National Institute of Technology Calicut
Department of Computer Science and Engineering
Fourth Semester B. Tech.(CSE)-Winter 2022-23
CS2094D Data Structures Laboratory
Assignment #3B

Submission deadline (on or before): 26.03.2023, 2:00 PM

Policies for Submission and Evaluation:

- You must submit your assignment in the Eduserver course page, on or before the submission deadline.
- Ensure that your programs will compile and execute without errors using gcc compiler.
- During the evaluation, failure to execute programs without compilation errors may lead to zero marks for that evaluation.
- Your submission will also be tested for plagiarism, by automated tools. In case your code fails to pass the test, you will be straightaway awarded zero marks for this assignment and considered by the examiner for awarding F grade in the course. Detection of ANY malpractice related to the lab course can lead to awarding an F grade in the course.

Naming Conventions for Submission

- Submit a single ZIP (.zip) file (do not submit in any other archived formats like .rar, .tar, .gz). The name of this file must be

ASSG<NUMBER>_<ROLLNO>_<BATCHNO>_<FIRST-NAME>.zip

(Example: $ASSG3B\_BxxyyyyCS\_CS01\_LAXMAN.zip$). DO NOT add any other files (like temporary files, input files, etc.) except your source code, into the zip archive.

- The source codes must be named as

ASSG<NUMBER>_<ROLLNO>_<BATCHNO>_<FIRST-NAME>_<PROGRAM-NUMBER>.c

(For example: $ASSG3B\_BxxyyyyCS\_CS01\_LAXMAN\_1.c$). If you do not conform to the above naming conventions, your submission might not be recognized by our automated tools, and hence will lead to a score of 0 marks for the submission. So, make sure that you follow the naming conventions.

Standard of Conduct

- Violation of academic integrity will be severely penalized. Each student is expected to adhere to high standards of ethical conduct, especially those related to cheating and plagiarism. Any submitted work MUST BE an individual effort. Any academic dishonesty will result in zero marks in the corresponding exam or evaluation and will be reported to the department council for record keeping and for permission to assign F grade in the course. The department policy on academic integrity can be found at: https://minerva.nitc.ac.in/?q=node/650.

QUESTIONS

1. Write a program to implement a BINOMIAL HEAP and perform the operations *insertion*, *deletion*, *extract_minimum* and *union*. Your program should contain the following functions:

   - MAKEHEAP() - Creates and returns a new heap H containing no elements.
   - INSERT(H, x) – Inserts a new node with key 'x' into the heap H.
   - MINIMUM(H) – Return the value of the smallest key in the heap H.
   - EXTRACTMIN(H) – Deletes the node with minimum key value from heap H and prints the deleted node.
   - DECREASEKEY(H, x, k) – If the node of H with key 'x' is at least 'k', then decreases the value of node with key 'x' by 'k'. Otherwise, it prints -1.
   - DELETE(H, x) - Deletes the node with key 'x' from the heap H. If node is present, it prints the deleted node else it prints -1.
   - UNION(H_1, H_2) - Create and return a new heap H that contains all the nodes of heaps H_1 and H_2. Heaps H_1 and H_2 are "destroyed" by this operation.

   Input Format:

   - Each line contains a character from 'i', 'j', 'm', 'x', 'r', 'd' and 'e' followed by at most one integer. The integers, if given, are in the range $[-10^6, 10^6]$.
   - i k - inserts k into the heap H_1.
   - j k - inserts k into the heap H_2.
   - d k - deletes the node with key k from the heap H_1 and prints the deleted node's key.
   - $p1$ - prints the binomial heap H_1.
   - $p2$ - prints the binomial heap H_2.
   - m - prints the minimum element in the binomial heap H_1 (Note:- In print function, level order traversal is to be used).
   - x - extracts and prints the minimum element from the heap H_1.
   - r y z - decreases the value of node with key y by z and print the new value in H_1.
   - u - combine two heaps (H_1 and H_2) and print a new heap.
   - e - 'exit' from the program.

   Output Format:

   - The output (if any) of each command should be printed on a separate line.

