# SUBJECT OUTLINE

## 41012 Programming for Mechatronic Systems

| | |
|---|---|
| **Course area** | UTS: Engineering |
| **Delivery** | Autumn 2017; City |
| **Subject classification** | Field of practice: Mechatronic Engineering major |
| **Credit points** | 6cp |
| **Requisite(s)** | 48623 Mechatronics 2 |
| **Result type** | Grade and marks |

Recommended studies: Knowledge of the C language and digital systems is essential for this subject.

## Subject coordinator

**Dr Alen Alempijevic**
Email: Alen.Alempijevic@uts.edu.au
Room: CB11.09.212
Phone: +61 2 9514 2963

## Teaching staff

**Dr Alen Alempijevic**
Email: Alen.Alempijevic@uts.edu.au
Room: CB11.09.212
Phone: +61 2 9514 2963

**Dr. Gavin Paul**
Email: Gavin.Paul@uts.edu.au
Room: CB11.09.208
Phone: +61 2 9514 2969

**Mr Alex Virgona (Tutor)**
Email: Alexander.Virgona@uts.edu.au
Room: CB11.09.300

**Contacting staff:**

If you wish to discuss your questions or need further help understanding concepts in the subject, please discuss these first with your lecturer or tutor in lab classes. If you have a question it may be better posted to the Discussion Board on UTSOnline so other students benefit from the response. Also, you may get a quicker response from another students.

Email messages will be responded to within two working days, though they are discouraged as inefficient means for the subject matter. Phone messages will not be responded to.

## Subject description

The subject presents the theoretical foundations of object-oriented programming and design using C++ and provides students with skills in the design and implementation of a code base for mechatronics systems of moderate complexity. This subject familiarises students with approaches to design and implement code that is modular, re-usable, reliable and maintainable.

Working on an individual project that requires the design and implementation of part of a mechatronic system allows students to apply their knowledge to a real-life problem.

Topics include: objects, classes, abstraction, inheritance, polymorphism, run-time instantiation, threading, thread communication, event handling, error handling, use of generic features of C++ such as the STL and utilising APIs, documentation and testing.

Class time is used for lectures, laboratories and project work. There are a number of formal laboratory sessions that apply theory to practical problems, which also familiarise students with the state of the art of mechatronic systems.

## Subject learning objectives (SLOs)

Upon successful completion of this subject students should be able to:

1. Design classes that are reusable, reliable and maintainable
2. Apply theoretical knowledge of sensors and control to practical programming problems
3. Select appropriate class structures and data handling methods for task at hand
4. Implement and test object-oriented applications of moderate complexity
5. Communicate programming design decisions, dependencies, interconnections, use cases and testing procedures in a written document

## Course intended learning outcomes (CILOs)

This subject also contributes specifically to the development of the following faculty Course Intended Learning Outcomes (CILOs) and Engineers Australia (EA) Stage 1 competencies:

- Problem solving and design - Engineering practice focuses on problem solving and design where artifacts are conceived, created, used, modified, maintained and retired. (B.0)
- Identify and apply relevant problem solving methodologies, which is linked to EA Stage 1 Competencies: 1.1, 2.1, 2.2, 2.3 (B.1)
- Design components, systems and/or processes to meet required specifications, which is linked to EA Stage 1 Competencies: 1.3, 1.6, 2.1, 2.2, 2.3 (B.2)
- Synthesise alternative/innovative solutions, concepts and procedures, which is linked to EA Stage 1 Competencies: 1.1, 3.3 (B.3)
- Apply decision making methodologies to evaluate solutions for efficiency, effectiveness and sustainability, which is linked to EA Stage 1 Competencies: 1.2, 2.1 (B.4)
- Implement and test solutions, which is linked to EA Stage 1 Competencies: 2.2, 2.3 (B.5)
- Develop models using appropriate tools such as computer software, laboratory equipment and other devices, which is linked to EA Stage 1 Competencies: 2.2,2.3, 2.4 (C.2)
- Communicate effectively in ways appropriate to the discipline, audience and purpose, which is linked to EA Stage 1 Competency: 3.2 (E.1)
- Be able to conduct critical self-review and performance evaluation against appropriate criteria as a primary means of tracking personal development needs and achievements, which is linked to EA Stage 1 Competency: 3.5 (F.1)

