Ben-Gurion University

Faculty of Engineering Sciences

Department of Electrical and Computer Engineering

Introduction to algorithms and data structures

Home assignment #2

Semester B / 2021

Instructions

These are the guidelines for submitting this home assignment:

1. The home assignment is submitted **only** through the moodle VPL! **Do not** send by email.
2. Do not ask questions regarding the project through email, Use the moodle forum only.
3. You should submit a single **.c** or **.cpp** file in the C/C++ VPL component.
4. The template is a .cpp file and you may use it (not mandatory), even if you are writing your project in proper C language!!
5. There are three problems you have to provide solutions to.
6. You do not have to check the input. You can assume the input is correct.
7. The program will print the solution to the screen. For floating point answers, print only the first two digits after the point.
8. Each problem will be tested with 8 different inputs.
9. **If your solution is not correct, or your algorithm times-out (more than 1 second) you will get 0 for the test input. Otherwise, 1.**
10. For each question write as a comment in your code a tight upper bound to your solution including an explanation for it.
11. The assignment is **solo**! Each student must submit his own file with his **ID number**!!. Deadline is in the moodle.
12. The inputs are entered by the user. You can use redirect values from a text file into your program from the command-line (Google it).
13. The first input will be the number of problem you want to solve.
14. A template file is attached, you are not allowed to add any other libraries,but you may implement by yourself in your file.
15. Don't use scanf_s! Make sure it compiles with gcc 9.3.0 , C++ 11- When submitting through the VPL you can check if your file compiled.
16. A few test cases have been entered into the VPL component, when submitting your file you must make sure your code complies with the given input and output.

Good Luck!

Problem 1

Find the K value of subarrays

You are given N integers that are arranged circularly (after the last number comes the first number). There are N ways to pick consecutive subsequences of length M (M < N). For any such subsequence we can find the "K"-value of that subsequence. "K"-value for a given subsequence is the Kth smallest number in that subsequence.

1 Task

Find the smallest "K"-value out of all possible subsequences.

1 < N <= 15,000

1 <= M < N

1 <= K <= M

0 <= any integer in the circle <= 2,147,483,647

2 Input

First line of the input will contain three integers N, M and K separated by spaces respectively.

Second line of the input will contain N integers separated by spaces in clockwise order starting from an arbitrary location.

3 Output

Output should contain only one integer, smallest "K"-value out of all possible subsequences.

4 Sample input

5 3 2

1 5 3 4 2

5 Sample output

2

5.1 Sample Explanation

2nd smallest of subsequence 1, 5, 3 is 3

2nd smallest of subsequence 5, 3, 4 is 4

2nd smallest of subsequence 3, 4, 2 is 3

2nd smallest of subsequence 4, 2, 1 is 2

2nd smallest of subsequence 2, 1, 5 is 2

Therefore the smallest “K”-value is 2.

Problem 2

The transplant list

The organ bank has an organ transplant list that prioritizes patients according to some algorithm.

The algorithm is: $\big((85 - AGE) + \big(\frac{3}{TTP}\big)\big)$, where AGE is the age of the patient, and TTP is the time to perish unless an organ is provided (value provided by the treating doctor).

The transplant list is a dynamic queue that helps the organ bank to decide which patient will get a donated organ.

0.1 Task

Write a program that manages a queue patients. The program can add new patients to the queue, remove patients from the queue and update patients in the queue.

Removing a patient from the queue can occur for two reasons:

1. Patient no longer needs a transplant.
2. An organ arrived and the first patient in the queue is removed.

1 Input

First line contains the maximum number of patients in queue, N, an integer K, and the number of tasks M.

5<=N<50,000

1<K<N - Used for printing the ID of the Kth patient in the queue.

1<M<= N

Each line after is a task index with its arguments if necessary. Argument can be a person's ID, or person's ID and age and TTP values.

The person ID is a 9 digit integer.

T - task index. An integer between 1-5.

T = 1:

Add a patient to the queue.

Line has the following construct:

1 ID_OF_PATIENT AGE TTP

T = 2:

Update patient in the queue. You can assume patient will be somewhere in the queue.

Line has the following construct:

2 ID_OF_PATIENT AGE TTP

T = 3:

Remove patient from the queue.

Line has the following construct:

3 ID_OF_PATIENT

T = 4:

Hallelujah!! an organ arrived for transplant. Remove first patient in the queue.

Line has the following construct:

4

T = 5:

Print the Kth ID_OF_PATIENT in the queue, and exit the program.
This operation will always be the last to come, and will only occur once.
Line has the following construct:
5

2 Output

Print the ID of the Kth patient in the queue. If queue is smaller than K - print 0.

3 Sample input

```
10 2 8
    1 222000212 60 1
    1 300100000 32 0.2
    1 300011002 39 0.03
    1 200000000 25 0.2
    3 200000000
    2 300100000 32 0.03
    4
    5
```

4 Sample output

```
222000212
```

5 Sample explained

1. 222000212(28.0)
2. 300100000(68.0) <-222000212(28.0)
3. 300011002(146.0) <-300100000(68.0) <-222000212(28.0)
4. 300011002(146.0) <-200000000(75.0)<-300100000(68.0) <-222000212(28.0)
5. 300011002(146.0) <-300100000(68.0) <-222000212(28.0)
6. 300100000(153.0)<-300011002(146.0) <-222000212(28.0)
7. 300011002(146.0) <-222000212(28.0)
8. 222000212

Problem 3

Sort IP Addresses

1 Task

Given a list of IP addresses, sort the list in descending order and print the sum of the K'th IP address values.

2 Input

The first line contains two integers:

$1 \leq N \leq 1,000,000$ - The number of IP addresses.

$1 \leq K \leq N$ - The Kth IP address of the sorted list.

Every line after that is a '.' separated 32bit IP address as follows: xxx.xxx.xxx.xxx.

3 Output

The sum of the K'th IP address values.

For example, if the K IP address is 192.168.1.1 you should return: 192+168+1+1=362.

4 Sample input

5 2
192.168.1.1
127.0.0.1
255.255.255.0
192.168.2.1
127.0.1.1

5 Sample output

363

6 Sample explained

After sorting the 5 IP addresses we get
255.255.255.0
192.168.2.1
192.168.1.1
127.0.1.1
127.0.0.1
The 2nd IP is 192.168.2.1, and its sum is 192+168+2+1=363.