# 55-502858: Object Oriented Programming for Computer Science

**Assignment 2:** OO Implementation

**Due:** Thursday 15th April 2021 at 3pm (via Blackboard)

**Marks Available:** 100 (70% of the module assessment)

In this task you will demonstrate your knowledge and ability to identify and apply best-practice object-oriented (OO) features in C++. You will additionally demonstrate your understanding of the standard template library (STL) and basic design patterns. This is an individual piece of work and in-module retrieval is not available for this assessment.

## Introduction

Some of the most enjoyable coding projects you will do are those you have an interest in (e.g. something to do with a hobby, something that might help a family member, etc), so you get to choose your own project for this task. However, your project needs to demonstrate specific areas and aspects as outlined below.

The project must be a C++ Visual Studio 2019 class-based object-oriented application that uses:

* Encapsulation
* Inheritance
* Polymorphism (static and dynamic)

The project must make use of:

* The EasyGraphics library (which you are free to enhance, but will require further reading and self-directed learning)
* Runtime type information operators
* Operator overloading
* The Standard Template Library (STL):
  + Use of explicit iterators
  + Use of a STL algorithm that takes a named function only such that all items in the collection are subjected to the function (e.g. **for\_each**)
  + Use of a STL algorithm that takes a named function and dynamic parameter using **bind** (e.g. **find\_if**)
* Singleton Design Pattern
* Template classes (i.e. your own templates and not the STL)
* Heap memory, balanced off appropriately with stack memory

Most OO applications can be coded to include a sensible demonstration of these requirements so start by thinking of a project that interests you. The project must not be something you have done (or are doing) elsewhere as part of another assignment (including the case study from the first assignment). I am happy to discuss ideas with you.

If you want validation of your project idea, please submit an optional report that outlines how you think your project meets the requirements of the task (1 page maximum) to blackboard (“*Formative feedback for task 2 project”* item under the Assessment section) by **3pm on Monday 15th February 2021** (start of the 5th week of teaching in semester 2). You should outline how you think the project will allow you to address the different aspects of the marking scheme. There are no marks for this report. A standard project brief will be provided for anyone unable to settle on a project by Monday 15th February. I strongly encourage everyone to engage in a self-selecting project process though as you will get much more out of it.

As an example of the scope of this piece of work, please see the drawing tool example provided on Blackboard.

## Code Development

You should write efficient, effective, and thoughtful OO code for this task. Do not overly complicate the project but give it due consideration. Think about the architecture of the code – which class and methods should own which bits of code. There is no single correct solution and you will have to make many decisions about the implementation, but do so logically and above all, think about what you are doing and the implications of your choices – would you be able to sensibly defend your choices if asked why you did it that way as opposed to another?

## Hints

* Remember that objects are data and algorithms together – avoid writing “dumb” classes that are simply a container for data.
* Use object references/pointers where appropriate as opposed to indices
* Use methods/functions to help segment your code into manageable and meaningful modules that can aid reuse and readability
* User interface buttons are simply images drawn to the screen with a hit test that performs a task when the mouse button is pressed inside the graphical area
* Check and fix memory leaks
* There is no single right solution, but some code is better than others… think about what you are doing, what you want to achieve and implement it in a sensible, robust and efficient manner
* Evolve your code and regularly test what you have written; when I put together applications, I didn’t sit down and write it in one go and expect it to run… I build the application in small stages, progressively adding in functionality and testing and debugging as I go (yes, everyone needs to test and debug code they write no matter how long they have been coding for so get practiced at doing so)
* Ensure you are demonstrating good practice OO and programming techniques through the use of appropriate language constructs; it is the quality of your code that is of most importance and your project will allow you to demonstrate your skills and knowledge towards that in an object-oriented setting.

*Continued on the next page…*

## Grading Guidelines

Marks will be awarded as illustrated below:

| **Area / Theme being assessed** | **Percentage Available** |
| --- | --- |
| OO design and choice of classes: use of encapsulation and inheritance | 15% |
| Use of language constructs in the development of the application: best-practice implementation within C++ using appropriate and consistent syntax and constructs, including constructor and destructors, polymorphism and runtime-type information, inlining, use of const, virtual, operator overloading, etc. | 40% |
| Use of the standard template library | 10% |
| Design Patterns | 10% |
| Templates | 10% |
| Memory usage and clean up | 10% |
| Consistent and sensible layout and formatting, structure of code (i.e. use of files) and naming | 5% |

The University common grading descriptor will be used to determine marks for each area (see last page). The mark scheme embeds the concept of extended work by rewarding only the highest marks to those who demonstrate evidence of independent investigation, learning, critical thought, and problem analysis (via good code solutions). Aligned with university policy, your overall grade across the topics outlined above will be mapped to one of the *Common Grade Percentages (CG%)* to give your final mark (see common grading descriptor on the last page and the CG column).

## Main Submission Process

Your assignment should be submitted electronically through the module Blackboard site as a single ZIP file that contains your entire project and source code **(but without the build and intellisence files and folders)**, and a ReadMe.txt file that gives brief instructions about using your application. Your last on-time attempt will be viewed and graded (as per university regulations).

The source code must be in the form of a Visual Studio 2019 project within the compressed ZIP file and contain all files that allow the project to be opened, built and run on a campus computer (this includes any assets you may have used, which must use relative pathing in the code - ***do not use absolute paths as I will not have the same drive structure as you outside the project folder***). Make sure that you upload the correct files by checking once you have submitted - mistakes discovered after the deadline cannot be corrected; it is your responsibility to ensure that you submit the correct files by the deadline. You may be asked to provide a walkthrough of your code during which you will need to discuss all aspects of the work you submitted before a grade is awarded.

**Remember to include all source code and check your submission once uploaded.**

The submission deadline is given at the top of this document.

**Learning Outcomes**

This task assesses the following learning outcomes from the module descriptor

* Identify and apply object-oriented features of a modern programming language to implement OO designs.
* Use OO techniques to encapsulate functionality associated with classic search algorithms and data-structures and relate those techniques to facilities provided in standard libraries and patterns.

# Level 5 - Generic grade descriptor: relationship of degree classification to percentage mark ranges and categorical grades (CG)

| **Class** | **Mark range** | **CG%** | **General Characteristics** |
| --- | --- | --- | --- |
| FIRST  (Excellent) | 93 - 100 | 96 | Exceptional breadth and depth of knowledge and understanding of the area of study; evidence of extensive and appropriate selection and critical evaluation/synthesis/analysis and of reading/research beyond the prescribed range, in both breadth and depth, to advance work/direct arguments; exceptional demonstration of relevant skills; excellent communication; performance deemed to be beyond expectation. |
| 85 - 92 | 89 |
| 78 - 84 | 81 | Outstanding/excellent knowledge and understanding of the area of study **as the student is typically able to go beyond what has been taught (particularly for a mid/high 1st)**; evidence of extensive and appropriate selection and critical evaluation/synthesis/ analysis of reading/research **beyond the prescribed range**, to advance work/direct arguments; excellent demonstration of relevant skills; excellent communication; performance deemed beyond expectation of the level. |
| 70 - 77 | 74 |
| UPPER  SECOND  (Very good) | 67 - 69 | 68 | Very good knowledge and understanding of the area of study as the student **is typically able to relate facts/concepts together with some ability to apply to known/taught contexts**; evidence of appropriate selection and evaluation of reading/research, some beyond the prescribed range, may rely on set sources to advance work/direct arguments; demonstrates autonomy in approach to learning; very good demonstration of relevant skills; strong communication skills. |
| 64 -66 | 65 |
| 60 - 63 | 62 |
| LOWER  SECOND  (Good) | 57 - 59 | 58 | Good knowledge and understanding of the area of study **balanced towards the descriptive rather than analytical**; evidence of appropriate selection and evaluation of reading/research but generally reliant on set sources to advance work/direct arguments; good demonstration of relevant skills, though may be limited in range; communication shows clarity but structure may not always be coherent. |
| 54 - 56 | 55 |
| 50 - 53 | 52 |
| THIRD  (Sufficient) | 47 - 49 | 48 | **Knowledge and understanding is sufficient to deal with terminology, basic facts and concepts** but fails to make meaningful synthesis; some ability to select and evaluate reading/research however work may be more generally descriptive; strong reliance on available support set sources to advance work; arguments may be weak or poorly constructed; adequate demonstration of relevant skills over a limited range; communication/presentation is generally competent but with some weaknesses. |
| 44 - 46 | 45 |
| 40 - 43 | 42 |
| FAIL  (Insufficient) | 30 - 39 | 35 | Insufficient knowledge and understanding of the area of study; some ability to select and evaluate reading/research however work is more generally descriptive; fails to address some aspects of the brief; a limited use of sources to advance work; arguments may be weak/poor or weakly/poorly constructed; demonstration of relevant skills over a reduced range; communication shows limited clarity, poor presentation, structure may not be coherent. |
| 20 - 29 | 25 |
| 10 - 19 | 15 | Highly insufficient knowledge or understanding of the area of study; **understanding is typically at the word level with facts being reproduced in a disjointed or decontextualised manner**; fails to address the outcomes addressed by the brief; typically ignores important sources in development of work and data/evidence inappropriately used; weak technical and practical competence hampers ability to demonstrate/communicate achievement of outcomes. |
| 1-9 | 5 |
| ZERO | 0 | 0 | Work of no merit OR absent, work not submitted, penalty in some misconduct cases. |