Manual C++

Term 2.1

Course Manual study year 2019/2020

Bachelor Creative Media and Game Technologies (CMGT)
School of Creative Technology

Publication date	30 August 2019
Version	1
Module coordinator	Yvens R Serpa (YRE03),
Lecturers	Yvens R Serpa (YRE03), Ron Talman (RTA04)
CMGT roles	Engineer

1 General overview

Module Name	C++
Unit code	L.25877
Year and Term	2.1
CMGT roles	Engineer
Credits	3 ECTS
Lessons	6 * 6 = 36 hours
Study load	52 hours
Responsible lecturer	Yvens R Serpa (y.reboucasserpa@saxion.nl)
Lesson structure	1.5 hours lecture, 2.5 hours labs
Module summary	The student is able to understand and apply the basic principles of C++, including compiling programs, using external libraries and managing memory.
Industry relevance	C++ is a powerful programming language used over a wide span of applications, from operational systems to real-time interactive applications. It is vastly used by the industry when performance is crucial, and resources should be efficiently handled. You should have knowledge in order to understand the more low-level details of engineering software related to the field of Creative Media & Game Technologies.
Type of exam	Assignment
Exam code	T.51689
	3
CMGT Competencies	1. Technical research and analysis 2. Designing, prototyping and realizing 3. Testing and rolling out 12. Responsibility
Required prior knowledge and skills / conditions for enrolment	The student needs to understand basic programming concepts, such as variables and control structures. The student must have experience working with an OOP programming language and understand the main OOP concepts (Inheritance, Encapsulation, and Polymorphism). The student must be used to work with a medium-size code base project.
Preparatory for:	The acquired knowledge will be further used in the following quarter in Open GL module and also on the project 3rd Project.

2 Why this module?

C++ is a powerful programming language used over a wide span of applications, from operational systems to real-time interactive applications. It was developed in the late '70s enhancing the C language features, by fully supporting Objected-Oriented Programming, or, as stated simply: by allowing the use of classes. According to the annual report by Stack Overflow[1], C++ is still the 10th most popular programming language for both the general public and professionals.

In order to create and develop interactive applications, the student must have knowledge of different levels of programming abstraction, including a lower-level approach that allows for more control of the resources available (such as computer memory) leading to a more efficient and high-performance application. C++ allows that approach and it's used by the industry to create breakthrough and innovative applications, especially in the field of Games and Interactive Media

Additionally, C++ has matured over the decades, adding features and becoming a multiparadigm language, leveling up to stand among the newest programming languages. While this course won't be able to grasp its full potential but will lead the student in a comfortable position to study on his/her own and learn its darkest and deepest secrets.

2.1 What happens in the labs and lectures?

During the lectures, we'll discuss the inner workings of C++, how the language is structured and what design decisions emerge from it. We'll give special attention to common faults and ways to overcome them. Every week, the students will have a lecture on C++ subject according to the Organization section. After the lecture, there's a lab in which the student has the opportunity to work on the module assignment and the specific lab assignments.

Every lecture and lab have its PDF Slides, which will be made available through Blackboard soon after the week lecture is done. Lab slides won't necessarily be displayed by the lab teacher. They contain additional content and exercises. The latter is not mandatory, but highly valuable to improve and test the student's skill.

After Week 3 lecture, there's going to be an online multiple-choice exam, via Blackboard, to test student's knowledge so far. It is not mandatory, but it is a good measure to test if the student can start with the Module Assignment right away, or if he/she should dedicate more time studying the basic content of the module. Also, the final grade for the exam doesn't influence the module's final grade and it can be done multiple times. The exam's objective is to expose the student into real C++ problems, questions, and scenarios.

2.2 How does this module relate to other modules in the CMGT study programme?

It is expected that the student has followed the Game Programming and Algorithms modules, being able to work with basic programming concepts (variables and control structures) and work the basic OOP principles (Inheritance, Encapsulation, and Polymorphism). Also, given the experience from previous projects of working with medium-size code bases (multiple code files and assets).

