CS2310: COMPUTER PROGRAMMING

Effective Term
Semester A 2023/24

Part I Course Overview

Course Title
Computer Programming

Subject Code
CS - Computer Science

Course Number
2310

Academic Unit
Computer Science (CS)

College/School
College of Engineering (EG)

Course Duration
One Semester

Credit Units
3

Level
B1, B2, B3, B4 - Bachelor's Degree

Medium of Instruction
English

Medium of Assessment
English

Prerequisites
Nil

Precursors
CS1102 Introduction to Computer Studies or CS1302 Introduction to Computer Programming or equivalent

Equivalent Courses
CS2311 Computer Programming
CS2315 Computer Programming

Exclusive Courses
CS1315 Introduction to Computer Programming
CS2313 Computer Programming
CS2360 Java Programming
Part II Course Details

Abstract
This course aims to equip students with essential programming skills for object-oriented (OO) programming, so as to lay a solid foundation for other computer science related courses. After finishing this course, students should be able to articulate and write an object-oriented program, and have adequately grasped the necessary fundamental computer programming knowledge on which other computer science courses are built.

Course Intended Learning Outcomes (CILOs)

<table>
<thead>
<tr>
<th>CILOs</th>
<th>Weighting (if app.)</th>
<th>DEC-A1</th>
<th>DEC-A2</th>
<th>DEC-A3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Describe and explain the fundamental concepts, syntax, style and structure of object-oriented programs.</td>
<td>10</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Create and develop solutions to simple problems / tasks and implement solutions with good programming practice in an object-oriented programming language.</td>
<td>60</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. Trace, analyse, validate and deduce the output of given program code of different complexities.</td>
<td>15</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>4. Demonstrate working knowledge on advanced programming features such as object initialization, inheritance, overloading and polymorphism.</td>
<td>15</td>
<td></td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>

A1: Attitude
Develop an attitude of discovery/innovation/creativity, as demonstrated by students possessing a strong sense of curiosity, asking questions actively, challenging assumptions or engaging in inquiry together with teachers.

A2: Ability
Develop the ability/skill needed to discover/innovate/create, as demonstrated by students possessing critical thinking skills to assess ideas, acquiring research skills, synthesizing knowledge across disciplines or applying academic knowledge to real-life problems.

A3: Accomplishments
Demonstrate accomplishment of discovery/innovation/creativity through producing /constructing creative works/new artefacts, effective solutions to real-life problems or new processes.

Teaching and Learning Activities (TLAs)

<table>
<thead>
<tr>
<th>TLAs</th>
<th>Brief Description</th>
<th>CILO No.</th>
<th>Hours/week (if applicable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Lecture</td>
<td>1, 2, 3, 4</td>
<td>3 hours per week</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Laboratory exercises, consisting of programming problems, are designed for students to put theory into practice and be proficient in an OO programming language. Students are required to create computer programs using a program development tool that supports an OO programming language.</td>
<td>1, 2, 3, 4</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>2</td>
<td>Laboratory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Assignment</td>
<td>Assignments are intended to require students to solve more challenging problems compared with laboratory exercises. Students have to analyse the problems, break them down into manageable sub-problems, and apply (and possibly combine) various techniques learnt from lectures and laboratory exercises in order to create and develop solutions. Then they are required to implement the solutions as computer programs which conform to good programming practice, and to explain their solutions using suitable presentation methods (e.g., using a report, flowchart, etc.).</td>
<td>1, 2, 3, 4</td>
</tr>
<tr>
<td>4</td>
<td>Quiz</td>
<td>The quizzes provide opportunities for students to reflect on their understanding of various programming concepts and problem solving techniques. They are required to develop and implement programs or program segments to solve problems; and to predict and explain the behaviour of programs involving various programming concepts and techniques.</td>
<td>1, 2, 3</td>
</tr>
</tbody>
</table>
### Assessment Tasks / Activities (ATs)

