University of Macau
Faculty of Science and Technology
Department of Computer and Information Science
CISB110 – Programming Science
Syllabus
1st Semester 2014/2015
Part A – Course Outline

Compulsory course in Computer Science

Course description:
(2-2) 3 credits. This course introduces the notion of algorithms and teaches principles of problem solving. It also introduces fundamental concepts of programming, such as scalar data types, variables, functions, choice, iteration, recursion, arrays, strings, enumerations, record types, and file processing. These concepts are introduced through the C programming language.

Course type:
Theoretical with substantial laboratory/practice content

Prerequisites:
none

Textbook(s) and other required material:
- Jeri R. Hanly and Elliot B. Koffman, *Problem Solving and Program Design in C*, 7th ed., 2012, Addison-Wesley. (Required)

References:
- C Programming Notes: http://www.eskimo.com/~scs/cclass/notes/top.html
  Introductory C programming notes, designed to be used in conjunction with Kernighan and Ritchie's C book.
- C Programming: http://www.strath.ac.uk/CC/Courses/NewCcourse/ccourse.html
  A C programming course for people with previous programming experience.

Major prerequisites by topic:
none

Course objectives:
1. Introduce students to fundamental principles of problem solving. [c]
2. Introduce students to fundamental concepts related to programming. [a, b]
3. Introduce students to principles and use of the C programming language. [a, j]
4. Foster students' ability to translate informal requirements into correct and efficient programs. [c, d]
5. Learning to apply course material to improve analytical thinking and problem solving. [a, c]

Topics covered:
1. **Problem solving and algorithms (3 hours):** Study the method of problem solving and algorithm development. Introduce the software development method, from problem specification, through analysis, design and implementation to testing and maintenance. Present the use of program outlines, and program development by adaptation. Introduce guidelines for good problem style.
2. **Overview of the C programming language (4 hours)**: Present a brief overview of the C language, and introduce the basics of C program development. Study a simple C program in detail to understand the purpose of the different parts of a program. Introduce arithmetic in C, same-type and mixed-type expressions, and rules for evaluation of arithmetic expressions.
3. **Data types, variables and operators (2 hours):** Introduce the purpose and function of data types. Study the primitive data types in C, including their typical range and storage requirements. Study numerical inaccuracies due to data type limitations. Show how data can be converted between types automatically

and through type casting. Discuss the use of computer memory through variables. Study the operators of the C language.

4. **Modular program design and functions (4 hours):** Introduce top-down program design, and program representation using structure charts. Study the use of input and output parameters to communicate with functions, as well as return values. Present the concept of transfer of control between functions in a program. Introduce the use of function prototypes.
5. **Selection and iteration (8 hours):** Study principles of selection structures, including conditional expressions and the if and switch statements in C. Introduce short-circuit evaluation, nested conditional statements, and multiple-alternative decisions. Study principles of iteration structures, including different kinds of standard loops using the for, while and do-while statements in C. Introduce loop-control variables, upward/downward counting loops and stepping, and nested loops.
6. **Arrays, strings, pointers, and user-defined types (12 hours):** Introduce the basic concept of data structures, including arrays and strings, as well as simple and complex user-defined types, namely enumerated types and structure types. Study typical operations on these kinds of data structures. Introduce the use of pointers as function parameters, and in arrays and strings.
7. **Terminal and file input/output (3 hours):** Introduce terminal input and output (I/O) and formatted I/O using the scanf and printf functions. Introduce the concept of I/O streams and study the application of I/O primitives to files, including text files and binary files.
8. **Recursion (2 hours):** Introduce the basic principle of recursion, and how to trace the execution of a recursive program. Study the design of recursive functions, including terminating recursion and returning result values. Show examples of recursion on numbers and strings.

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	1	1	14	56	3	1 / 2 hours

Student study effort required:

Class contact:	
Lecture	26 hours
Quiz	2 hours
Mid-term exam	2 hours
Tutorial	12 hours
Practice	14 hours
Other study effort:	
Self-study	28 hours
Programming practices	16 hours
Total student study effort	100 hours

Student assessment:

Final assessment will be determined on the basis of:

Programming practices	20%
Quizzes	20%
Mid-term Exam	20%
Final Exam	40%

Course assessment:

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

1. Programming practices and exams
2. Course evaluation

Course outline:

Weeks	Topic	Course work
1-2	**Introduction and Overview** Basic concepts of computer operation and memory, algorithms, problem solving and software development method.	
2-3	**Introduction to C** History of C. C program development. Detailed study of a simple C program.	
3	**Arithmetic** Arithmetic operators, same-type and mixed-type expressions, mixed-type assignments, rules for evaluation of complex arithmetic expressions, number output.	
4	**Program Design and Functions** Design documentation by program outlines, program development by adaptation, code re-use and program libraries, structure charts, function definitions, function input, output and return values, transfer of control, function prototypes.	
5-6	**Selection Structures** C if and switch selection structures, simple and complex conditions, comparison and logical operators, short-circuit evaluation, nested conditional statements, multiple-alternative decisions, conditional operator.	Practice 1
6-7	**Iteration Structures** C while, for, do-while repetition structures, counting loops, sentinel-controlled loops, endfile-controlled loops, input validation loops, nested loops, flag-controlled loops, loop control variables, compound assignment operators, increment/decrement operators.	Practice 2
8	**Advanced Use of Functions** Simple functions vs. functions with output parameters, pointers, scope of names, output parameters as function arguments.	Practice 3
9	**Simple Data Types** Comparison of different data types, numerical inaccuracies in programs, automatic type conversion, type casting, enumerated types.	Quiz 1
9-11	**Arrays** Arrays as data structures, array subscripts, parallel arrays, array initialization, array elements and arrays as function parameters, partially filled arrays, linear search on an array, multi-dimensional arrays.	Mid-term exam Practice 4
11-13	**Strings** String variables, string termination character, string input/output, string library functions, string assignment, sub-strings, joining strings, string length, string comparison, character operations, command line parameters.	Practice 5
13	**Recursion** Concept of recursion, analysis of recursive problems, tracing recursion, terminating recursion, recursion on numbers and strings, recursion vs. iteration.	Practice 6
14	**Structure Types** Records, defining and using structure types, hierarchical structures, structures as input/output parameters, indirect component selection operator.	Practice 7
14	**File Processing** Files and streams, standard streams, file pointers, text and binary files, file operations.	Quiz 2 Practice 8

Contribution of course to meet the professional component:

This course prepares students to work professionally in the area of software development.

Relationship to CS program objectives and outcomes:

This course primarily contributes to the Computer Science program outcomes that develop these student abilities:
(a) An ability to apply knowledge of computing and mathematics appropriate to the programme outcomes and to the 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.
(j) An ability to use current techniques, skills, and tools necessary for computing practice with an understanding of the limitations.

Relationship to CS programme outcomes:

	Programme Outcomes									
	(a)	**(b)**	**(c)**	**(d)**	**(e)**	**(f)**	**(g)**	**(h)**	**(i)**	**(j)**
CISB 110 Programming Science	TP	TP	TP	TP						TP

T – Teach, P – Practice, M – Measured

Relationship to CS programme criteria:

Criterion	DS	PF	AL	AR	OS	NC	PL	HC	GV	IS	IM	SP	SE	CN
Scale: 1 (highest) to 4 (lowest)		1					3						4	

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. Xiaoshan Li

Persons who prepared this description:

Dr. Robert P. Biuk-Aghai
Dr. Fai Wong

Part B General Course Information and Policies

1st semester 2014/2015

Instructor:	Dr. Robert P. Biuk-Aghai	Office: E11-4006
Office Hour:	Tue 10-11am, 4-5 pm, Thu 10-11 am, or by appointment	Phone: 8822 4375
Email:	robertb@umac.mo	

Time/Venue: Wed 8:30 am – 10:30 am (lecture)
Thu 11:30 am – 12:30 pm (Class A), Fri 8:30 am – 9:30 am (Class B) (tutorial)
Mon 11:30 am – 12:30 pm (Class A), Tue 8:30 am – 9:30 am (Class B) (laboratory)

Grading Distribution:

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

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 lectures. Students who wish to succeed in this course should read the lecture notes prior to the lecture and should do all programming practices and lab exercises. You are encouraged to look at other sources (other texts, etc.) to complement the lectures and primary text.

Programming Practice Policy:
The completion and correction of programming practices is a powerful learning experience; therefore:

- There will be 8 graded programming practices.
- Programming practices are due one week after assignment unless otherwise noted, and late submissions will lose points (1% point off per hour late).
- Possible revision of grades may be discussed with the grader within one week from the return of the marked programming practice.
- The course grade will be based on the average of all programming practice grades.

Quizzes:
There will be two quizzes, after about 1/3 and at the end of the semester.

Mid-term Exam:
One mid-term exam will be held at about the middle of the semester.

Note

- The lecture session is an important part of this course and attendance is compulsory. At most 20% absence without leave is allowed.
- Check UMMoodle (ummoodle.umac.mo) for announcements, programming practice assignments and lecture notes. Report any mistakes on your grades within one week after posting.
- No make-up exam is given except for CLEAR medical proof.
- No exam is given if you are 30 minutes late in the midterm exam, or 45 minutes late in the final exam. Even if you are late in the exam, you must turn in at the due time.
- Cheating is strictly prohibited by the university and will be severely punished.

Student Disabilities Support Service:
The University of Macau is committed to providing an equal opportunity in education to persons with disabilities. If you are a student with a physical, visual, hearing, speech, learning or psychological

impairment(s) which substantially limit your learning and/or activities of daily living, you are encouraged to communicate with your instructors about your impairment(s) and the accommodations you need in your studies. You are also encouraged to contact the Student Disability Support Service of the Student Counselling and Development Section (SCD) in Student Affairs Office, which provides appropriate resources and accommodations to allow each student with a disability to have an equal opportunity in education, university life activities and services at the University of Macau. To learn more about the service, please contact SCD at scd.disability@umac.mo, or 8822 4901 or visit the following website: