Algorithms

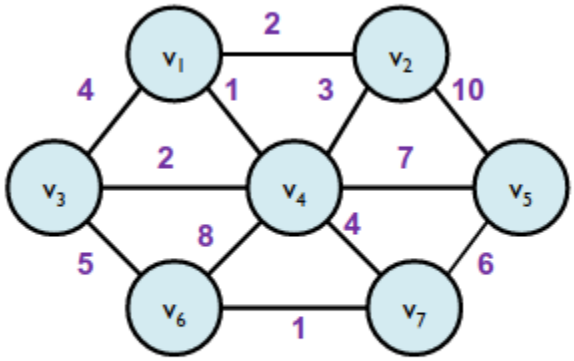
Assignment #2 (Deadline 05/01/2022)

Guidelines:

- Solve 5 problems and any extras are a bonus
- The groups of this assignment are the same as assignment 1's groups
- Please submit your solutions via blackboard
- You should submit a zip file (name this file with your ids: A1_ID1_ID2_ID3_ID4.zip) with all files (codes) for each problem (name code file p1 , p2 , .. etc)

Question (1):

Given Graph G:



Determine the MST, using Prim's starting with vertexV1:

Please write a mst Prim's algorithm to solve the following problem with the same input and print the output graph including nodes and vertices.

Question (2):

The ackermann function is one of those very difficult functions to compute and it's a prime example of non-primitive recursive functions, meaning it can't be de-recursed and rewritten into a loop (Check this video for a fun watch). The naive recursive implementation suffers problems starting with ack(4, 1) and at the time of writing on a machine with a 4.2GHz base CPU clock speed and 3200GHz 32GBs of DDR3 RAM is not computable without further optimizations or settings applied.

The ackermann function is described using the equation below.

$$ack(m,n) = \begin{cases} n+1 & m=0 \\ ack(m-1,1) & m>0 \quad n=0 \\ ack(m-1,ack(m,n-1)) & m>0 \quad n>0 \end{cases}$$

You are required to

1. Describe why the base Ackermann function is so heavy to compute
2. Write an optimization to make it's calculation more efficient and possible for arguments starting Ack(4, 1) and above

Inputs: 2 integers m & n where m & n >= 0
Output: ack(m, n)

Sample Inputs:
3 3
4 1

Sample Outputs:
61
65533

Note: In the case you're not able to optimize it please also submit a file with the naive implementation describing at the very least the first step of the solution where you describe the problem with the Ackermann function itself.

Question (3):

Given an undirected weighted connected graph, find the Really Special SubTree in it. The Really Special SubTree is defined as a subgraph consisting of all the nodes in the graph and:

- There is only one exclusive path from a node to every other node.
- The subgraph is of minimum overall weight (sum of all edges) among all such subgraphs.
- No cycles are formed

To create the Really Special SubTree, always pick the edge with the smallest weight. Determine if including it will create a cycle. If so, ignore the edge. If there are edges of equal weight available:

- Choose the edge that minimizes the sum **u + v + wt** where u and v are vertices and **wt** is the edge weight.
- If there is still a collision, choose any of them.

Print the overall weight of the tree formed using the rules.
For example, given the following edges:

u	v	wt
1	2	2
2	3	3
3	1	5

First choose 1->2 at weight 2 . Next choose 2-> 3 at weight 3 . All nodes are connected without cycles for a total weight of $3 + 2 = 5$.

Function Description

Complete the Kruskals function in the editor below. It should return an integer that represents the total weight of the subtree formed.
kruskals has the following parameters:

- g_nodes: an integer that represents the number of nodes in the tree
- g_from: an array of integers that represent beginning edge node numbers
- g_to: an array of integers that represent ending edge node numbers
- g_weight: an array of integers that represent the weights of each edge

Input Format

The first line has 2 space-separated integers **g_nodes and g_edges** the number of nodes and edges in the graphs
The next **g_edges** line each consists of 3 space-separated integers g_from, g_to and g_weight, where g_from and g_to denote the 2 nodes between which the **undirected** edge exists and **g_weight** denotes the weight of the edge.

Output Format

Print a single integer denoting the total weight of the Really Special SubTree.

Sample input:
4 6
1 2 5
1 3 3
4 1 6
2 4 7
3 2 4
3 4 5
Sample Output
12

Question (4)

Given a set of activities and the starting and finishing time of each activity, find the maximum number of activities that can be performed by a single person assuming that a person can only work on a single activity at a time.

This problem is called the activity selection problem, which concerns the selection of non-conflicting activities to perform within a given time frame, given a set of activities each marked by a start and finish time.

