COIT20245 *Introduction to Programming*
Term 1 - 2024

Profile information current as at 19/05/2024 05:23 am

General Information

Overview
In this unit, you will apply computational thinking to develop fundamental algorithms for specified problems and implement them using Python. It is assumed that you have little or no programming experience. You will apply problem-solving techniques such as decomposition and abstraction. You will learn about the parts of a program, including variables, types, control structures and methods. A key aspect of this unit is practical, hands-on development and testing, which you will do in an industry standard Integrated Development Environment (IDE).

Details
Career Level: *Postgraduate*
Unit Level: *Level 8*
Credit Points: *6*
Student Contribution Band: *8*
Fraction of Full-Time Student Load: *0.125*

Pre-requisites or Co-requisites
Anti-requisite: COIT29222 Programming Principles.
Important note: Students enrolled in a subsequent unit who failed their pre-requisite unit, should drop the subsequent unit before the census date or within 10 working days of Fail grade notification. Students who do not drop the unit in this timeframe cannot later drop the unit without academic and financial liability. See details in the Assessment Policy and Procedure (Higher Education Coursework).

Offerings For Term 1 - 2024

- Brisbane
- Melbourne
- Online
- Rockhampton
- Sydney

Attendance Requirements
All on-campus students are expected to attend scheduled classes – in some units, these classes are identified as a mandatory (pass/fail) component and attendance is compulsory. International students, on a student visa, must maintain a full time study load and meet both attendance and academic progress requirements in each study period (satisfactory attendance for International students is defined as maintaining at least an 80% attendance record).

Website
This unit has a website, within the Moodle system, which is available two weeks before the start of term. It is important that you visit your Moodle site throughout the term. Please visit Moodle for more information.

Class and Assessment Overview

Recommended Student Time Commitment

Each 6-credit Postgraduate unit at CQUniversity requires an overall time commitment of an average of 12.5 hours of study per week, making a total of 150 hours for the unit.

Class Timetable

Regional Campuses
Bundaberg, Cairns, Emerald, Gladstone, Mackay, Rockhampton, Townsville

Metropolitan Campuses
Adelaide, Brisbane, Melbourne, Perth, Sydney

Assessment Overview

1. **Practical Assessment**
Weighting: 30%
2. **Portfolio**
Weighting: 30%
3. **In-class Test(s)**
Weighting: 40%

Assessment Grading

This is a graded unit: your overall grade will be calculated from the marks or grades for each assessment task, based on the relative weightings shown in the table above. You must obtain an overall mark for the unit of at least 50%, or an overall grade of 'pass' in order to pass the unit. If any 'pass/fail' tasks are shown in the table above they must also be completed successfully ('pass' grade). You must also meet any minimum mark requirements specified for a particular assessment task, as detailed in the 'assessment task' section (note that in some instances, the minimum mark for a task may be greater than 50%). Consult the University's Grades and Results Policy for more details of interim results and final grades.

CQUniversity Policies

All University policies are available on the CQUniversity Policy site.
You may wish to view these policies:

- Grades and Results Policy
- Assessment Policy and Procedure (Higher Education Coursework)
- Review of Grade Procedure
- Student Academic Integrity Policy and Procedure
- Monitoring Academic Progress (MAP) Policy and Procedure – Domestic Students
- Monitoring Academic Progress (MAP) Policy and Procedure – International Students
- Student Refund and Credit Balance Policy and Procedure
- Student Feedback – Compliments and Complaints Policy and Procedure
- Information and Communications Technology Acceptable Use Policy and Procedure

This list is not an exhaustive list of all University policies. The full list of University policies are available on the CQUniversity Policy site.

Previous Student Feedback

Feedback, Recommendations and Responses

Every unit is reviewed for enhancement each year. At the most recent review, the following staff and student feedback items were identified and recommendations were made.

Feedback from Unit coordinator reflection.

Feedback
Attendance particularly in lectures has been poor.

Recommendation
Possibly automate attendance recording in large lectures. Contact students with poor attendance and identify them as being at risk.

Feedback from Unit coordinator reflection.

Feedback
Academic misconduct has been a problem especially collusion.

Recommendation
Continue to educate students on the consequences of academic misconduct. Possibly employ third party software to detect such breaches. With such a large cohort it would be difficult to individualise the various assessment items.

Feedback from Student feedback.

Feedback
Some students find the unit's advanced level challenging, particularly when Java is used as the primary programming language, which may not align with the interests of all students in future programming pursuits.

Recommendation
Consider introducing Python as the primary teaching tool, as it could provide a more approachable and engaging experience for students navigating the complexities of an introductory unit with advanced content in Java.

Feedback from Student feedback.

Feedback
Students are happy with the teaching staff.

Recommendation
Continue employing experienced and dedicated teaching staff.

Unit Learning Outcomes

On successful completion of this unit, you will be able to:

1. Implement, document and refactor functions that use Python's syntax, data representations, scope rules, and procedural concepts including iterations and conditionals
2. Devise algorithms using computational thinking techniques (decomposition and abstraction) and communicate algorithms (oral and written)
3. Use industry tools to efficiently and ethically develop quality applications (Integrated Development Environment (IDE), debugger, linter, Generative AI and version control)
4. Demonstrate secure coding practices (variable typing and scoping, testing and input validation)
5. Develop modules that implement standard algorithms (searching, sorting), process hierarchical data (JSON), and adhere to design principles (coupling and cohesion) and construct applications that use modules and Python libraries.

The Australian Computer Society (ACS), the professional association for Australia's ICT sector, recognises the Skills Framework for the Information Age (SFIA). SFIA is adopted by organisations, governments, and individuals in many countries and provides a widely used and consistent definition of ICT skills. SFIA is increasingly being used when developing job descriptions and role profiles. ACS members can use the tool MySFIA to build a skills profile.
This unit contributes to the following workplace skills as defined by SFIA 8 (the SFIA code is included):

- Programming/Software Development (PROG)
- Testing (TEST)
- Methods and tools (METL)

Alignment of Learning Outcomes, Assessment and Graduate Attributes

— N/A Level ∘ Introductory Level • Intermediate Level • Graduate Level ∘ Professional Level ∘ Advanced Level

Alignment of Assessment Tasks to Learning Outcomes

Assessment Tasks	Learning Outcomes								
	1					2	3	4	5
1 - Knowledge	∘	∘	∘	∘	∘				
2 - Communication	∘	∘							
3 - Cognitive, technical and creative skills									
4 - Research									
5 - Self-management									
6 - Ethical and Professional Responsibility			∘	∘					
7 - Leadership									
8 - Aboriginal and Torres Strait Islander Cultures									

Textbooks and Resources

Textbooks

COIT20245

Prescribed

Java How to Program : Early Objects Edition
11th Edition (2018)
Authors: Paul Deitel and Harvey Deitel
Pearson Education
Upper Saddle River , NJ , USA
ISBN: 9780134743356
Binding: Paperback

View textbooks at the CQUniversity Bookshop

IT Resources

You will need access to the following IT resources:

- CQUniversity Student Email
- Internet
- Unit Website (Moodle)
- Visual Studio Code (latest version)
- Python PyPI (pip) packages including black, pylint and requests
- Python 3.10 (or higher)

Referencing Style

All submissions for this unit must use the referencing style: Harvard (author-date)
For further information, see the Assessment Tasks.

Teaching Contacts

Jamie Shield Unit Coordinator
j.shield@cqu.edu.au

Schedule

Week 1 - 04 Mar 2024

Module/Topic	Chapter	Events and Submissions/Topic
1 Introduction		

Week 2 - 11 Mar 2024

Module/Topic	Chapter	Events and Submissions/Topic
2 Conditionals, Functions and Lists		

Week 3 - 18 Mar 2024

Module/Topic	Chapter	Events and Submissions/Topic
3 Map, Filter, Else and Methods		A1 Quiz 1 Conditionals...(6%)

Week 4 - 25 Mar 2024

Module/Topic	Chapter	Events and Submissions/Topic
4 Dictionaries and Modules		A1 Quiz 2 Map...(6%)

Week 5 - 01 Apr 2024

Module/Topic	Chapter	Events and Submissions/Topic
5 Nesting		A1 Quiz 3 Dictionaries... (6%)
Vacation Week - 08 Apr 2024		
Module/Topic	**Chapter**	**Events and Submissions/Topic**
No classes		
Week 6 - 15 Apr 2024		
Module/Topic	**Chapter**	**Events and Submissions/Topic**
6 Algorithms		A1 Quiz 4 Nesting (6%)
Week 7 - 22 Apr 2024		
Module/Topic	**Chapter**	**Events and Submissions/Topic**
7 Requests and IDEs		A1 Quiz 5 Algorithms (6%)
Week 8 - 29 Apr 2024		
Module/Topic	**Chapter**	**Events and Submissions/Topic**
8 Mutability		
Week 9 - 06 May 2024		
Module/Topic	**Chapter**	**Events and Submissions/Topic**
9 Sorting		
Week 10 - 13 May 2024		
Module/Topic	**Chapter**	**Events and Submissions/Topic**
10 Design		A2 Project (30%)
Week 11 - 20 May 2024		
Module/Topic	**Chapter**	**Events and Submissions/Topic**
11 Review		
Week 12 - 27 May 2024		
Module/Topic	**Chapter**	**Events and Submissions/Topic**
Test in tutorial (No lecture)		A3 In-class Test (40%) Oncampus students must attend the Week 12 tutorial. You must achieve 25% in the test to pass the unit.
Review/Exam Week - 03 Jun 2024		
Module/Topic	**Chapter**	**Events and Submissions/Topic**
Exam Week - 10 Jun 2024		
Module/Topic	**Chapter**	**Events and Submissions/Topic**

Term Specific Information

Unit Coordinator: Jamie Shield, Cairns, j.shield@cqu.edu.au
Oncampus students must attend the Week 12 tutorial. Distance students must organise their own exam including the location, computer resources and supervisor who will be vetted by the unit coordinator.
It is expected that your tutorial will occur earlier in the week than your lecture.
There is NO prescribed textbook for this unit.

Assessment Tasks

1 Quizzes

Assessment Type
Practical Assessment

Task Description
There are five quizzes. You will be assessed on key concepts in programming such as variables, types, control structures and methods. You will apply fundamental algorithms for specified problems and implement them using Python.

Assessment Due Date

Weeks 3, 4, 5, 6 and 7

Return Date to Students

Immediate feedback

Weighting
30%

Assessment Criteria
This assessment consists of small programming activities. Each question will be marked on aspects such as functionality, coding style, documentation of code and testing, error handling, no use of banned language features, variable naming, code reuse and referencing.

Referencing Style

- Harvard (author-date)

Submission
Online

Submission Instructions
Complete the quiz on the unit website.

Learning Outcomes Assessed

- Implement, document and refactor functions that use Python's syntax, data representations, scope rules, and procedural concepts including iterations and conditionals
- Devise algorithms using computational thinking techniques (decomposition and abstraction) and communicate algorithms (oral and written)
- Use industry tools to efficiently and ethically develop quality applications (Integrated Development Environment (IDE), debugger, linter, Generative AI and version control)
- Demonstrate secure coding practices (variable typing and scoping, testing and input validation)

2 Project

Assessment Type
Portfolio

Task Description
You will develop a Python application that interfaces with web services such as geocoding and movie web APIs. Your application will be modular, for example, you will create wrapper modules for each web service. Your application will safely request and validate data from the webservices and then analyse and report the data retrieved. You may only use the language features taught in this unit.
You must complete the project alone or in groups of 2 or 3 people. You will be responsible for creating your own groups. All group members must be identified in the groupwork artefacts. Evidence must be provided that all group members contributed adequately to the final submissions. All group members must submit via the unit website. The moderation process might allocate group members different marks. Sharing of artefacts, for example, code, between groups is not permitted.

Assessment Due Date

Week 10

Return Date to Students

Feedback will be provided within 2 weeks of the due date.

Weighting
30%

Assessment Criteria
You will be marked on aspects such as evidence of contribution to your group, functionality, coding style, quality of test plan, documentation of code and testing, ease of use, error handling, no use of banned language features, variable naming, use of Git repository, code reuse and referencing.

Referencing Style

- Harvard (author-date)

Submission
Online

Submission Instructions
Submit artefacts to both a private code repository and to the unit website. Submit a link to your private repository to the unit website. All group members must submit.

Learning Outcomes Assessed

- Implement, document and refactor functions that use Python's syntax, data representations, scope rules, and procedural concepts including iterations and conditionals
- Devise algorithms using computational thinking techniques (decomposition and abstraction) and communicate algorithms (oral and written)
- Use industry tools to efficiently and ethically develop quality applications (Integrated Development Environment (IDE), debugger, linter, Generative AI and version control)
- Demonstrate secure coding practices (variable typing and scoping, testing and input validation)
- Develop modules that implement standard algorithms (searching, sorting), process hierarchical data (JSON), and adhere to design principles (coupling and cohesion) and construct applications that use modules and Python libraries.

3 Test

Assessment Type
In-class Test(s)

Task Description
You must attend the Week 12 tutorial to complete an in-class test (exam). You must achieve 25% in the test to pass the unit. You will complete an online quiz similar to those in Assignment 1. The test duration is 120 minutes. There is no perusal time. The test is closed book. No calculators or phones are permitted. It is forbidden to access the Internet, Visual Studio Code or any other applications including web browser developer tools. You may bring in one A4 sheet of paper printed on both sides. You must bring your student card for identification. Distance students must organise their own exam including the location, computer resources and supervisor who will be vetted by the unit coordinator.

Assessment Due Date

In-class in Week 12 tutorial

Return Date to Students

Feedback will be returned on the Certification of Grades day.

Weighting
40%

Minimum mark or grade
25%

Assessment Criteria
This assessment consists of activities such as programming, documentation and testing. Each question will be marked on aspects such as functionality, coding style, documentation of code and testing, error handling, no use of banned language features, variable naming, code reuse and referencing.

Referencing Style

- Harvard (author-date)

Submission
Online

Submission Instructions
Submit in-class in Week 12 tutorial.

Learning Outcomes Assessed

- Implement, document and refactor functions that use Python's syntax, data representations, scope rules, and procedural concepts including iterations and conditionals
- Devise algorithms using computational thinking techniques (decomposition and abstraction) and communicate algorithms (oral and written)
- Demonstrate secure coding practices (variable typing and scoping, testing and input validation)
- Develop modules that implement standard algorithms (searching, sorting), process hierarchical data (JSON), and adhere to design principles (coupling and cohesion) and construct applications that use modules and Python libraries.

Academic Integrity Statement

As a CQUniversity student you are expected to act honestly in all aspects of your academic work.

Any assessable work undertaken or submitted for review or assessment must be your own work. Assessable work is any type of work you do to meet the assessment requirements in the unit, including draft work submitted for review and feedback and final work to be assessed.

When you use the ideas, words or data of others in your assessment, you must thoroughly and clearly acknowledge the source of this information by using the correct referencing style for your unit. Using others' work without proper acknowledgement may be considered a form of intellectual dishonesty.

Participating honestly, respectfully, responsibly, and fairly in your university study ensures the CQUniversity qualification you earn will be valued as a true indication of your individual academic achievement and will continue to receive the respect and recognition it deserves.

As a student, you are responsible for reading and following CQUniversity's policies, including the **Student Academic Integrity Policy and Procedure**. This policy sets out CQUniversity's expectations of you to act with integrity, examples of academic integrity breaches to avoid, the processes used to address alleged breaches of academic integrity, and potential penalties.

What is a breach of academic integrity?
A breach of academic integrity includes but is not limited to plagiarism, self-plagiarism, collusion, cheating, contract cheating, and academic misconduct. The Student Academic Integrity Policy and Procedure defines what these terms mean and gives examples.

Why is academic integrity important?
A breach of academic integrity may result in one or more penalties, including suspension or even expulsion from the University. It can also have negative implications for student visas and future enrolment at CQUniversity or elsewhere. Students who engage in contract cheating also risk being blackmailed by contract cheating services.

Where can I get assistance?
For academic advice and guidance, the Academic Learning Centre (ALC) can support you in becoming confident in completing assessments with integrity and of high standard.

What can you do to act with integrity?

Be Honest
If your assessment task is done by someone else, it would be dishonest of you to claim it as your own

Seek Help
If you are not sure about how to cite or reference in essays, reports etc, then seek help from your lecturer, the library or the Academic Learning Centre (ALC)

Produce Original Work
Originality comes from your ability to read widely, think critically, and apply your gained knowledge to address a question or problem