## Teaching and learning strategies

The subject uses a model of brief interactive lectures followed by class activities to support student learning. The students are expected to go through online materials and videos before coming to class as instructed in the program so they are prepared for class activities. These activities include collaborative discussions, problem solving and hands on activities. Students will be guided and assisted to reach learning objectives through a series of milestones with deliverables strategically set throughout the session. A code peer review mechanism enables students to gain a wider appreciation of applying theoretical knowledge on given tasks and the merits of appropriate technical solutions. Students undertake an individual project that reinforces the theoretical content and culminates in a formal submission of code and appropriate documentation for evaluation.

## Content (topics)

The following topics will be covered:

• The Objected Orientated Paradigm: objects, classes, abstraction, inheritance, polymorphism

• Basics of initialization and memory handling: compile time / run-time initialisation, heap and stack

• Basics of multi-threading, cross thread communication and error handling: synchronizing threads, sharing data across threads, mutexes/semaphores, exclusive locking, asynchronous events

•The use of generic features of C++ such as the STL containers and data structures, selection of appropriate structures for effective programming

• Basics of libraries, using APIs demonstrated on state of the art Open Source Software. Designing, Developing and Documenting an API.

• Basics of the Component Based Software Engineering approach, with the Robotics Operating System (ROS) as an example

• Basics of utilisng code automation and auto documentation generation tools (CMake and Doxygen)

Each of these topics addresses an important aspect in programming modern Mechatronics systems. The intention is that, as you work your way through the subject, your learning will be cumulative. That is, the content you cover in one section should directly help you to understand the topics that follow. For each of the above topics, a separate list of sub-topics and weekly learning schedule is given in the Program.

The students' existing knowledge of sensor and control, combined with activities to understand and appreciate the object oriented paradigm is leveraged to complete a final individual project.

## Program

| Week/Session | Dates | Description |
|---|---|---|
| 01 | 13 Mar | Orientation and Preparation **(No formal class)**<br><br>**Notes:**<br><br>**Pre-reading**<br><br>Log on to UTSOnline and watch the videos designated to provide a general introduction to the subject, the **syntax of C++** , drawing from your knowledge of C. The Linux OS is introduced (the operating system used in the subject), **CMake** tools as well as the recommended Integrated Development Environment (IDE) **QtCreator.** |
| 02 | 20 Mar | Ensure you have covered Orientation and Preparation weeks materials, knowledge of them are essential for in class activities.<br><br>Active hands on: intro to C++ paradigm, functions, pointers, macros<br><br>**Notes:**<br><br>**Practise Quiz** in interactive Lecture<br><br>**Pred-reading:** Read the slides and watch the videos on **Functions** and **Pre-Processor**.<br><br>**PROVIDED - Assessment I : Developing Sensor Class** |
| 03 | 27 Mar | Ensure you have covered pre-readings, knowledge of them is essential for in class activities.<br><br>Active hands on: C++ paradigm, objects, operators, polymorphism, example of a sensor class<br><br>**Notes:**<br><br>**Quiz 1** in interactive Lecture<br><br>**Pre-reading:** Read the slides and watch the videos on **Classes, Data Types and Operators** |

