LAB-4

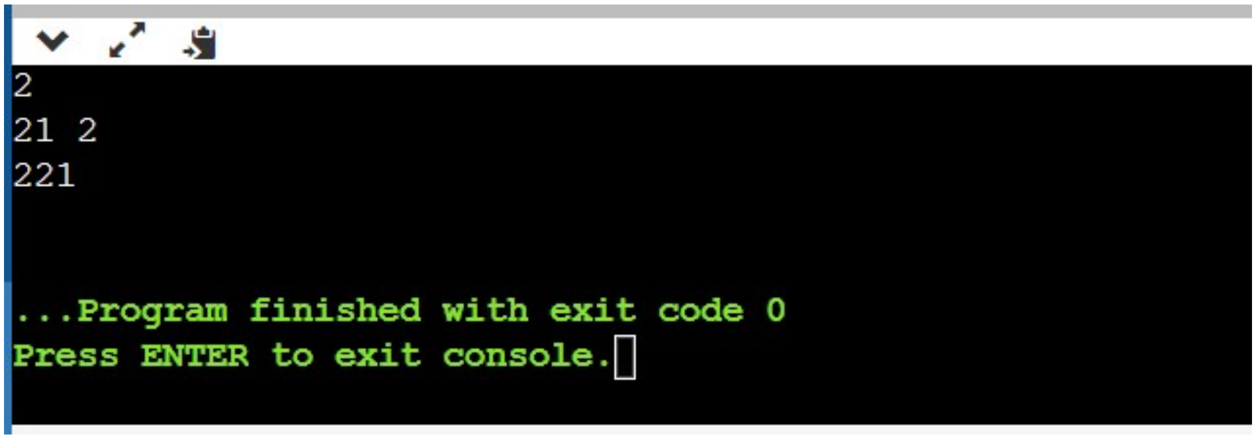
Name: K SASANK PRABHATH

Reg.No.: 19BCE7564

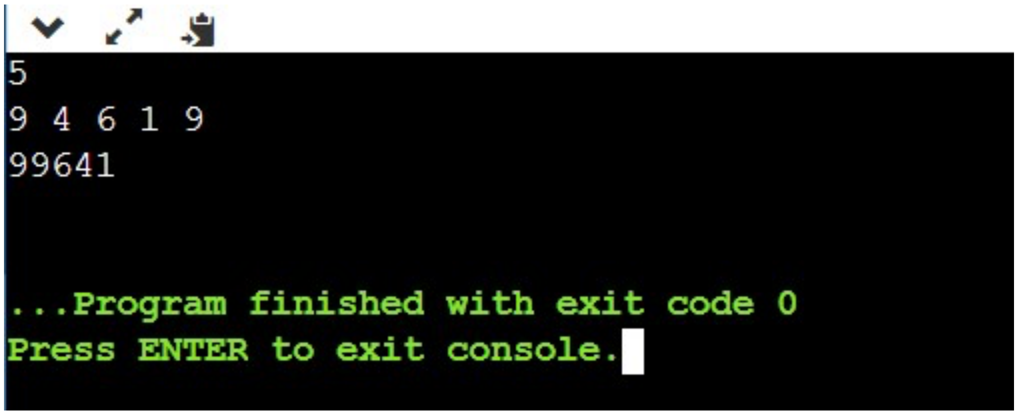
1.Maximum Salary

```
import java.util.*;

public class Main {
   private 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 scanner = new
      Scanner(System.in); int n =
      scanner.nextInt(); String[] salaryParts =
      new String[n]; for (int i = 0; i < n; i++) {
         salaryParts[i] = scanner.next();
      }
      System.out.println(largestNumber(salaryParts));
   }
}
```

