1] = key;
      }
    }

    public static void printArray(int[] array) {
      for (int i : array) {
        System.out.print(i + " ");
      }
      System.out.println();
    }
}
```

```
Sorting Descending Array:
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9

Sorting Almost Sorted Array:
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9

=== Code Execution Successful ===
```

Q2

```
public class Task2 {


   public static void main(String[] args) {
      int[] array1 = {5, 8, 12, 1, 7};
      int[] array2 = {4, 9, 3, 15, 6};

      System.out.println("Array 1: ");
      printArray(array1);

      System.out.println("Array 2: ");
      printArray(array2);

      int[] mergedArray = new int[array1.length + array2.length];

      for (int i = 0; i < array1.length; i++) {
         mergedArray[i] = array1[i];
      }

      for (int i = 0; i < array2.length; i++) {
         mergedArray[array1.length + i] = array2[i];
      }

      sortArray(mergedArray);

      System.out.println("Merged and Sorted Array: ");
      printArray(mergedArray);
```

```
    }

    public static void printArray(int[] array) {
        for (int i = 0; i < array.length; i++) {
            System.out.print(array[i] + " ");
        }
        System.out.println();
    }

    public static void sortArray(int[] array) {
        for (int i = 0; i < array.length - 1; i++) {
            for (int j = 0; j < array.length - i - 1; j++) {
                if (array[j] > array[j + 1]) {
                    int temp = array[j];
                    array[j] = array[j + 1];
                    array[j + 1] = temp;
                }
            }
        }
    }
}
```

```
[Running] cd "d:\BS computer Science\Semester 3\DSA\DSA Lab\Lab07\"
Array 1:
5 8 12 1 7
Array 2:
4 9 3 15 6
Merged and Sorted Array:
1 3 4 5 6 7 8 9 12 15
```

Q3

```
class Student{
    String name;
    String cmsid;
    double gpa;
    Student(String name,String cmsid,double gpa){
```

```
        this.name=name;
        this.cmsid=cmsid;
        this.gpa=gpa;
    }
    void print(){
        System.out.println("Name :"+name+"  CMS ID :"+cmsid+"  GPA  :"+gpa);
    }
}
public class Task3{
    public static void main(String[] args) {
        Student []student=new Student[3];
        student[0]=new Student("Tariq","23-023-0127",3.8);
        student[1]=new Student("Asad","23-023-0128",3.9);
        student[2]=new Student("GMM","23-023-0129",4.0);
        for(int i=0; i<student.length; i++){
            student[i].print();
        }
        TopperOrder(student);
        System.out.println("In Topper Order");
        for(int i=0; i<student.length; i++){
            student[i].print();
        }

    }
    public static void TopperOrder(Student student[]){
        for (int i=0; i<student.length-1; i++){
            for(int j=0;j<student.length-i-1; j++){
                if(student[j].gpa<student[j+1].gpa){
                    Student temp=student[j];
                    student[j]=student[j+1];
                    student[j+1]=temp;
                }
            }
        }
```

```
    }
}
```

```
java -cp /tmp/th3qfksaC9/Task3
Name :Tariq  CMS ID :23-023-0127  GPA  :3.8
Name :Asad  CMS ID :23-023-0128  GPA  :3.9
Name :GMM  CMS ID :23-023-0129  GPA  :4.0
In Topper Order
Name :GMM  CMS ID :23-023-0129  GPA  :4.0
Name :Asad  CMS ID :23-023-0128  GPA  :3.9
Name :Tariq  CMS ID :23-023-0127  GPA  :3.8

=== Code Execution Successful ===
```

Q4

```
class Node {
  int data;
  Node next;

  Node(int data) {
    this.data = data;
    this.next = null;
  }
}

class LinkedList {
  Node head;

  public void insert(int data) {
    Node newNode = new Node(data);
    if (head == null) {
      head = newNode;
      return;
    }
    Node current = head;
    while (current.next != null) {
```

```
            current = current.next;
        }
        current.next = newNode;
    }

    public void printList() {
        Node current = head;
        while (current != null) {
            System.out.print(current.data + " -> ");
            current = current.next;
        }
        System.out.println("null");
    }

    public void insertionSort() {
        if (head == null) {
            return;
        }
        Node sorted = null;
        Node current = head;
        while (current != null) {
            Node next = current.next;
            sorted = sortedInsert(sorted, current);
            current = next;
        }
        head = sorted;
    }

    private Node sortedInsert(Node sorted, Node newNode) {
        if (sorted == null || sorted.data >= newNode.data) {
            newNode.next = sorted;
            return newNode;
        } else {
            Node current = sorted;
            while (current.next != null && current.next.data < newNode.data) {
```

```
            current = current.next;
        }
        newNode.next = current.next;
        current.next = newNode;
      }
      return sorted;
    }
}

public class Task4 {
    public static void main(String[] args) {
        LinkedList list = new LinkedList();
        list.insert(4);
        list.insert(2);
        list.insert(5);
        list.insert(1);
        list.insert(3);