Input:
{1, 4}, {3, 5}, {0, 6}, {5, 7}, {3, 8}, {5, 9}, {6, 10}, {8, 11}, {8, 12}, {2, 13}, {12, 14}

Output:
{1, 4}, {5, 7}, {8, 11}, {12, 14}

Question (5)

Given an array of size n such that each element contains either a 'P' for policeman or a 'T' for thief. Find the maximum number of thieves that can be caught by the police.
Keep in mind the following conditions :

1. Each policeman can catch only one thief.
2. A policeman cannot catch a thief who is more than K units away from him.

Example 1:

Input:

N = 5, K = 1

arr[] = {P, T, T, P, T}

Output: 2

Explanation: Maximum 2 thieves can be caught. First policeman catches the first thief and the second policeman can catch either the second or third thief.

Example 2:

Input:
N = 6, K = 2
arr[] = {T, T, P, P, T, P}
Output: 3
Explanation: Maximum 3 thieves can be caught.

Question (6)

There are n smtp servers connected by network cables. Each of the m cables connects two computers and has a certain latency measured in milliseconds required to send an email message. What is the **shortest** time required to send a message from server to server along a sequence of cables.

Input

The first line of input gives the number of cases, N. N test cases follow. Each one starts with a line containing n (number of servers), m (number of route), S (source) and T(destination). S != T. The next m lines (routes) will each contain 3 integers: 2 different servers (in the range [0, n-1]) that are connected by a bidirectional cable and the latency (w)

Output

For each test case, output the line Case #x:, followed by the number of milliseconds required to send a message from S to T. Print ‘unreachable, if there is no route from S to T

Sample Input

3

2 1 0 1

0 1 100

3 3 2 0

0 1 100

0 2 200

1 2 50

2 0 0 1

Sample Output

Case #1: 100

Case #2: 150

Case #3: unreachable

Question (7)

Alice wants to send an important message to Bob. Message $a = (a_1, ..., a_n)$ is a sequence of positive integers (*characters*).

To compress the message Alice wants to use binary Huffman coding. We recall that *binary Huffman code*, or *binary prefix code* is a function f, that maps each letter that appears in the string to some binary string (that is, string consisting of characters '0' and '1' only) such that for each pair of different characters a_i and a_j string $f(a_i)$ is not a prefix of $f(a_j)$ (and vice versa). The result of the encoding of the message $a_1, a_2, ..., a_n$ is the concatenation of the encoding of each character, that is the string $f(a_1)f(a_2)...f(a_n)$. Huffman codes are very useful, as the compressed message can be easily and uniquely decompressed, if the function f is given. Code is usually chosen in order to minimize the total length of the compressed message, i.e. the length of the string $f(a_1)f(a_2)...f(a_n)$.

Because of security issues Alice doesn't want to send the whole message. Instead, she picks some substrings of the message and wants to send them separately. For each of the given substrings $a_{l_i}...a_{r_i}$ she wants to know the minimum possible length of the Huffman coding. Help her solve this problem.

Input:

The first line of the input contains the single integer n ($1 \le n \le 100\,000$) — the length of the initial message. The second line contains n integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 100\,000$) — characters of the message.

Next line contains the single integer q ($1 \le q \le 100\,000$) — the number of queries.

Then follow q lines with queries descriptions. The i-th of these lines contains two integers l_i and r_i ($1 \le l_i \le r_i \le n$) — the position of the left and right ends of the i-th substring respectively. Positions are numbered from 1. Substrings may overlap in any way. The same substring may appear in the input more than once.

Output:

Print q lines. Each line should contain a single integer — the minimum possible length of the Huffman encoding of the substring $a_{l_i}...a_{r_i}$.

Example:

Input:

```
7
1 2 1 3 1 2 1
5
1 7
1 3
3 5
2 4
4 4
```

Output:

```
10
3
3
5
0
```

Note:

- In the first query, one of the optimal ways to encode the substring is to map 1 to "0", 2 to "10" and 3 to "11".
- It is correct to map the letter to the empty substring (as in the fifth query from the sample).

Question (8)

Given an array of jobs where every job has a deadline and associated profit if the job is finished before the deadline. It is also given that every job takes a single unit of time, so the minimum possible deadline for any job is 1.

How to maximize total profit if only one job can be scheduled at a time. Time Complexity less than exponential.

Examples:

Input:
Four Jobs with following deadlines and profits:

JobID	Deadline	Profit
a	4	20
b	1	10
c	1	40
d	1	30

Output:
Following is maximum profit sequence of jobs:
c, a

Input:
Five Jobs with following deadlines and profits:

JobID	Deadline	Profit
a	2	100
b	1	19
c	2	27
d	1	25
e	3	15

Output:
Following is maximum profit sequence of jobs:
c, a, e