| | | |
|---|---|---|
| | | **DUE - Assessment I: Developing Sensor Class** |
| 04 | 03 Apr | Ensure you have covered pre-readings, knowledge of them is essential for in class activities.<br>Active hands on: objects, overriding, operators, inheritance<br>**Notes:**<br>**Quiz 2** in interactive Lecture<br>**Pre-reading:** Read the slides and watch the videos on **Inheritance**<br>**DUE - Peer Code Review I** |
| 05 | 10 Apr | Ensure you have covered pre-readings, knowledge of them is essential for in class activities. General feedback from Assignment 1 provided.<br>Active hands on: objects, overriding, inheritance, polymorphism, templates and STL<br>**Notes:**<br>**Quiz 3** in interactive Lecture<br>**Pre-reading:** Read the slides and watch the videos on **Templates** and **STL**<br>**Notes:**<br>**FEEDBACK - Assessment I: Developing Sensor Class**<br>**PROVIDED - Assessment II: Utilising Abstraction for a Range of Sensor Classes** |
| 06 | 17 Apr | **Public holiday, no class**<br>**Notes:**<br>Ensure you have attempted exercises available on UTSOnline on: encapsulation, inheritance, polymorphism, templates and STL containers. |
| SV1 | 24 Apr | **Mid-session StuVac**.<br>**Notes:**<br>**DUE - Assessment II: Utilising Abstraction for a Range of Classes** |
| 07 | 01 May | Ensure you have covered pre-readings, knowledge of them is essential for in class activities.<br>Active hands on: data structures and algorithms<br>**Notes:**<br>**Pre-reading:** Read slides and watch the videos on **Data Structures** and **Algoritms** |

| | | |
|---|---|---|
| 08 | 08 May | Ensure you have covered pre-readings, knowledge of them is essential for in class activities. General feedback from Assignment 2 provided.<br><br>Active hands on: multi-threaded programming, synchronisation (mutexes, conditional variables)<br><br>**Notes:**<br><br>**Quiz 4** in interactive Lecture<br><br>**Pre-reading:** Read the slides and watch the videos on **Threading** and **Synchronisation**<br><br>**PROVIDED - Assessment III: Threading, Synchronisation and Data Integrity**<br><br>**DUE - Peer Code Review II** |
| 09 | 15 May | Ensure you have covered pre-readings, knowledge of them is essential for in class activities.<br><br>Active hands on: using libraries, developing a library, unit testing, documentation via Doxygen<br><br>**Notes:**<br><br>**Quiz 5** in interactive Lecture<br><br>**Pre-reading:** Read the slides and watch the videos on **Doxygen, Unit Testing, Libraries** and **OpenCV** (exmaple of modern library) |
| 10 | 22 May | Ensure you have covered pre-readings, knowledge of them is essential for in class activities.<br><br>Active hands on: ROS as example of CBSE, designing and using systems, middleware subscriber / publisher (topic and service)<br><br>**Notes:**<br><br>**Quiz 6** in interactive Lecture<br><br>**Pre-reading:** Read the slides and watch videos on **Component Based Software Engineering (CBSE)** and **ROS**<br><br>**FEEDBACK - Assessment II: Utilising Abstraction for a Range of Sensor Classes**<br><br>**DUE - Assessment III: Threading, Synchronisation and Data Integrity** |
| 11 | 29 May | Ensure you have covered pre-readings, knowledge of them is essential for in class activities.<br><br>Active hands on: using ROS ecosystem and using existing libraries (OpenCV)<br><br>**Notes:**<br><br>**Pre-reading:** Read slides and watch the videos on **ROS**<br><br>**PROVIDED - Final Assessment: Individual Project**<br><br>**DUE - Peer Code Review III** |

|  |  |  |
|---|---|---|
| 12 | 05 Jun | General feedback from Assignment 3 provided. Students work on projects - mentoring<br><br>**Notes:**<br><br>**FEEDBACK - Assessment III: Threading, Synchronisation and Data Integrity** |

## Additional information

***Repeated Failure in this Subject***

The Faculty takes repeated failures in a subject seriously and enforces Rule 10.6 of the University's Student and Related Rules. You should read these rules and be aware of the consequences of failure.

If you have failed **twice** before in this subject, then:

(i) You must seek advice from the Subject Coordinator. You will be asked to draw up and submit a study plan that outlines your strategy for passing this subject on the third attempt. A signed copy of this study plan will be kept by the Faculty for internal records.

(ii) If you do not seek advice from the Subject Coordinator by Week 2, then you do not have the Faculty's permission to enrol in the subject. If you stay enrolled in the subject then you will be breaking Rule 10.6.2 (1) of the University's Student and Related Rules.

(iii) You need to be aware that if you fail this subject for a third time, you will need to seek permission from the Deputy Head of School (Teaching & Learning) for any further enrolment in this subject (see below).

If you fail this subject for a **third** time, then:

(i) The Subject Coordinator will deny permission for any further enrolment unless you can produce documentary evidence of extenuating circumstances that require special consideration. In such cases, the Subject Coordinator will refer the matter to the Deputy Head of School (Teaching & Learning), who will grant or deny enrolment for a fourth or subsequent attempt based on a student's overall performance in the course and the extent to which extenuating circumstances have contributed to one or more of the failures.

(ii) If you are granted permission for a fourth or subsequent attempt at this subject, then you must seek continuing assistance throughout this semester from the Subject Coordinator.

## Assessment

***Late Submission of Assessment Tasks***

Unless otherwise specified, late submission of an assessment task will attract a 20% penalty per working day, up to a maximum of 5 working days. If late submission of an assessment item is due to extenuating or special circumstances beyond your control, then you should contact the Subject Coordinator.

### Assessment task 1: Developing Sensor Class

|  |  |
|---|---|
| **Intent:** | Skills in utilising, classes, functions, pointers and macros will be examined in developing sensor class on a provided mechatronics sensor specification. |
| **Objective(s):** | This assessment task addresses the following subject learning objectives (SLOs):<br><br>1, 2 and 4<br><br>This assessment task contributes to the development of the following course intended learning outcomes (CILOs):<br><br>B.0, B.5 and C.2 |

**Type:** Case study

**Groupwork:** Individual

**Weight:** 5%

**Task:** Write a program in C++ using object oriented paradigms that implements a given Mechatronics senor in line with a specification. The solution will need to provide appropriate access to sensor attributes (encapsulates the parameters) and enable access to underlying sensor data and attributes.

**Due:** 11.59pm Sunday 2 April 2017

**Criteria linkages:**

| Criteria | Weight (%) | SLOs | CILOs |
|---|---|---|---|
| Encapsulation of all sensor attributes and data with appropriate access methods | 50 | 2 | B.0 |
| Proper code execution | 25 | 4 | B.5 |
| Modularity of software | 25 | 1 | C.2 |

SLOs: subject learning objectives
CILOs: course intended learning outcomes

## Assessment task 2: Utilising Abstraction for a Range of Sensor Classes

**Intent:** Skills in utilising, classes, functions, pointers and utilising abstraction, encapsulation, inheritance, polymorphism with appropriate documentation will be assessed.

**Objective(s):** This assessment task addresses the following subject learning objectives (SLOs):

1, 2, 4 and 5

This assessment task contributes to the development of the following course intended learning outcomes (CILOs):

B.2, B.5, C.2 and E.1

**Type:** Case study

**Groupwork:** Individual

**Weight:** 20%

**Task:** Write a program in C++ using object oriented paradigms that embodies a range of Mechatronics sensors, utilising abstraction, encapsulation, inheritance and polymorphism. Supply appropriate auto-generated documentation utilising inline source mark-up.

**Due:** 11.59pm Sunday 30 April 2017

**Criteria linkages:**

| Criteria | Weight (%) | SLOs | CILOs |
|---|---|---|---|
| Sensor classes exploits abstraction (encapsulation, inheritance and polymorphism) to cover a range of sensors | 40 | 2 | B.2 |
| Proper code execution | 20 | 4 | B.5 |
| Documentation | 20 | 5 | E.1 |

| | | | |
|---|---|---|---|
| Modularity of software | 20 | 1 | C.2 |

SLOs: subject learning objectives
CILOs: course intended learning outcomes

## Assessment task 3: Threading, Synchronisation and Data Integrity

**Intent:** Skills in utilising, classes, abstraction, data structures, threading, data synchronisation and documentation will be assessed.

