Course Syllabus

University of Macau
Faculty of Science and Technology
Department of Computer and Information Science
CISB354 Programming Language
2nd Semester 2014/2015
Part A – Course Outline

Elective course in Computer Science

Course description:
This course provides in-depth coverage of object-oriented programming principles and techniques using C++. Topics include classes and objects, vectors, overloading, inheritance, polymorphism, templates, etc.

Course type:
Theoretical with substantial laboratory/practice content

Prerequisites:
- SFTW120

Textbook(s) and other required material:
- Paul Deitel and Harvey Deitel. (2009) *C++ How to Program***.** Prentice Hall, US.

References:
- Stephen Prata. (2004). *C++ Primer Plus*. 5th ed., Sams.

Major prerequisites by topic:
- Programming languages and algorithms

Course objectives:
- Learn the fundamentals, structure, logic, and syntax of object-oriented programming (OOP) in C++
- Design and implement programs using C++
- Run and analyze a given program; identify ways in which it fails
- Apply basic tools to aid in developing programs

Topics covered:
- **Introduction (5 hours)**: These lectures will give an overview of different types of computer languages: machine languages, assembly languages, and high-level languages. We will introduce the object technology, the C++ language, and the typical C++ development environment. Students will also have a hands-on experience on test-driving a C++ application.
- **Basic concepts of classes and objects (5 hours)**: We will learn how to define a class and use it to create an object, to define member functions to implement the class's behaviors, to declare data members to implement the class's attributes, and initialize a class by constructor.
- **Arrays, vectors, and pointers (8 hours)**: We will review the important data structure of the C language: arrays and pointers. This includes the topics of declaring, initializing, referencing arrays and pointers. We will also discuss the relationship and usage of them with functions. The class template `vector` in the C++ standard library will be introduced.
- **Classes (10 hours):** These lectures cover a deeper understanding on C++ classes. We will learn to preprocessor wrapper, class scope, `public` and `private` functions and data, `friend` functions and classes. The concept of container classes will also be introduced. Finally, we will learn to use proxy classes to hide implementation details from a class's clients.
- **Operator overloading (8 hours)**: We will learn how operator overloading can help to craft classes. This includes the use of unary and binary operators, converting objects, and the `explicit` keyword.

- **Object-oriented programming techniques (10 hours)**: These lectures cover the important OOP techniques - inheritance and polymorphism. We will learn the notion of base classes and derived classes, the use of `protected` member access specifier, the use of constructors and destructors, the differences between `public`, `protected`, and `private` inheritance, the distinction between abstract and concrete classes, the use of `virtual` functions and dynamic binding.
- **Templates (8 hours)**: Here, we will study the use of templates to create a group of related (overloaded) functions. We will learn the differences between class templates and class-template specializations, function templates and function-template specializations, and to overload function templates.

Class/laboratory schedule:

Timetabled work in hours per week			No of teaching weeks	Total hours	Total credits	No/Duration of exam papers
Lecture	Tutorial	Practice				
2	0	2	14	56	3	1 / 3 hours

Student study effort required:

Class contact:	
Lecture	28 hours
Hands-on practice	28 hours
Other study effort	
Self-study	10 hours
Homework assignment	16 hours
Project / Case study	30 hours
Total student study effort	112 hours

Student assessment:
Final assessment will be determined on the basis of:

Homework	20%	Project	30%
Mid-term	20%	Final exam	30%

Course assessment:
The assessment of course objectives will be determined on the basis of:

- Homework, project and exams
- Course evaluation

Course outline:

Weeks	Topic	Course work
1	**Introduction** Different types of programming languages, basic object-technology concepts, a typical C++ development environment, test-drive a C++ application	
2	**Introduction to classes and objects** Classes, objects, member functions, data members, constructors, and destructors	Lab 1
3-5	**A deeper look on Classes** Preprocessor wrapper, class scope, `public` and `private` functions and data, `friend` functions and classes, container classes, proxy classes	Lab 2, Lab 3, Lab 5, Mid-term
6-7	**Arrays, vectors, and pointers** Declaration, initialization of arrays and pointers, the relationship between arrays/pointers and functions, class template `vector`	Homework 1 and Lab 4
8	**Operator overloading** Unary and binary operators, object conversion, the `explicit`	Lab 6