        System.out.println("Linked list before sorting:");
        list.printList();

        list.insertionSort();

        System.out.println("Linked list after sorting:");
        list.printList();
    }
}
```

```
java -cp /tmp/ichQxeCDvl/Task4
Linked list before sorting:
4 -> 2 -> 5 -> 1 -> 3 -> null
Linked list after sorting:
1 -> 2 -> 3 -> 4 -> 5 -> null

=== Code Execution Successful ===
```

Q5

```
public class Task5 {

    public static String reconstructSentence(String s) {

        String[] words = new String[9];

        int wordCount = 0;


        // Manually split the sentence into words

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

            StringBuilder word = new StringBuilder();

            while (i < s.length() && s.charAt(i) != ' ') {

                word.append(s.charAt(i));

                i++;

            }

            if (word.length() > 0) {

                words[wordCount++] = word.toString();

            }

        }


        // Bubble sort to sort words based on the last character (index)

        for (int i = 0; i < wordCount - 1; i++) {

            for (int j = 0; j < wordCount - 1 - i; j++) {

                if (words[j].charAt(words[j].length() - 1) > words[j + 1].charAt(words[j + 1].length() - 1)) {

                    String temp = words[j];

                    words[j] = words[j + 1];

                    words[j + 1] = temp;

                }

            }

        }


        // Build the original sentence

        StringBuilder originalSentence = new StringBuilder();

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

            // Remove the last character (index) from each word

            for (int j = 0; j < words[i].length() - 1; j++) {
```

```
            originalSentence.append(words[i].charAt(j));
        }
        if (i < wordCount - 1) {
            originalSentence.append(" ");
        }
    }

    return originalSentence.toString();
  }

  public static void main(String[] args) {
    String shuffledSentence = "sentence4 a3 is2 This1";
    String originalSentence = reconstructSentence(shuffledSentence);
    System.out.println(originalSentence);
  }
}
```

```
java -cp /tmp/mtPG3MGrCd/Task5
This is a sentence

=== Code Execution Successful ===
```

Q7

```
public class Task7{
    public static void main(String[] args) {
        int A[]={1,2,3,4,5,6};
        int O[]=new int[A.length];
        for(int i:A){
            System.out.print(i+"  ");
        }
        System.out.println();
        int even=0, odd=1;
        for(int i=0; i<A.length; i++){
            if(A[i]%2==0){
                O[even]=A[i];
                even=even+2;
```

```
            }
            else{
                O[odd]=A[i];
                odd=odd+2;
            }
        }
        System.out.println("Arrange order Even elements on Even index @ odd Elements on odd index ");
        for(int i:O){
            System.out.print(i+"  ");
        }
    }
}
```

```
java -cp /tmp/wdXMle9POW/Task7
1  2  3  4  5  6
Arrange order Even elements on Even index @ odd Elements on odd index
2  1  4  3  6  5
=== Code Execution Successful ===
```

Q9

```
class Node {
   int data;
   Node next;
   Node (int data) {
      this.data = data;
      this.next = null;
   }
}

public class Task9 {
   Node head1;
   Node head2;
   Node head3;

   public void addtoFrontL1(int data) {
```

```
    Node newNode = new Node(data);
    if (head1 == null) {
      head1 = newNode;
    } else {
      newNode.next = head1;
      head1 = newNode;
    }
  }

  public void addtoFrontL2(int data) {
    Node newNode = new Node(data);
    if (head2 == null) {
      head2 = newNode;
    } else {
      newNode.next = head2;
      head2 = newNode;
    }
  }

  public void mergeLists() {
    Node temp1 = head1;
    Node temp2 = head2;
    Node tail = null;

    while (temp1 != null && temp2 != null) {
      Node newNode;
      if (temp1.data <= temp2.data) {
        newNode = new Node(temp1.data);
        temp1 = temp1.next;
      } else {
        newNode = new Node(temp2.data);
        temp2 = temp2.next;
      }

      if (head3 == null) {
```

```
            head3 = newNode;
            tail = head3;
        } else {
            tail.next = newNode;
            tail = newNode;
        }
    }

    while (temp1 != null) {
        Node newNode = new Node(temp1.data);
        tail.next = newNode;
        tail = newNode;
        temp1 = temp1.next;
    }

    while (temp2 != null) {
        Node newNode = new Node(temp2.data);
        tail.next = newNode;
        tail = newNode;
        temp2 = temp2.next;
    }
}

public void displayMerged() {
    if (head3 == null) {
        System.out.println("Merged list is empty");
        return;
    }
    Node temp = head3;
    while (temp != null) {
        System.out.print(temp.data + " > ");
        temp = temp.next;
    }
    System.out.println("null");
}
```

Sukkur IBA

```
    public static void main(String[] args) {
        Task9 list = new Task9();
        list.addtoFrontL1(5);
        list.addtoFrontL1(20);
        list.addtoFrontL1(15);
        list.addtoFrontL2(10);
        list.addtoFrontL2(30);
        list.addtoFrontL2(25);
        list.mergeLists();
        list.displayMerged();
    }
}
```