**Java Developer Assignment**

1. Write a program for sorting the string according to the word character length and number of occurrences of a character.

Input - “readable content of a page when looking at its layout”

Output - “readable content looking layout page when its of at a”

2. Write a program to group by route according to fare without inbuilt libraries.

Input - Route Fare

13 10

13-c 15

342-R 10

146-Q 10

27 15

29-A 12

215-U 12

27-E1 15

13J 12

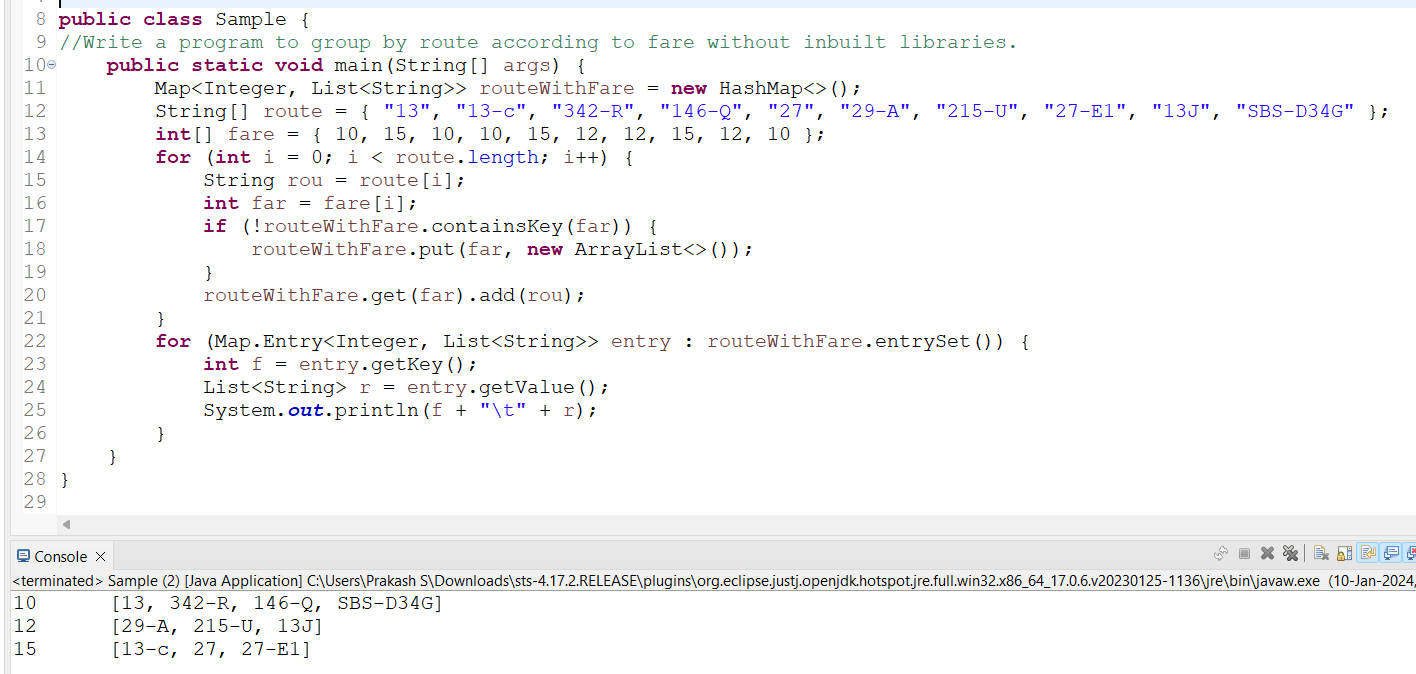
SBS-D34G 10

Expected Output - Fare Route

15 ["13-C","27","27-E1"]

10 ["13","342-R","146-Q","SBC-D34G"]

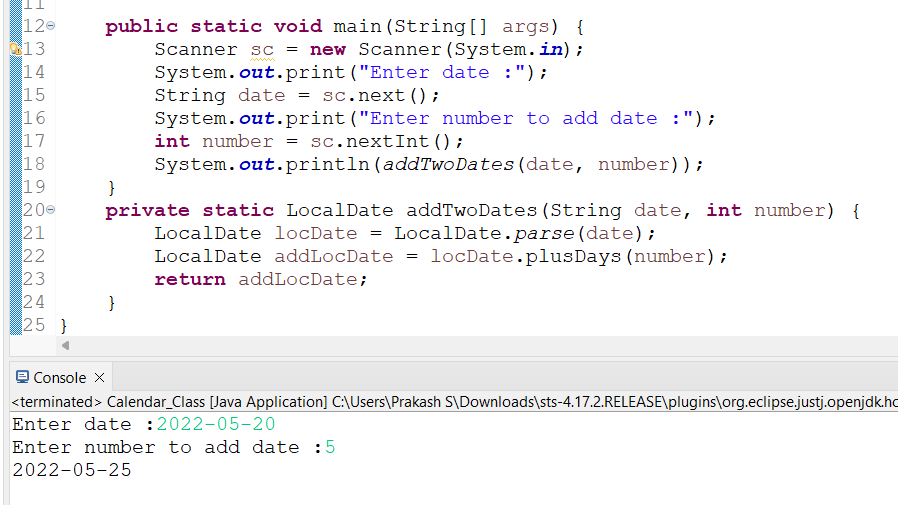
12 ["29-A","215-U","13J"]