Weeks	Topic	Course work
9-12	**OOP techniques - Inheritance and Polymorphism** Base classes and derived classes, `protected` member access specifier, constructors and destructors in inheritance hierarchies, `public`, `protected`, and `private` inheritance, abstract and concrete classes, `virtual` functions and dynamic binding	Lab 7
13	**UML** UML diagrams, object-oriented design	Project
14	**Review for exam**	

Contribution of course to meet the professional component:
This course prepares students with fundamental knowledge and experiences to object-oriented programming.

Relationship to CS program objectives and outcomes:
This course primarily contributes to the Computer Science program outcomes that develop student abilities to:

(a) An ability to apply knowledge of computing and mathematics to solve complex computing problems in computer science discipline.

(b) An ability to apply knowledge of a computing specialisation, and domain knowledge appropriate for the computing specialisation to the abstraction and conceptualisation of computing models

(c) An ability to analyse a problem, and identify and define the computing requirements appropriate to its solution

(d) An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs with appropriate consideration for public health and safety, social and environmental considerations

Relationship to CS program criteria:

Criterion	DS	PF	AL	AR	OS	NC	PL	HC	GV	IS	IM	SP	SE	CN
Scale: 1 (highest) to 4 (lowest)		4	2		2		4						1	

Discrete Structures (DS), Programming Fundamentals (PF), Algorithms and Complexity (AL), Architecture and Organization (AR), Operating Systems (OS), Net-Centric Computing (NC), Programming Languages (PL), Human-Computer Interaction (HC), Graphics and Visual Computing (GV), Intelligent Systems (IS), Information Management (IM), Social and Professional Issues (SP), Software Engineering (SE), Computational Science (CN).

Course content distribution:

Percentage content for			
Mathematics	Science and engineering subjects	Complementary electives	**Total**
0%	100%	0%	100%

Coordinator:
Prof. Zhiguo Gong

Persons who prepared this description:
Dr. Jiantao Zhou

Part B – General Course Information and Policies

2nd Semester 2014/2015

Instructor: Dr. Jiantao Zhou
Office: E11-4089
Office hour: 14:30 – 15:30 Every Thursday
Phone: 88224495
Email: jtzhou@umac.mo

Time/Venue: *09:30-10:45, Friday/E12-G021*

Grading distribution:

Percentage Grade	Final Grade	Percentage Grade	Final Grade
100 - 93	A	92 - 88	A–
87 - 83	B+	82 - 78	B
77 - 73	B–	72 - 68	C+
67 - 63	C	62 - 58	C–
57 - 53	D+	52 - 50	D
below 50	F		

Comment:

The objectives of the lectures are to explain and to supplement the text material. Students are responsible for the assigned material whether or not it is covered in the lecture. Students who wish to succeed in this course should read the textbook prior to the lecture and should work all homework and project assignments. You are encouraged to look at other sources (other texts, etc.) to complement the lectures and text.

Homework policy:

The completion and correction of homework is a powerful learning experience; therefore:

- Homework is due 10 days after assignment unless otherwise noted.
- The course grade will be based on the average of the homework grades.

Course project:

The project is probably the most exciting part of this course and provides students with meaningful experience to design and implement a medium size system applying all the OOP techniques learnt throughout this course:

- You will work in a group of four students for the course project.
- The requirements will be announced and discussed in class.
- The project includes design, implementation, demonstration, and report writing.

Exam:

One 2-hour mid-term exam will be held during the semester. The final examination will be a 3-hour exam.

Note:

- Check UMMoodle (https://ummoodle.umac.mo/) for announcement, homework and lectures. Report any mistake on your grades within one week after posting.
- No make-up exam is given except for CLEAR medical proof.
- Cheating is absolutely prohibited by the university.