

Name: B Sree Rama Karthikeya

Regno: 19bce7637

DAA-Assaignment-4

Q) You are going to travel to another city that is located d miles away from your home city. You can travel at most m miles on a full tank and you start with a full tank. Along your way, there are gas stations at distances s_1, s_2, \dots, s_n from your home city. What is the minimum number of refills needed?

Problem Description

Input Format. The first line contains an integer d . The second line contains an integer m . The third line specifies an integer n . Finally, the last line contains integers s_1, s_2, \dots, s_n .

Input Format. Assuming that the distance between the cities is d miles, a car can travel at most m miles on a full tank, and there are gas stations at distances s_1, s_2, \dots, s_n along the way, output the minimum number of refills needed. Assume that the car starts with a full tank. If it is not possible to reach the destination, output -1 .

Constraints. $1 \leq d \leq 105$. $1 \leq m \leq 400$. $1 \leq n \leq 300$. $0 < s_1 < s_2 < \dots < s_n < m$.

Sample 1.

Input: 950 400 4 200 375 550 750

Output:

2

Code:

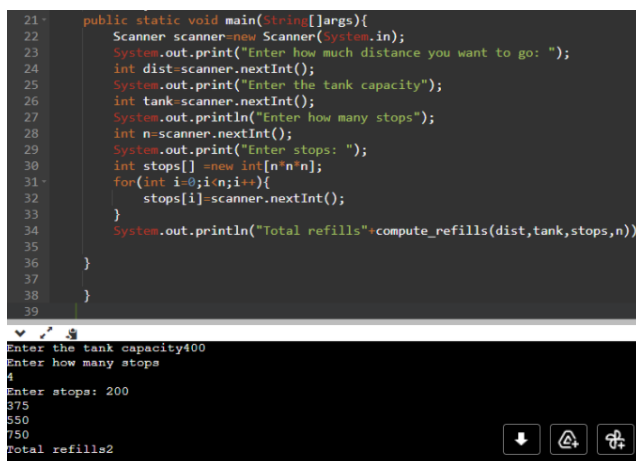
```
import java.util.*;
import java.io.*;
public class Main
{
    static int compute_refills(int dist,int tank,int stops[],int n){
        int current_refills=0;
        int num_refills=0;
        int last_refill=0;
        while(current_refills<n-1){
            last_refill=current_refills;
            while((current_refills <n-1)&&(stops[current_refills +1] - stops[last_refill])<=tank){
                current_refills=current_refills+1;
                break;}
            if (current_refills == last_refill)
                return -1;
            if(current_refills<n-1)
                num_refills=num_refills+1;}
        return num_refills;
    }
    public static void main(String[]args){
        Scanner scanner=new Scanner(System.in);
        System.out.print("Enter how much distance you want to go: ");
        int dist=scanner.nextInt();
```

```

        System.out.print("Enter the tank capacity");
        int tank=scanner.nextInt();
        System.out.println("Enter how many stops");
        int n=scanner.nextInt();
        System.out.print("Enter stops: ");
        int stops[] =new int[n*n*n];
        for(int i=0;i<n;i++){
            stops[i]=scanner.nextInt();
        }
        System.out.println("Total refills"+compute_refills(dist,tank,stops,n));
    }
}

```

Output:



```

21 public static void main(String[] args){
22     Scanner scanner=new Scanner(System.in);
23     System.out.print("Enter how much distance you want to go: ");
24     int dist=scanner.nextInt();
25     System.out.print("Enter the tank capacity");
26     int tank=scanner.nextInt();
27     System.out.println("Enter how many stops");
28     int n=scanner.nextInt();
29     System.out.print("Enter stops: ");
30     int stops[] =new int[n*n*n];
31     for(int i=0;i<n;i++){
32         stops[i]=scanner.nextInt();
33     }
34     System.out.println("Total refills"+compute_refills(dist,tank,stops,n));
35 }
36 }
37 }
38 }
39 }

```

Enter the tank capacity400
Enter how many stops
4
Enter stops: 200
875
550
750
Total refills2

Q7) Problem IntroductionAs the last question of a successful interview, your boss gives you a few pieces of paperwith numbers on it and asks you to compose a largest number from these numbers. The resulting number is going to be your salary, so you are very much interested in maximizingthis number. How can you do this?In the lectures, we considered the following algorithm for composing the largest number out of the givensingle-digit numbers.LargestNumber (Digits) : answer←empty stringwhileDigitsis not empty: maxDigit←−∞fordigitinDigits: ifdigit≥maxDigit: maxDigit←digitappendmaxDigittoanswerremove maxDigitfromDigitsreturnanswer

```
import java.util.*;
```

```

public class Main{
    public static String largestnumber(String[] salaryParts){
        int numParts = salaryParts.length;
        if(salaryParts == null || numParts == 0)
            return "";
        String[] maxSalary = new String[numParts];
        for(int i =0;i<numParts;++i) {
            maxSalary[i] =String.valueOf(salaryParts[i]);

```

```

    }
    Arrays.sort(maxSalary, (s1,s2)->(s2 +s1).compareTo(s1 + s2));

    StringBuilder sb = new StringBuilder();
    for(String salaryPart : maxSalary) {
        sb.append(salaryPart);
    }
    return sb.toString();
}

public static void main(String[] args){
    Scanner sc=new Scanner(System.in);
    int n=sc.nextInt();
    String[] salaryParts = new String[n];
    for(int i= 0;i<n;i++){
        salaryParts[i] =sc.next();
    }
    System.out.println(largestnumber(salaryParts));
}
}

```

```

12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30

```

```

2
21 2
221

...Program finished with exit code 0
Press ENTER to exit console.

```