**Objective(s):** This assessment task addresses the following subject learning objectives (SLOs):

1, 2, 4 and 5

This assessment task contributes to the development of the following course intended learning outcomes (CILOs):

B.3, B.5, C.2 and E.1

**Type:** Case study

**Groupwork:** Individual

**Weight:** 20%

**Task:** Write a program in C++ using object oriented paradigms that shares data originating from a range of Mechatronics sensors between a number of threads. Ensure data integrity between threads and enable relating data between them via suitable data structure that enables time synchronisation and subsequently interpolation of data for task at hand. Supply appropriate auto-generated documentation utilising inline source mark-up.

**Due:** 11.59pm Sunday 21 May 2017

**Criteria linkages:**

| Criteria | Weight (%) | SLOs | CILOs |
|---|---|---|---|
| Use of appropriate data structures, locking mechanisms, data sorting mechanisms | 40 | 2 | B.3 |
| Proper code execution | 20 | 4 | B.5 |
| Documentation | 20 | 5 | E.1 |
| Modularity of software | 20 | 1 | C.2 |

SLOs: subject learning objectives
CILOs: course intended learning outcomes

## Assessment task 4: Peer Code Review

**Intent:** The code review enables students to gain a wider appreciation of applying theoretical knowledge on given tasks, benchmark their own solutions against other students and develop insights into the vast array of appropriate technical solutions

**Objective(s):** This assessment task addresses the following subject learning objectives (SLOs):

5

This assessment task contributes to the development of the following course intended learning outcomes (CILOs):

E.1 and F.1

**Type:** Reflection

**Groupwork:** Individual

**Weight:** 9%

**Due:** Three peer reviews are undertaken. Due on Sunday 23:59 on the week feedback is provided (ie feedback provided Monday, peer review due Sunday 23:59)

**Criteria linkages:**

| Criteria | Weight (%) | SLOs | CILOs |
|---|---|---|---|
| Mark and comment according to the Assignment marking criteria | 60 | 5 | E.1 |
| Provide insights on potential limitations of existing design, with an example test / code suggestion | 40 | 5 | F.1 |

SLOs: subject learning objectives
CILOs: course intended learning outcomes

### Assessment task 5: Individual Project

**Intent:** To analyse a set of specifications, design, test, document and practically evaluate code to perform analysis of sensor data and simple actions based on data on a simulated robotic platform.

**Objective(s):** This assessment task addresses the following subject learning objectives (SLOs):

1, 3, 4 and 5

This assessment task contributes to the development of the following course intended learning outcomes (CILOs):

B.2, B.4, B.5, C.2 and E.1

**Type:** Project

**Groupwork:** Individual

**Weight:** 40%

**Task:** Write a series of components that exploits ROS CBSE framework to share data originating from a range of Mechatronics sensors between a number of threads and components. Employ suitable number of threads per component and data structures that enables time synchronisation and subsequently interrogation of data to allow simple actions of a robotic platform. Supply appropriate auto-generated documentation utilising inline source mark-up. Exploit unit testing framework with test cases evaluating code.

**Due:** 11.59pm Sunday 25 June 2017

**Criteria linkages:**

| Criteria | Weight (%) | SLOs | CILOs |
|---|---|---|---|
| Use of appropriate data structures, data exchange, data sorting mechanisms, components | 20 | 1 | B.2 |
| Use of appropriate threading, topics and service mechanisms | 20 | 3 | B.4 |
| Proper code execution with appropriate unit testing framework | 20 | 4 | B.5 |
| Supporting documentation | 20 | 5 | E.1 |
| Modularity of software | 20 | 1 | C.2 |

SLOs: subject learning objectives
CILOs: course intended learning outcomes

### Assessment task 6: Review Quizes

**Intent:** Test the student's knowledge on Object Oriented Programming with C++, in an incremental manner. Provide feedback to students throughout the session.