3. Write a program to add some days in date which have been taken from command line arguments.

Input – date : 20-05-2022 and days : 5 // 30-05-2022 and days : 5

Output – 25-05-2022 // 04-06-2022



4. Write a program to achieve concurrency

5. Create query for result

Table – routes

|  |  |  |
| --- | --- | --- |
| id | name | number |
| 1 | 200-D | SBV-BHJ |
| 2 | 300-A | ASD-WE |

Table - route\_points

|  |  |  |  |
| --- | --- | --- | --- |
| id | route\_id | order | distance |
| 1 | 1 | 1 | 0 |
| 2 | 1 | 2 | 100 |
| 3 | 2 | 1 | 0 |
| 4 | 2 | 2 | 50 |
| 5 | 2 | 3 | 100 |

Expected Result

|  |  |  |
| --- | --- | --- |
| route\_id | name | total\_distance |
| 2 | 300-A | 150 |
| 1 | 200-D | 100 |

**Query :**

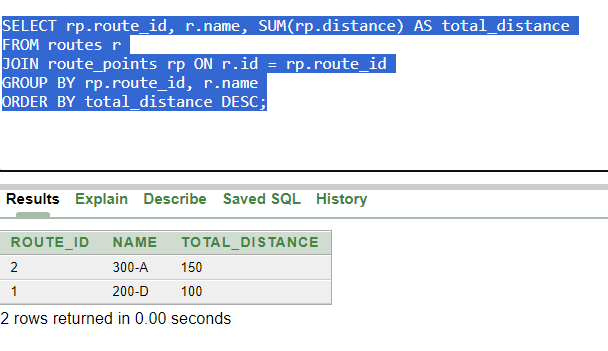
SELECT rp.route\_id, r.name, SUM(rp.distance) AS total\_distance

FROM routes r

JOIN route\_points rp ON r.id = rp.route\_id

GROUP BY rp.route\_id, r.name

ORDER BY total\_distance DESC;



6. Make a query for the result by station\_id and slot. Find all stations where slot 1 and time less than 7.45 and order by time.

Table – station

|  |  |
| --- | --- |
| id | station\_name |
| 1 | STA 1 |
| 2 | STB 2 |

Table – times

|  |  |  |  |
| --- | --- | --- | --- |
| id | station\_id | slot | time |
| 1 | 1 | 1 | 6:30 |
| 2 | 1 | 2 | 6:45 |
| 3 | 1 | 1 | 7:40 |
| 4 | 1 | 2 | 7:25 |
| 5 | 2 | 1 | 6:40 |
| 6 | 2 | 2 | 6:50 |
| 7 | 2 | 1 | 7:50 |
| 8 | 2 | 2 | 8:15 |

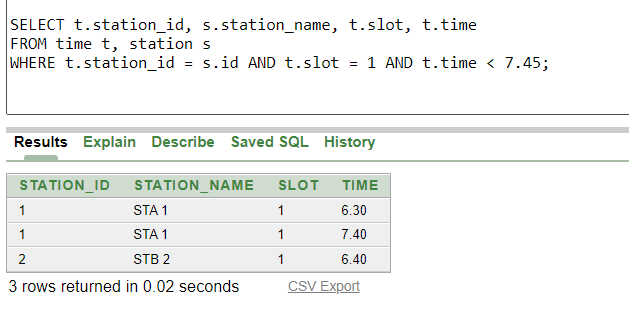
Require columns in output

|  |  |  |  |
| --- | --- | --- | --- |
| station\_id | station\_name | slot | time |

SELECT t.station\_id, s.station\_name, t.slot, t.time

FROM time t, station s

WHERE t.station\_id = s.id AND t.slot = 1 AND t.time < 7.45;



7. Make a query for get result for same route by stop id for source and destination

Table – routes

|  |  |  |
| --- | --- | --- |
| id | name | number |
| 1 | 200-D | SBV-BHJ |
| 2 | 300-D | ASD-WER |

Table - route\_points

|  |  |  |  |
| --- | --- | --- | --- |
| id | station\_id | order | stop\_id |
| 1 | 1 | 1 | 1 |
| 2 | 1 | 2 | 2 |
| 3 | 2 | 1 | 1 |
| 4 | 2 | 2 | 2 |
| 5 | 2 | 3 | 3 |