<table>
<thead>
<tr>
<th>ATs</th>
<th>CILO No.</th>
<th>Weighting (%)</th>
<th>Remarks (e.g. Parameter for GenAI use)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignments</td>
<td>1, 2, 3, 4</td>
<td>25</td>
<td>Students are required to work on assignments at least once every four weeks</td>
</tr>
<tr>
<td>Lab assessment</td>
<td>1, 2, 3, 4</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Quiz</td>
<td>1, 2, 3</td>
<td>15</td>
<td></td>
</tr>
</tbody>
</table>

**Continuous Assessment (%)**

50

**Examination (%)**

50

**Examination Duration (Hours)**

2

**Additional Information for ATs**

For a student to pass the course, at least 40% of the maximum mark for the continuous assessment and 30% of the maximum mark for the examination must be obtained.

**Assessment Rubrics (AR)**

#### Assessment Task

*Assignment*

**Criterion**

ABILITY to WRITE PROGRAM to solve simple problems with good programming practices

DEMONSTRATE working knowledge on object oriented programming

<table>
<thead>
<tr>
<th>Excellent (A+, A, A-)</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good (B+, B, B-)</td>
<td>Significant</td>
</tr>
<tr>
<td>Fair (C+, C, C-)</td>
<td>Moderate</td>
</tr>
<tr>
<td>Marginal (D)</td>
<td>Basic</td>
</tr>
<tr>
<td>Failure (F)</td>
<td>Not even reaching marginal levels</td>
</tr>
</tbody>
</table>

#### Assessment Task

*Quiz*

**Criterion**

ABILITY to ANALYZE requirements to derive the correct logic and process to solve problems

ABILITY to WRITE PROGRAM to solve simple problems with good programming practices
**Assessment Task**
Examination

**Criterion**
ABILITY to DESCRIBE and EXPLAIN the fundamental concepts, syntax, style and structure of object-oriented programs

**Excellent (A+, A, A-)**
High

**Good (B+, B, B-)**
Significant

**Fair (C+, C, C-)**
Moderate

**Marginal (D)**
Basic

**Failure (F)**
Not even reaching marginal levels

---

**Assessment Task**
Examination

**Criterion**
ABILITY to WRITE PROGRAM to solve simple problems with good programming practices

**Excellent (A+, A, A-)**
High

**Good (B+, B, B-)**
Significant

**Fair (C+, C, C-)**
Moderate

**Marginal (D)**
Basic

**Failure (F)**
Not even reaching marginal levels
Marginal (D)
Basic

Failure (F)
Not even reaching marginal levels

Assessment Task
Examination

Criterion
ABILITY to TRACE and VALIDATE computer program

Excellent (A+, A, A-)
High

Good (B+, B, B-)
Significant

Fair (C+, C, C-)
Moderate

Marginal (D)
Basic

Failure (F)
Not even reaching marginal levels

Assessment Task
Examination

Criterion
DEMONSTRATE working knowledge on object oriented programming

Excellent (A+, A, A-)
High

Good (B+, B, B-)
Significant

Fair (C+, C, C-)
Moderate

Marginal (D)
Basic

Failure (F)
Not even reaching marginal levels
Part III Other Information

Keyword Syllabus
Programming language, program development, control structures, data types, arrays, files, recursion, pointers and dynamic variables. Object-oriented programming: class and object, inheritance, overloading, polymorphism. Scoping, programming style and tools.

Syllabus:
- Computers and programming
  Hardware/software hierarchy, the computer as a multi-level language machine. Program development process and environments.
- Programming techniques and the development of algorithms
  Algorithms, programming language, modular decomposition and procedural abstraction, automatic and dynamic variables, parameter-passing by reference and by value for atomic data, objects, arrays, control structures, iteration, recursion.
- Data types and manipulation
- Program development practice
  Professional programming styles. Program testing. Program documentation.

Reading List

Compulsory Readings

<table>
<thead>
<tr>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nil</td>
</tr>
</tbody>
</table>

Additional Readings

<table>
<thead>
<tr>
<th>Title</th>
</tr>
</thead>
</table>