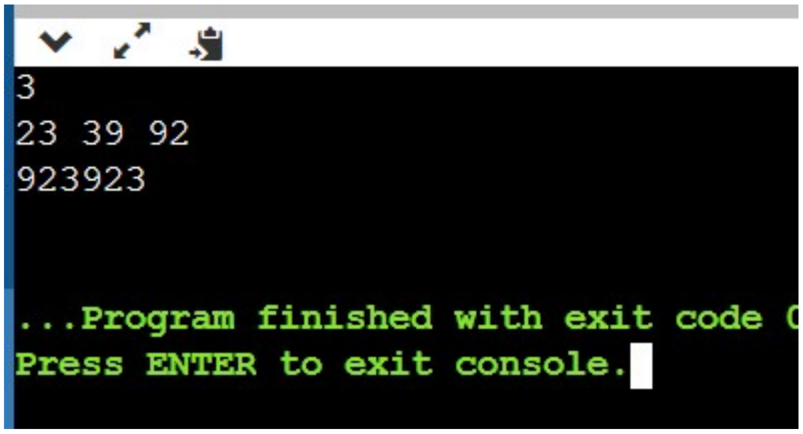
Output:



```
2
21 2
221


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



```
5
9 4 6 1 9
99641


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



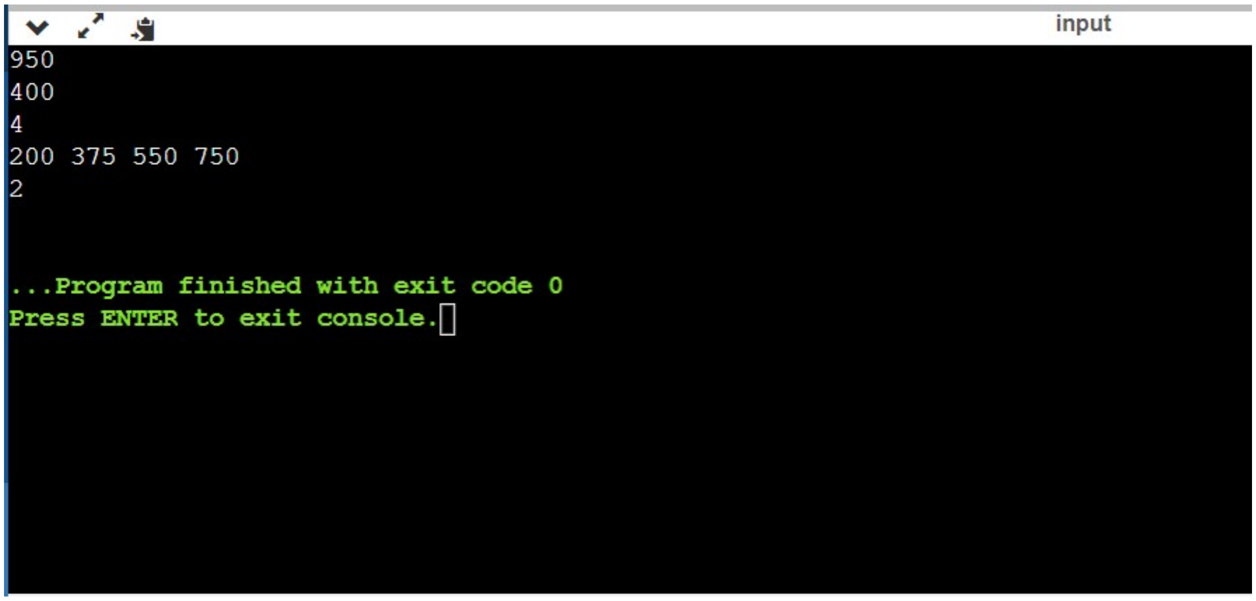