**Objective(s):** This assessment task addresses the following subject learning objectives (SLOs):

1 and 3

This assessment task contributes to the development of the following course intended learning outcomes (CILOs):

B.1 and B.2

**Type:** Quiz/test

**Groupwork:** Individual

**Weight:** 6%

**Due:** Review Quizzes are undertaken as in-class activities in interactive Lectures, refer to program for due dates.

**Criteria linkages:**

| Criteria | Weight (%) | SLOs | CILOs |
|---|---|---|---|
| Correctness of answer | 50 | 3 | B.1 |
| Justication of results | 50 | 1 | B.2 |

SLOs: subject learning objectives
CILOs: course intended learning outcomes

## Use of plagiarism detection software

Turnitin software is used to check the originality in all assessment reports

## Assessment feedback

Formative individual feedback on each assignment will be provided via UTSOnline within two weeks of Assignment submission deadline.A graded rubric with each element of the assignment (mapped to marking criteria) contains a mark and constructive feedback. In addition, general feedback is provided on the merits and weaknesses of the submission with suggestions on how to improve the submission to reach learning objectives.

Group intercommonalties will be used as feedback to entire class at Lectures where a deconstruction of an exemplar submission is undertaken and approaches in problem solving and code modularity/design are contrasted.

## Recommended texts

Elliot B. Koffman & Paul A.T. Wolfgang, *Objects, Abstraction, Data Structures and Design Using C++,* John Wiley & Sons, Inc
ISBN 0-471-46755-3

## References

Roberts, E.,*Programming abstractions in C++*, Pearson, 2014

D. Ryan Stephens; Christopher Diggins; Jonathan Turkanis, Jeff Cogswell, *C++ Cookbook*, O'Reilly Media, Inc., 2005

Lippman, Stanley B, *C++ primer*, Addison-Wesley, 2005

## Other resources

UTSOnline provides online access to notes in PDF format, links to online lectures, examples and a Discussion Board. Faculty Linux Computer Laboratories in Building 11 have been set up with Linux Ubuntu which will be used to deliver content.

## Graduate attribute development

For a full list of the faculty's graduate attributes and EA Stage 1 competencies, refer to the Student Guide.

## Assessment: faculty procedures and advice

**Special Consideration**

If you believe your performance in an assessment item or exam has been adversely affected by circumstances beyond your control, such as a serious illness, loss or bereavement, hardship, trauma, or exceptional employment demands, you may be eligible to apply for Special Consideration.

## Academic integrity

Work submitted electronically may be subject to similarity detection software. Student work must be submitted in a format able to be assessed by the software (e.g. doc, pdf (text files), rtf, html).

For information about avoiding plagiarism see:

https://avoidingplagiarism.uts.edu.au
www.gsu.uts.edu.au/rules/student/section-16.html#r16.2

## Academic liaison officer

Academic liaison officers (ALOs) are academic staff in each faculty who assist students experiencing difficulties in their studies due to: disability and/or an ongoing health condition; carer responsibilities (e.g. being a primary carer for small children or a family member with a disability); and pregnancy.

ALOs are responsible for approving adjustments to assessment arrangements for students in these categories. Students who require adjustments due to disability and/or an ongoing health condition are requested to consult a disability services officer in the Special Needs Service before speaking to the relevant ALO.

The ALO for undergraduate students is:

Chris Wong
telephone +61 2 9514 4501

The ALO for postgraduate students is:

Associate Professor Rob Jarman
telephone +61 2 9514 2368

## Disclaimer

This outline serves as a supplement to the Faculty of Engineering and Information Technology Student Guide. On all matters not specifically covered in this outline, the requirements specified in the Student Guide apply.