This module prepares for Project Lift Off. In the project Designers and Engineers have to work together to build an Arcade game controller.

[1] https://insights.stackoverflow.com/survey/2018#technology

This module will prepare the student using C++ and 3rd party libraries in the Open GL module and also on the project 3rd Project.

Besides that, C++ and the concepts taught during this module should be applied in future projects and work in general.

3 What are you going to learn in this module (learning objectives)?

The student:

1. understands the basic C++ syntax (including CONST modifier).
2. understands the concepts of compiling, linking and executing a program.
3. uses pointers and references to handle memory efficiently.
4. employs the C++ standard libraries to handle file persistency.
5. employs the C++ standard libraries to handle simple data structures.
6. uses a Graphical API (SFML) to develop a 2D game in C++.

4 Which resources do you need?

The student must have a C++ IDE on its computer (suggested Visual Studio / Visual Studio Code for Windows users and Visual Studio Code / CLion for Mac users).

For Windows users, the student should download and install the required C++ modules before the lectures, since it may take a considerable amount of time. It is also recommended to install Git Bash as a command line to facilitate the usage of 3rd party libraries and package managers. Additionally, the student should install a Windows package manager, such as Chocolatey.

For Mac users, the student should download and install the required C++ modules (usually in the form of installing XCode), since it may take a considerable amount of time. It is also recommended to install iTerm as a command line to facilitate the usage of 3rd party libraries and package managers. Additionally, the student should install a Mac packager manager, such as Brew.

5 What does the programme of this module look like?

On Blackboard you'll find the course content and a detailed course overview.

week	Lecture/Lab	Topic(s)
2.4	Lecture	Introduction to C++ and Pointers: C++ basic syntax, pointers, basic memory management, and input.
2.4	Lab	Lab Exercises.
2.5	Lecture	Introduction to SFML, and the concepts of scope, references and operator overload.
2.5	Lab	Lab Exercises.
2.6	Lecture	CONST-ness, STL, and Applications: working with OOP in C++ using SFML [Part I].
2.6	Lab	Lab Exercises.
2.7	Lecture	CONST-ness, STL, and Applications: working with OOP in C++ using SFML [Part II].
2.7	Lab	Lab Exercises.
2.8	Lecture	Advanced topics in C++ (smart pointers, lambda operations, and advanced conversion operations), and a quick look at performance and memory profilers.
2.8	Lab	Lab Exercises.

6 How is this module assessed?

6.1 Assessment

To pass the module, the student must submit and present the module assignment, achieve at least sufficient in all Rubric's criteria and attend all its requirements. One insufficient criterion is enough to fail the student, even if the overall grade is above 5.5. The module assignment is described in full detail in the Assignment Document provided in Blackboard. The deadline for the assignment submission will be 12 hours before the presentation date. The assignment cannot be submitted afterward.

The Assignment consists of developing a simple 2D Battle Game in the style of Pokémon and Final Fantasy using C++ and SFML. The game will consist of simple menu navigation and a fight scene. No animations are necessary, and the game can be mostly text-based with graphics to portray the characters and menus. The student can use any art asset (including sound and soundtrack) from free sources and even make them him/herself, but those will not be graded or influence by any means the final grading.

The assignment must be done individually, and it is expected that all students have ownership of their code. It is allowed to use the code excerpts from the examples in the lecture slides and lab slides. It is not allowed to use copied code from online/offline tutorials, but it is recommended that the student use them to learn and study. It is also not allowed to have "group" assignments or that the students do a general assignment together and present separately. If the assessing teacher found that out, both students will be granted insufficient[2].

The presentations will be done in Week 9 or 10 of the current quarter (the specific date will be informed through Blackboard as soon as it is scheduled). The scheduling for presentations will be done via e-mail and a shared Excel file with timeslots. During the presentation, the student will also be questioned about the module's content and assignment. Answering questions incorrectly can demonstrate that the student doesn't have sufficient knowledge of certain criteria and may result in an insufficient grade. The questions are also a way to validate the ownership of the presented work. In the case of a non-submission, the timeslot reserved will be freed.

