**Faculty of computers and Artificial intelligence, Cairo University**

**CS213: Programming II  
Year 2022-2023**

**First Semester**

**Assignment 3 – Version 1.0**

**Course Instructors:**

Dr. Mohammad El-Ramly

**Revision History**

**Version 1.0** By Dr Mohammed El-Ramly 26 Nov. 2022 Main Doc

**Objectives**

This assignment aims to (1) learn OOP concepts in C++, (2) how to use OOP modeling and design with C++ to build systems with intermediate complexity, (3) Templates and STL, (4) Exceptions and (5) Recursion and Backtracking.

**Instructions**

1. **Deadline is Tuesday 20th of Dec. 2022 @ 11:59 pm. Weight is 7 marks** + **3.5 bonus** marks.
2. Students will forms teams of **two** students **from the same lab/section**.

|  |
| --- |
| 1. Please submit **only work that you did yourself**. If you copy work from your friend or book or the net **you will fail the course**. تسليم حلول منقولة من أى مصدر يؤدى إلى الرسوب فى هذا المقرر لا تغش الحل أو تنقله من أى مصدر و اسألنى فى أى شئ لا تفهمه لكن لا تنقل الحلول من النت أو من زملائك أو أى مكان |

**Task 0 (0 marks)**

1. Review STL / Templates / Recursion / Backtracking / Exceptions
2. Create a **private NOT public GitHub** repo for the project and **use it for development.**

**Task 1 (4 marks) - Classes, objects, templates, STL**

**Objectives:**

This is a great development task to conclude the course and prepare you for the next course on data structures. The objective is:

1. Progress development skills to the level library developers who extend the language with new features.
2. Learn how to develop a template library class.
3. Learn how C++ vector works and how it is implemented.

Before working on the class **sit and design on paper how you store the data internally and how you will different algorithms wok on it.** For example how will you increase capacity if needed? How will you implement iterators?

**Task:**

Students should divide the work equally and then **should integrate their code** together and make sure it works properly.

Assume **vector** class template was not invented yet and that we want to develop our own version of it in C++. Already a skeleton version is provided which was developed in lecture. We want to advance this version to a fully working **vector** class with all the important functionalities implemented in it. We will call it **XYVector** where **X** and **Y** are the first letters of team member names. Here is a guide on doing this task <https://www2.cs.sfu.ca/CourseCentral/125/tjd/vector_example.html> **but do not copy any code.**

Our **XYVector** will have the following public interface:

**template ………**

**class XYVector {**

**private:**

**............**

**public:**

**// Constructors and Big 4**

**XYVector (int) // Initialize by specific capacity**

**// No content is added, size = 0**

**// Assign a default size value**

**XYVector (T\*, int n ) // Initialize by n items from array**

**XYVector (const XYVector&) // Initialize with a copy**

**~XYVector() // Delete allocated memory**

**XYVector &operator=(const Vector&); // Copy assignment**

**XYVector &operator=(const Vector&&); // Move assignment**

**// Access operations**

**T& operator[](int) // Access item by reference**

**// Throw an exception if out of range**

**// Modifying operations**

**int push\_back(T) // Add item to end of vec & return # of items**

**// Increase capacity of needed**

**T pop\_back() // Remove and return last element in vec**

**void erase(iterator) // Remove item at iterator**

**// Throw exception if invalid iter**

**void erase(iterator1, iterator2)// Remove items between**

**// iterator 1 <= iterator 2 otherwise do nothing**

**// Throw exception if any iterator outside range**

**void clear() // Delete all vector content**

**void insert(iterator, T) // Insert item at iterator**

**// Throw exception if invalid**

**// Iterators // Supports \*, + and ++ operations at least**

**// Can use: typedef T\* iterator**

**// Or u can use std::iterator so you can**

**// apply STL algorithms on XYVector**

**iterator begin() // Return an iterator (T\*)**

**iterator end() // Return an iterator (T\*)**

**// Comparison operations**

**bool operator==(const XYVector<T>&) // Return true if ==**

**bool operator< (const XYVector<T>&) // Compares item by item**

**// Return true if first different item in this is < in other**

**// Capacity operations**

**int size() const // Return current size of vec**

**int capacity() const // Return size of current allocated array**

**int resize() // Relocate to bigger space**

**bool empty() // Return true if size is 0**

**// Friends**

**friend ostream& operator << (ostream& out, XYvector<T>)**

**};**

It is required to **separate your design into a header file and an implementation file**.

You should write **multiple client programs** to test your class templates on different types.

Team members **should divide the work equally between them.**

**What to deliver?**

1. **Written code in standard C++ not using third-party libraries. Put in a separate directory.**
2. Name your file **A3\_Task1\_YourGroup\_YourIDs.cpp / .h** or **.zip** (if more than one file)
3. In the pdf report, include a detailed program description **and the idea behind each function.**
4. A work breakdown table saying **who did what.**

**Task 2 (3 marks) – Individual Problems from Sheet 3**

**Team is two people.** The **smallest ID** will solve problems 1, 2, 3. The **higher ID** will solve problems 4, 5, 6. These problems will be in **Sheet 3 under Acadox.**

**What to deliver?**

* **Working code in standard C++ not using third-party libraries in a separate directory.**
* Name your file A3\_SheetPb**XX**\_YourID.cpp or .zip (if more than one file) (XX is pb num)
* In the report, write **who did which problems**.

**Group Bonus: Task 3 (1.5 mark) – Doxygen**

Most of developers' time is dedicated to generating ideas and code and understanding. Code documentation, although boring to do, is a very essential task in professional software development. It helps current and future developers understand what the code does.

Doxygen tool supports automatic generation of code documentation from specially written comments. It supports C++, C, Python, etc. **In this task,** you will document Task 1 solution you made with Doxygen and generate documentation.

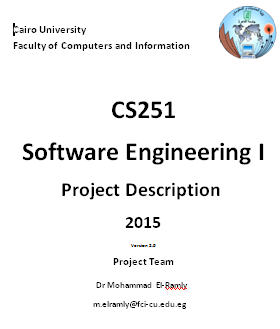
**What to deliver:**

1. A documented version of the code of your vector.
2. The generated documentation.
3. Include a copy of the documentation in the **report.**

**Individual Bonus: Task 4 (2 marks) – Extra Individual Problems from Sheet 3**

**Team is two people.** The **smallest ID** will solve problem 7 and 8. The **higher ID** will solve problems 9 and 10. These problems will be in **Sheet 3 under Acadox.**

**Submission Instructions**



**Team will submit into acadox the following:**

1. **Original non-cheated solutions. Cheated solutions prevent you from learning. Basically, you cheat yourself not me.**
2. A zip file with the following components.
3. A pdf report with the following items.

* The document should have a cover page like this.
* A screen shot for every GitHub account for the shared project
* Work break-down table explaining **who did what**.

1. Team will create a **private** project in **GitHub** to upload code.
2. The source code of each program in a separate folder with suitable name for the folder. For individual problems, each student should put his solutions divided into 2 folders, one for each problem and folder name should have his name and ID.
3. Each team member will work individually on his part. **But the team must provide ONE integrated and working program and report.**
4. Team members are expected to help each other but not do work of others.
5. Team members are responsible of testing all the programs and making sure they work.
6. **All team members must understand the details** of all programs and be able to explain it or even modify it if needed. TA can ask any team member about any of the programs developed**.**

**Marking Criteria**

1. 4 for finished, fully working vector class
2. 3 1 mark for each individual problem that works correctly
3. -8 for both students if any part of the work of any students is cheated (both get –ve)
4. 1.5 for properly generating doxygen documents for task 1
5. 2 for an extra individual problem