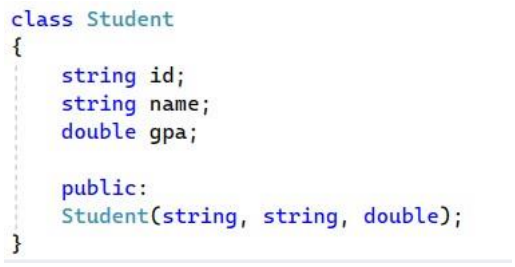
Assignment 1

1. <u>Sorting Algorithms:</u>

Assuming that you have a class, Student:



```
class Student
{
    string id;
    string name;
    double gpa;

    public:
    Student(string, string, double);
}
```

1. Implement the student class with its constructor.

2. Overload the operator < such that it compares the names of two student objects.

3. Read student objects from a file named **students.txt**, which will have the number of students followed by their information as follows:

```
4
Sara Ahmed
78697
3.1 Ali
3541
3.5
Mariam
69712
3.7
Mohamed Kamal
97848
2.2
```

4. Implement Insertion Sort, Selection Sort, Bubble Sort, Shell Sort, Merge Sort, Quick Sort and Count Sort algorithms. temp
   a. Each algorithm should be a separate function implemented using templates to allow sorting of different types of data.

5. Sort the array of students' objects with each of the previous algorithms.
   a. Sort the data one time by Name and another time by GPA.

6. Calculate the running time of each algorithm for each array.

7. The output will be two files, **SortedByGPA.txt** and **SortedByName.txt.** Each file contains:
   a. Algorithm name.
   b. Number of comparisons. with 6
   c. Running Time.
   d. Sorted Student Elements.

```
Algorithm: Insertion Sort
Running Time: 50 milliseconds
Ali
3541
3.5
Mariam
69712
3.7
Mohamed Kamal
97848
2.2
Sara Ahmed
78697
3.1
Algorithm: Selection Sort
Running Time:  45 milliseconds
Ali
3541
3.5
Mariam
69712

3.7
Mohamed Kamal
97848
2.2
Sara Ahmed
78697
3.1


And so one for each algorithm
```

SortedByName.txt

```
Algorithm: Insertion Sort
Running Time: 50 milliseconds
Mariam
69712
3.7
Ali
3541
3.5
Sara Ahmed
78697
3.1
Mohamed Kamal
97848
2.2

Algorithm: Selection Sort
Running Time: 45 milliseconds
Mariam
69712
3.7
Ali
3541
3.5
Sara Ahmed
78697
3.1
Mohamed Kamal
97848
2.2

And so one for each algorithm
```

SortedByGPA.txt

2. Linear Structures:

Implement the following data structures and demonstrates how to use them in the **main function**.

- Implement a **Single Linked List, Double Linked List and Circular Linked List** with the following methods:

as single, and double linked list

    - insertAtHead (elementType element) : void
    - insertAtTail (elementType element) : void
    - insertAt (elementType element, int index) : void
    - removeAtHead () : void
    - removeAtTail () : void
    - removeAt (int index) : void
    - retrieveAt (int index): elementType
    - replaceAt (elementType newElement, int index)
    - isExist (elementType element) : bool
    - isItemAtEqual (elementType element, int index) : bool
    - swap (int firstItemIdx, int secondItemIdx) : void // swap two nodes without swapping data.
    - isEmpty () : bool
    - linkedListSize () : int
    - clear (): void
    - print () : void

- Implement a **Stack** with the following methods:

    - push (elementType element) : void
    - pop () : elementType element //return the first element and remove it.
    - top () : elementType element //return the first element without removing it.
    - isEmpty () : bool
    - stackSize () : int
    - clear (): void
    - print () : void

- Implement a **Queue** with the following methods:

    - enqueue (elementType element) : void
    - dequeue () : elementType element //return the first element and remove it.
    - first () : elementType element //return the first element without removing it.
    - isEmpty () : bool
    - queueSize () : int
    - clear (): void
    - print () : void

Rules:

1- All the code must be in C++.

2- The solution should compile, run without run-time errors, and handle all the cases.

3- Assignment is submitted in teams of 5.

4- You will upload a zipped folder that contains your code (Don’t include any .exe files in your submission).

5- Assignment submission is on Google Classroom (No submission through mail).

6- Follow this convention for naming your folder: ID1_ID2_ID3_A#_G# (i.e 20200111_20200222_20200333_A2_G5_G6)

7- Deadline of the Assignment: 25 April, 2024, at 11:59 p.m.

Any cheating in any part of the assignment is the responsibility of the whole team, and all of the team members will be punished.

Failure to follow any of the above rules will result in your submission being discarded and your team being considered to have not submitted.