```
3
23 39 92
923923


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

2.Car fuelling problem:

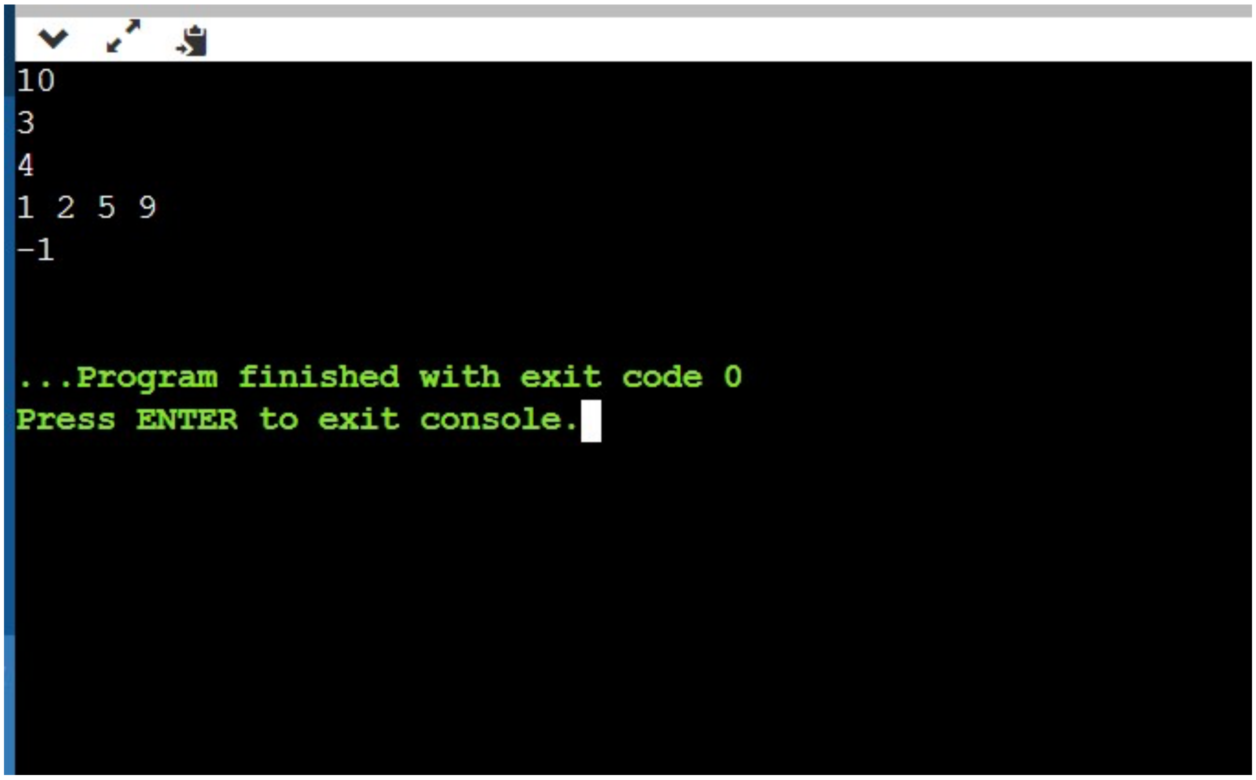
```
import java.util.*;
import java.lang.*;
import java.io.*;