Additionally, after Week 3 lecture, there's going to be an online multiple-choice exam, via Blackboard, to test student's knowledge so far. It is **not mandatory**, but it is a good measure to test if the student can start with the Module Assignment right away, or if he/she should dedicate more time studying the basic content of the module. The exam doesn't influence the final grade and it can be done multiple times. The exam's objective is to expose the student into real C++ problems, questions, and scenarios.

6.2 Procedure

The student must submit the assignment via Blackboard with all files listed below in a .zip file with the following name format: StudentName-StudentNumber-Year.zip

- Source Code;

[2] It doesn't mean that students cannot work together to solve problems. However, every student must present their unique source code and assignment. Additionally, the questioning during the presentation will make sure that the student understands the code being presented and owns it.

- 3rd party libraries used (including the mandatory ones, such as SFML);
- Document file (.txt file, for example) with a list of 3rd party libraries used and their specific versions (including the mandatory ones, such as SFML);
- An executable file (runnable on Windows **OR** MacOS **OR** Linux);
    - Name must be StudentName-StudentNumber-Year.exe.
- Example of IO Files used in-game;
- Resources used (images, audio files, data files, etc.);

If the student fails to submit one of the listed files, he/she will automatically fail the assessment by not meeting the requirements.

6.3 Criteria & Assessment form

The student must achieve at least sufficient for all criteria in the rubrics.

6.4 Resit

The second opportunity will be at the end of the following term (in week 2.9 or 2.10).

7 Who are the contact persons for this module?

Module coordinator:
Yvens Rebouças Serpa y.reboucasserpa@saxion.nl

Lecturers:
Yvens Rebouças Serpa y.reboucasserpa@saxion.nl
Ron Talman r.g.talman@saxion.nl

8 Rubric

Note: the rubrics are used to determine your grade and are visible in Blackboard under 'Grades and Feedback' → 'View rubrics'.

Rubric Input - Output				
Preconditions: Assignment via Blackboard with all files listed below in a .zip file with the following name format: StudentName-StudentNumber-Year.zip • Source Code; • 3rd party libraries used (including the mandatory ones, such as SFML); • Document file (.txt file, for example) with a list of 3rd party libraries used and their specific versions (including the mandatory ones, such as SFML); • Executable file (runnable on Windows **OR** MacOS **OR** Linux); ○ Name must be StudentName-StudentNumber-Year.exe. • Example of IO Files used in-game; • Resources used (images, audio files, data files, etc.);				
	Insufficient	**Sufficient**	**Good**	**Excellent**
Variables and Structures (25%)	0% Doesn't match the criteria mentioned sufficient.	15% You understand the usage of variables and data structures in C++, including concepts as **typedef** and **const** modifier. Correct usage of classes and inheritance.	20% See sufficient+: You use **iterators** while working with STL structures. Consistent use of the **const** modifier when applicable.	25% See good+: You use **templates** in **at least one** student-made class.
Compiling and Executing Program (20%)	0% Doesn't match the criteria mentioned under sufficient.	10% The assignment was compiled correctly into an executable file that runs without the aid of an IDE.	15% See sufficient+: The executable file correctly linked with 3rd party libraries (mandatory ones don't count).	20% See good+: The executable file can be executed with command-line arguments to change the program's behavior (for instance, to change the screen resolution).

Memory Management (25%)	0% Doesn't match the criteria mentioned under sufficient.	15% You understand and make basic use of **pointers** and **references**.	20% See sufficient+: The assignment has no memory leak (proved by using a memory profiler program).	25% See good+: You understand and use **smart pointers** in the assignment (or adequately can explain why they were not needed).
Assignment – Main Features (30%)	0% Doesn't match the criteria mentioned under sufficient.	20% The assignment contains all mandatory features, including IO file operations. It was built using SFML.	25% See sufficient+: The assignment contains **at least 3** additional features.	30% See good+: The assignment contains **all** additional features **AND/OR at least 2** challenge features.