NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

CX1107 DS&A: Graph II **2021/22 Semester 1**

Assignment 7 & 8: Graphs

School of Computer Science and Engineering *Nanyang Technological University*

Q1 Write a function, BFS(), to find the shortest distance from vertex v to vertex w, in a directed graph. Vertices ranged from 1 to $|V|$. The distance is measured by the number of edges. If there is no path from v to w, then -1 is returned. You may assume that the input graph is always valid (no duplicate or any invalid link etc.). The function prototype is given as follows:

```
int BFS (Graph G, int v, int w);
```

Q2 Write a function, CC(), to determine if a directed graph is strongly connected or not. Vertices ranged from 1 to $|V|$. You may assume that the input graph is always valid (no duplicate or invalid link etc.).

```
int CC (Graph g);
```

Q3 Write a function, sumToC() to determine and print all possible sequences in ascending positive integers that are summed to give a positive integer C where $C < 50$. For example, if the input value for C is 6, the output should be

```
1 2 3
1 5
2 4
6
```

The function prototype is given as follows:

```
void sumToC(LinkedList* ll, int C, ArrayList* al);
```

The following structures are given for storing sequences:

```
typedef struct _arraynode
{
    int *itemArray;
    int sizeArray; //the size of a possible sequence
    struct _arraynode *next;
}ArrayNode;

typedef struct _arraylist{
   int size; //the number of possible sequences
   ArrayNode *head;
} ArrayList;

typedef struct _listnode
{
    int item;
  struct _listnode *next;
} ListNode;

typedef struct _linkedlist{
   int sum;
   int size;
   ListNode *head;
} LinkedList;
```

All possible sequences (arrays with different length) will be stored in an ArrayList. LinkedList and ListNode is used for finding a sequence. The following utility functions are provided. The details can refer to the given template:

```
int insertNode(LinkedList *ll, int index, int coin);
int removeNode(LinkedList *ll, int index);
ListNode *findNode(LinkedList ll, int index);
void removeAllItems(LinkedList *ll);
```

Hint: Using backtracking algorithm for finding all possible sequences.

Q4 In CX4321, each student will be assigned to a project and a mentor. Students has their own preferences but each project or mentor can only be assigned to one of the students. The course coordinator needs an algorithm to maximize the matching between students and projects and between students and mentors.

Only This Question: You are strongly encouraged not to use the given template.

Input: First line is number of students, **S**, number of projects, **P** and number of mentors, **M**. The following **S** lines indicate the corresponding students' The number of preference projects, the number of preference mentors, the IDs of preference projects [1,P] and the IDs of preference mentors, [1,M].

Output: The maximum number of matches

Sample Case:
Input:
4 3 3
2 2 1 3 2 1
2 2 2 3 1 2
2 2 1 3 1 2
2 1 1 3 3

Output:
3

In the given input, we have 4 students, 3 projects and 3 mentors (first line). Student 1 likes project 1 and project 3. Student 1 prefers to mentor 2 and mentor 1 (second line). The corresponding bipartite graph is as follow:

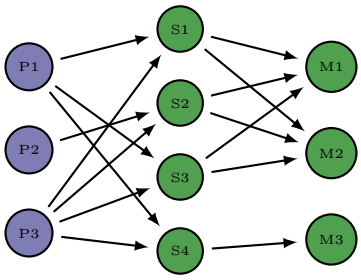

Figure 7.1: The graph of the given sample case: The Project indicates as Pi, the Student indicates as Sj, and the Mentor indicates as Mk.

Hint: You may not be able to use this graph to solve the problem directly. Think about some extreme cases like all students selected the same project and the same mentor.