CST 370 – Fall (B) 2021
Homework 1
Due: 11/02/2021 (Tuesday) (11:55 PM)

How to turn in: Write **three programs** in **either C++ or Java** and submit them on Canvas before the due.

- You **can submit** your programs **multiple times** before the due. However, the **last submission will be used for grading**.
- You have to **submit three programs together**, especially at your last submission. **If you submit, for example, only one program** at the last submission, **we are able to see only that program when we grade** your homework.
- Due time is 11:55(PM). Since there could be a long delay between your computer and Canvas, you should submit it early.
- When you submit your homework program, don't forget to include "Title", "Abstract", "ID", "Name", and "Date".

1. Write a C++ program (or Java program) called **hw1_1.cpp (**or **hw1_1.java)** that reads input numbers from a user and displays the closest distance between two numbers among all input numbers.

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

```
5
7
-7
20
8
15
```

The first number (= 5 in the example) indicates that there will be five integer numbers in the input. Then, all numbers from the second line (= 7 in the example) to the last line (15 in the example) are actual numbers. Thus, your program should read them and present the closest distance among five numbers 7, -7, 20, 8, and 15. Because the distance between 7 and 8 is 1, your answer should be 1. Note that the distance between two numbers should be **always positive**. For the problem, you can assume that all **input numbers are distinct (= no duplication)**.

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

```
5
7
-7
20
8
15
```

This is the correct output of your program.

```
Min Distance:1
Pair:7 8
```

Note that **the sequence of two numbers for the pair is important**. Your program has to display the small number first. So, the correct answer for the pair is 7 and then 8. If your program displays 8 and then 7, it's not correct. If there is **more than one pair for the minimum distance**, **all pairs** must be **displayed in ascending order**.

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

```
10
-1
10
-15
3
72
14
12
-20
5
30
```

This is the correct output of your program.

```
Min Distance:2
Pair:3 5
Pair:10 12
Pair:12 14
```

For the program, you can assume that the **number of input values** is **less than 500**.

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

```
7
1234
45678
-9090
8901
0
-12345
5678
```

This is the correct output of your program.

```
Min Distance:1234
Pair:0 1234
```

2. Write a C++ program (or Java program) called **hw1_2.cpp (**or **hw1_2.java)** that reads input numbers from a user and displays the number(s) that occurs most frequently among all the input numbers.

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

```
6
7
-7
20
12
7
15
```

The first number (= 6 in the example) indicates that there will be six integer numbers in the input. Then, all numbers from the second line (= 7 in the example) to the last line (15 in the example) are actual input numbers. Thus, your program should read them and display the most frequent number among 7, -7, 20, 12, 7, and 15. Because the number 7 appears twice in the input, your answer should be 7. If you **have more than one number that occurs most often**, you have to **display all of them in descending order**. For the program, you can assume that there will be less than 500 input integer numbers.

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

```
6
7
-7
20
12
7
15
```

This is the correct output of your program.

```
Frequency:2
Number:7
```

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

```
10
-1
20
-15
5
72
20
5
20
5
30
```

This is the correct output of your program. Note that the frequencies of 5 and 20 are 3. So, you should display 20 and then 5.

```
Frequency:3
Number:20 5
```

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

```
3
5
5
5
```

This is the correct output of your program.

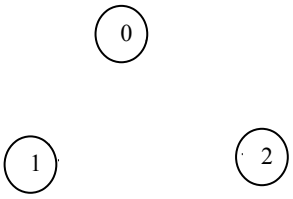
```
Frequency:3
Number:5
```

3. Write a program called **hw1_3.cpp (or hw1_3.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, the sample input 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.

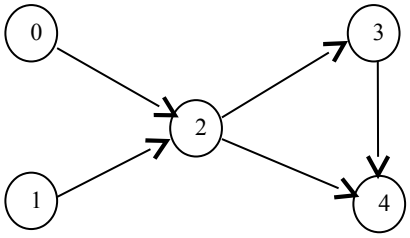
Sample Run 1: Assume that the user typed the following lines

```
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
```

[**Hint**] For this problem, you can **store the input data in an adjacency matrix such as a 2-D array** or a similar container. After that, display the input data in the correct output format. You may not need to store the input data in an adjacency list or a similar format.