Expected Result

|  |  |  |
| --- | --- | --- |
| route\_id | source\_stop\_id | dest\_stop\_id |
| 1 | 1 | 2 |
| 2 | 1 | 3 |

Optional: If the above output is achieved. Please try the following

Table – station

|  |  |
| --- | --- |
| id | station\_name |
| 1 | STA 1 |
| 2 | STB 2 |
| 3 | STC 3 |

Expected Result

|  |  |  |
| --- | --- | --- |
| route\_id | source\_stop\_name | dest\_stop\_name |
| 1 | STA 1 | STB 2 |
| 2 | STA 1 | STC 3 |

8. Write a program to implement a linked list in Java and print in reverse order.

**package** com.assignment;

**class** Node {

**int** data;

Node next;

Node(**int** data) {

**this**.data = data;

**this**.next = **null**;

}

}

**class** LinkedList {

Node head;

LinkedList() {

**this**.head = **null**;

}

**public** **int** size = 0;

**public** **boolean** isEmpty() {

**return** head == **null** ? **true** : **false**;

}

**public** **void** insert(**int** data) {

Node num = **new** Node(data);

**if** (isEmpty()) {

head = num;

} **else** {

Node temp = head;

**while** (temp.next != **null**) {

temp = temp.next;

}

temp.next = num;

}

}

**public** **void** printReverseLinkedList(Node node) {

**if** (node == **null**) {

**return**;

}

printReverseLinkedList(node.next);

System.***out***.print(node.data + " ");

}

**public** **void** printLinkedList() {

Node temp = head;

**while** (temp != **null**) {

System.***out***.print(temp.data + " ");

temp = temp.next;

}

System.***out***.println();

}

}

**public** **class** Solutions {

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

LinkedList myList = **new** LinkedList();

myList.insert(10);

myList.insert(20);

myList.insert(30);

myList.insert(40);

myList.insert(50);

System.***out***.println("Original Linked List:");

myList.printLinkedList();

System.***out***.println("Reversed Linked List:");

myList.printReverseLinkedList(myList.head);

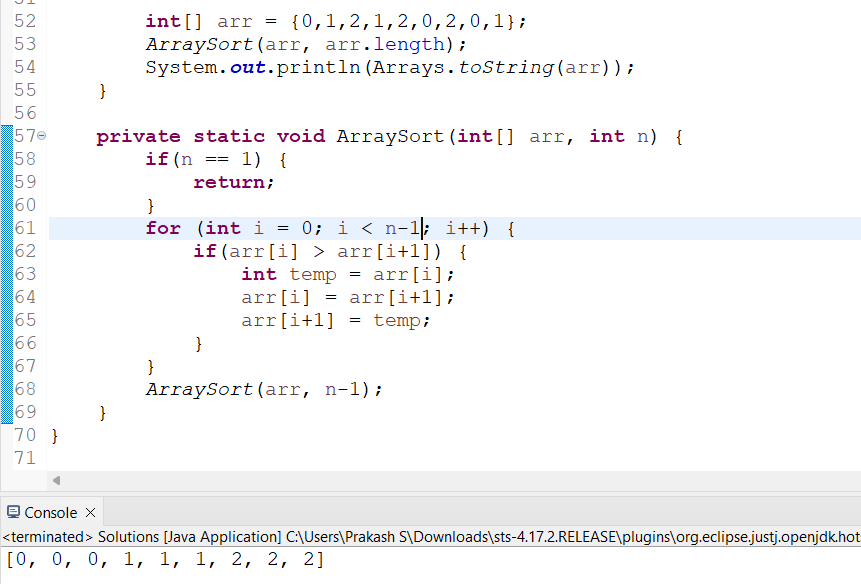
}

}

9. Write a program to sort the given array without any inbuilt functions to achieve O(n).

Input - [0,1,2,1,2,0,2,0,1]

Program :



Output – [0,0,0,1,1,1,2,2,2]

10. Give the output for the following and explain.

class Dog {

int leg;

}

class Main {

public static void main(String[] args) {

Dog d1 = new Dog(); // this is object creation of another class at reference d1

d1.leg = 4; // assign the value using object

Dog d2 = new Dog(); //and also another one object creation reference d2

d2.leg = 3;

swap(d1, d2); // call function and passing parameter obj d1, d2

System.out.println("d1.leg = "+d1.leg); // output : 4

System.out.println("d2.leg = "+d2.leg); // output : 3

modify(d1); // call function and passing the parameter object d1;

System.out.println("d1.leg = "+d1.leg); // output : 0

}

// this is the function swap the two values without return type

static void swap(Dog d1, Dog d2) {

Dog temp = d2; // temp : 3

d2 = d1; // d2 : 4

d1 = temp; // d1 = 3

}

// this is the function modify the values without return type

static void modify(Dog d1) {

d1.leg = 0;

} }