   Sample Input:

   ```
   i 10
   i 20
   i 30
   i 40
   i 50
   p 1
   m
   x
   p 1
   r 50 4
   p 1
   r 70 5
   j 60
   j 70
   j 80
   ```

p 2
u
e

Sample Output:

50 10 30 20 40
10
10
20 30 50 40
46
20 30 46 40
-1
80 60 70
10 50 30 20 60 80 40 70

2. Write a program that implements the DISJOINT-SET data structure using rooted forests. Also,write functions to implement the **ranked union and path compression** heuristics on your data structure. Your program should compute the efficiency of the DISJOINT-SET data structure FIND operation by applying neither, either or both of the heuristics. The efficiency is calculated by counting the total number of data accesses performed over the course of the program. Your program must support the following functions:

   - MAKESET(x) - creates a singleton set with element x.
   - FIND(x) - finds the representative of the set containing the element x.
   - UNION(x,y) - merges the sets containing elements x and y into a single set. The representative of the resultant set is assigned with FIND(x), unless the ranked union heuristic is used and the ranks of both FIND(x) and FIND(y) are different. Otherwise, the representative is assigned in accordance with the ranked union heuristic.

   Note that looking up an element in the data structure must be done in $\mathcal{O}(1)$ time.

   Input Format:

   - The input consists of multiple lines, each one containing a character from {'m', 'f', 'u', 's'} followed by zero, one or two integers separated by single space. The integer(s), if given, is in the range 0 to 10000.
     - Call the function MAKESET(x) if the input line contains the character 'm' followed by an integer x. Print -1 if x is already present in some set, and the value of x, otherwise.
     - Call the function FIND(x) if the input line contains the character 'f' followed by an integer x. Print the value of FIND(x) if x is present, and -1 if x is not present.
     - Call the function UNION(x,y) if the input line contains the character 'u' followed by space separated integers x and y. Print -1, without terminating, if either x or y isn't present in the disjoint set. Print FIND(x) itself if FIND(x)=FIND(Y). Otherwise, print the representative of the resultant set. The representative of the resultant set is assigned with FIND(x), unless the ranked union heuristic is used and the ranks of both FIND(x) and FIND(y) are different. Otherwise, the representative is assigned in accordance with the ranked union heuristic.
     - If the input line contains the character 's', print the number of data accesses performed by the FIND function by each of the data structures over the course of the program and terminate.

   Output Format:

   - The output consists of multiple lines of single space separated columns. The columns correspond to the following disjoint-set data structures:
     (a) with neither ranked union nor path compression applied.

(b) with only ranked union applied.
(c) with only path compression applied.
(d) with both ranked union and path compression applied.

- Each line in the output contains the output of the corresponding line in the input, after applying to the respective data structures.
- The last line of the output contains the number of data accesses performed by the FIND function by each of the data structures over the course of the program.

Sample Input

```
m 1
m 2
m 3
m 4
m 5
m 6
m 7
m 8
m 9
u 1 2
u 3 4
u 5 6
u 7 8
u 9 8
u 6 8
u 4 8
u 2 8
f 9
m 10
u 10 9
s
```

Sample Output

```
1
2
3
4
5
6
7
8
9
1 1 1 1
3 3 3 3
5 5 5 5
7 7 7 7
9 7 9 7
5 5 5 5
3 5 3 5
1 5 1 5
1 5 1 5
10
10 5 10 5
38 32 33 30
```

3. A company manager wants to implement an onsite project. For the implementation of the project,

several teams have to be formed. To formulate the teams, the manager collects preferences as pairs from the employees, which consists of a pair of employee IDs, EID. For each of the preferences (a, b), the emplyees a and b are made part of the same team. All the employees should take part in the selection process. One employee can give more than one preference in which he/she is part of. Based on the given preferences, formulate teams. Once the teams are created, several groups have to be sent out for data collection. It has to be ensured that each of the data collecting group consists of exactly two employees, from different teams.
Write a program using the disjoint set data structure to formulate the teams, print all possible data collecting groups and checks whether a given data collecting group is valid or not.
The output should print the following:

- The number of teams formed using the preferences.
- Count of all possible data collecting groups (x, y) such that $x < y$.
- 1 if the given pair is a valid data collecting group, -1 otherwise.

Note that EID ranges from 1 to 10^3 and all EIDs are unique. If there is only one team formed at the end, then print -1 as no data collecting groups can be obtained.

Input Format:

- Each line contains a character t, T, d, v and e, followed by at most two integers.
- t a b - create the team with EIDs a and b.
- T - Prints the number of teams formulated.
- d - Prints the count of all possible data collecting groups.
- v x y - checks whether data collecting group (x, y) is valid or not. If it is valid, print 1 else -1.
- e - 'exit' from the program.

Output Format:

- The output (if any) of each command should be printed on a separate line.

Input 1:
t 2 1
t 4 1
t 5 6
t 6 12
t 109 34
T
d
v 4 5
v 34 2
Output 1:
3
21
1
-1
Input 2:
t 657 566
T
d
v 566 657
v 200 205
Output 2:
1
-1
-1
-1