Design and Analysis of Algorithm Lab5

Name:V.Prasanna Vyshnavi

Regno:19BCE7661

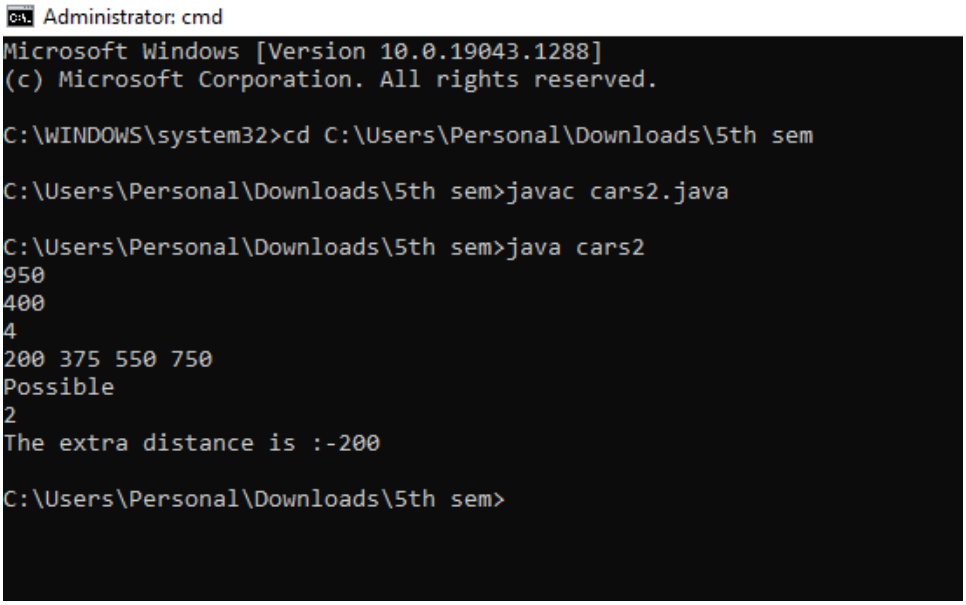
Car fueling sub problem:

Code:

```
import java.util.Scanner;
public class cars2 {
static int cars(int distance,int tank,int stops[],int n) {
int currentrefills=0;
int numrefills=0;
int lastrefills=0;
while(currentrefills<=n-1) {
lastrefills=currentrefills;
while((currentrefills<=n-1) && stops[currentrefills+1]-stops[lastrefills]<=tank) {
currentrefills=currentrefills+1;
}
if(currentrefills==lastrefills)
return -1;
if(currentrefills<=n)
numrefills=numrefills+1;
}
System.out.println("Possible");
return numrefills;
}
```

```
public static void main(String[] args) {
Scanner sc=new Scanner(System.in);
int distance=sc.nextInt();
int tank=sc.nextInt();
int n=sc.nextInt();
int stops[]=new int[n*n*n];
for(int i=0;i<n;i++) {
stops[i]=sc.nextInt();
}
System.out.println(cars(distance,tank,stops,n));
tank=distance-stops[3];
System.out.println("The extra distance is :-"+tank);
}
}
```

Output:



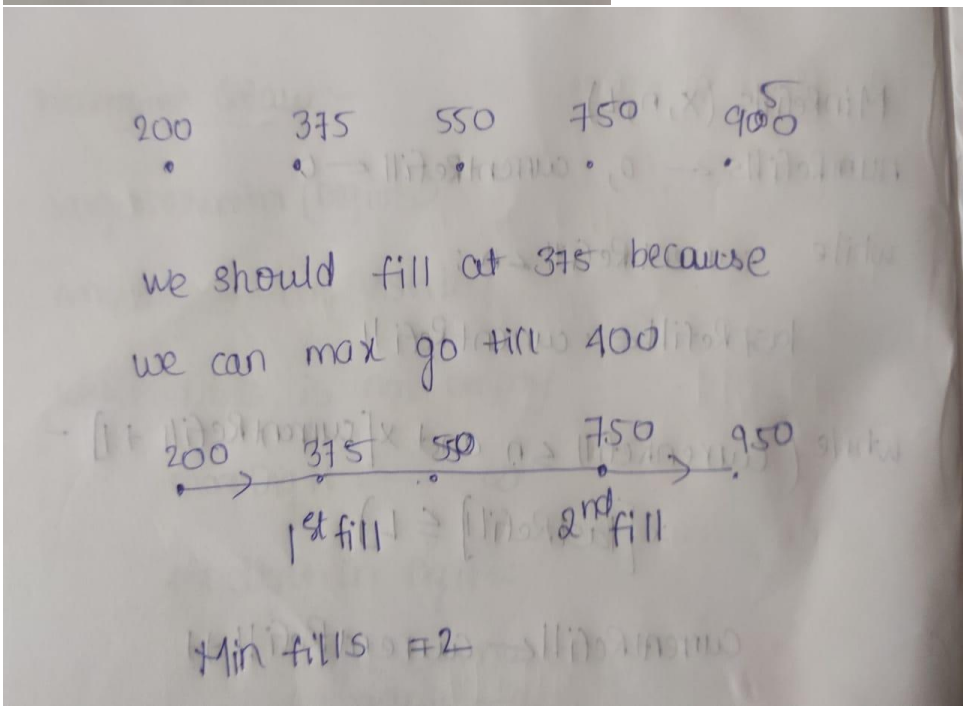
Asymptotic Analysis:

```
MinRefills (x, n, L)
numRefills ← 0, currentRefill ← 0
while currentRefill < n:
    lastRefill ← currentRefill
while (currentRefill ≤ n and x[currentRefill + 1] -
        x[lastRefill] ≤ L):
    currentRefill ← currentRefill + 1
if currentRefill == lastRefill:
    return -1
if currentRefill ≤ n:
    numRefills ← numRefills + 1
return numRefills
```

$T(n) = O(n)$

Input: 950
400
4
200 375 550 750

output: 2



To go back to 750 you need to travel 200 km we can still travell upto 400km we already completed 200km by going to 950 from 750 remaining 200 is enuff to go back to 750