CST 370 – Fall B 2020
Homework 2
Due: 11/10/2020 (Tuesday) (11:55 PM)

How to turn in?

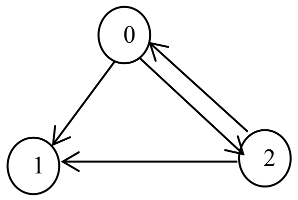
- Submit **three C++ or Java programs** on the iLearn. When you submit your homework programs, don't forget to include "Title", "Abstract", "ID", "Name", and "Date".
- Note that the **due time is 11:55(PM)**. This is the iLearn's timestamp, not your submission time. Since there could be a long delay between your computer and iLearn, you should **submit early**.

1. Write a program called **hw2_1.cpp (or hw2_1.java)** that converts a directed graph data from a user into a corresponding adjacency list format. Your program should read an input graph data from a user. Then, your program should convert it to the adjacency list format. In the assignment, you can **assume that the maximum number of vertices in the input graph is less than or equal to 50**.

Input format: This is a sample input from a user.

```
3
4
0 1
0 2
2 1
2 0
```

The first line (= 3 in the example) indicates that there are three vertices in the graph. The second line (= 4 in the example) presents the number of edges in the graph. The remaining four lines are the edge information in the graph. For the homework, you should assume that the first vertex starts from the number 0. Thus, **t1.txt** describes a directed graph like below:



Sample Run 0: Assume that the user typed the following lines

```
3
4
0 1
0 2
2 1
2 0
```

This is the correct output of your program.

```
0->1->2
1
2->0->1
```

Note that **the sequence of output values is important**. For example, when you display the neighbor vertices of the vertex 0, the sequence should be 1 and then 2 (= ascending order). If your program displays 2 and then 1, it's not correct. Similarly, for the vertex 2, your program should display 0 and then 1.

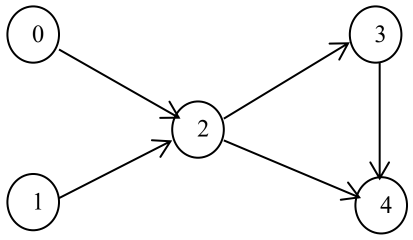
Sample Run 1: Assume that the user typed the following lines

```
5
5
2 3
3 4
1 2
0 2
2 4
```

This is the correct output of your program.

```
0->2
1->2
2->3->4
3->4
4
```

Note that this is the directed graph of the input data:



Sample Run 2: Assume that the user typed the following lines

```
3
6
0 1
1 0
1 2
2 1
2 0
```

```
0 2
```

This is the correct output of your program.

```
0->1->2
1->0->2
2->0->1
```

2. Go to https://www.geeksforgeeks.org/power-set/ and understand the operation of the program. Write a C++ program called **hw2_2.cpp (or hw2_2.java)** which has to modify the sample program in https://www.geeksforgeeks.org/power-set/

Your program should read a number of elements in a set and then the elements of the set. Then, your program should display all possible binary numbers and corresponding subsets one by one. For the program, you can assume that **the number of elements in a set is less than 10**.

Sample Run 0: Assume that the user typed the following input. Note that there are three elements in the set with the elements A, B, and C.

```
3
A B C
```

This is the correct output.

```
000:EMPTY
001:C
010:B
011:B C
100:A
101:A C
110:A B
111:A B C
```

The first line shows that the binary number is "**000**". Note that the first bit '0' is for the element 'A', the second bit '0' for the element B, and the last bit '0' for the element C. Since all three bits are '0', the binary number "000" means the empty set.

The next line indicates that the binary number is "**001**" and it corresponds to the subset {C} because only the last bit is '1'. Similarly, the last line indicates that the binary number is "**111**" and it corresponds to the subset {A, B, C} because all bits are '1'.

Sample Run 1: Assume that the user typed the following input.

```
2
CST238 CST370
```

This is the correct output.

```
00:EMPTY
01:CST370
10:CST238
11:CST238 CST370
```

Sample Run 2: Assume that the user typed the following input.

```
0
```

This is the correct output of your program.

```
0:EMPTY
```

3. Write a C++ (or Java) program called **hw2_3.cpp (or hw2_3.java)** that collects the maximum number of apples in boxes. For the problem, you should assume that there are *n* boxes (= from the box number *0* to box *n-1*) on a table and they contain a few apples. Your program should determine the maximum number of apples you can collect from the boxes. **One constraint** in this problem is that **if you collect the apples in a box, you should skip at least one box next to the box**. For example, let's assume that there are four boxes on the table like below:

- Box 0: 3 apples
- Box 1: 2 apples
- Box 2: 3 apples
- Box 3: 5 apples

To collect the maximum number of apples from the boxes, you have to choose the box 0 and box 3 which contain total 8 apples. Since you choose the box 0, you have to skip the next box (= box 1). And also, if you choose the box 2, you can collect total 6 apples. But by skipping this box (= box 2) as well, you can get the max 8 apples.

This is another example. This time, assume that there are 6 boxes on the table like below:

- Box 0: 5 apples
- Box 1: 1 apples
- Box 2: 2 apples
- Box 3: 10 apples
- Box 4: 6 apples
- Box 5: 2 apples

For the sequence, you can collect maximum 17 apples by selecting the boxes 0, 3, and 5.

Input format: This is a sample input from a user.

```
4
3
2
3
5
```

The first line (= 4 in the example) indicates that there are four boxes on the table. And following four lines are the number of apples in each box. For the homework, you can assume that there can be maximum 15 boxes on the table.

Sample Run 0: Assume that the user typed the following lines

```
4
3
2
3
5
```

This is the correct output.

```
Boxes:0 3
Max Apples:8
```

Sample Run 1: Assume that the user typed the following lines

```
6
5
1
2
10
6
2
```

This is the correct output.

```
Boxes:0 3 5
Max Apples:17
```

Sample Run 2: Assume that the user typed the following lines

```
2
1
2
```

This is the correct output.

```
Boxes:1
Max Apples:2
```

Hint: You can generate all subsets using the program developed in **hw2_2**. Then, consider only the subsets which meets the constraint.
For instance, let's assume that you solve a problem with four boxes. Then, you can generate all subsets from 0000 to 1111. If you interpret "0" has no apple in the corresponding box and "1" has apples in the corresponding box, you can generate all possible combinations. Among them, you should exclude the combinations which do not meet the constraint. For example, you have to exclude the subset 1100 because the combination means that there are apples in the first and second boxes. Among all valid subsets, the subset which provides the maximum number of apples will be the answer to the question.