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) {
    last_refill = current_refills;
      while ((current_refills <= n) && (stops[current_refills + 1] -
stops[last_refill]) <= tank) {
        current_refills = current_refills + 1;
      }

      if (current_refills == last_refill)
        return -1;
```

```
            if (current_refills <= n)
                num_refills = num_refills + 1;
        }
        return num_refills;
    }
    public static void main(String[] args) {
        Scanner scanner = new
        Scanner(System.in); int dist =
        scanner.nextInt(); int tank =
        scanner.nextInt(); int n = scanner.nextInt();
        int stops[] = new int[n+2]; stops[0] = 0;
        stops[n+1] = dist; for (int i = 1; i <= n; i++) {
            stops[i] = scanner.nextInt();
        }

       System.out.println(compute_refills(dist,tank,stops,n));
    }
}
```

Output:



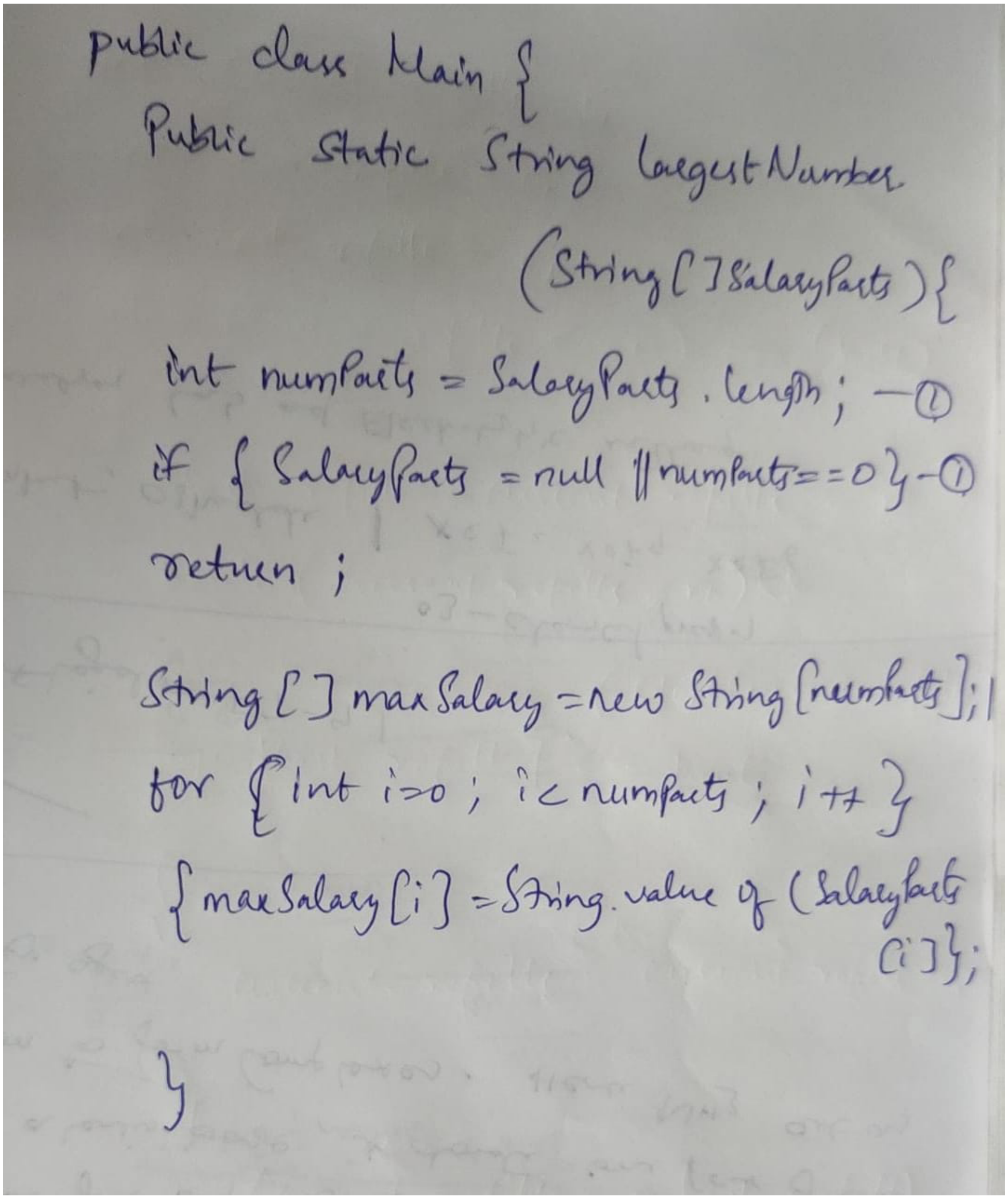
```
950
400
4
200 375 550 750
2


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



```
10
3
4
1 2 5 9
-1


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

Analysis:

1.Maximum Salary problem:



```
public class Main {
  Public Static String largestNumber
                    (String [] SalaryParts){
   int numParts = SalaryParts . length; —①
   if { SalaryParts = null || numParts==0 }—①
   return ;

   String [] maxSalary = new String [numParts];1
   for {int i=0 ; i<numParts ; i++ }
    {maxSalary [i] = String. value of (SalaryParts
                                        [i]};

   }
```

2. Car fuelling problem:

```
int  current_refills = 0;
int  num_refills = 0;
int  last_refills = 0;
while (current_refills < = n) — (n+1)
{
   last refill = current_refills — n
   while (current_refills < n) &&
                         (stops current_refills]
  stops [last_refill ] < = tank

  {
       current_refills = current_refills +1;

  }
  if (current_refills = = last_refill)

       return -1 ;

            n = 0 (n)
```