Contents

General Information 2
Learning Outcomes 3
General Assessment Information 3
Assessment Tasks 3
Delivery and Resources 8
Unit Schedule 9
Policies and Procedures 10
Graduate Capabilities 12
Changes from Previous Offering 16
Grading Standards 18

Disclaimer
Macquarie University has taken all reasonable measures to ensure the information in this publication is accurate and up-to-date. However, the information may change or become out-dated as a result of change in University policies, procedures or rules. The University reserves the right to make changes to any information in this publication without notice. Users of this publication are advised to check the website version of this publication [or the relevant faculty or department] before acting on any information in this publication.

https://unitguides.mq.edu.au/unit_offerings/103824/unit_guide/print
## General Information

### Unit convenor and teaching staff
Convenor, Lecturer  
Gaurav Gupta  
gaurav.gupta@mq.edu.au  
Contact via via dialogue utility (contact teaching staff)  
to be posted on iLearn

Tutor  
Emma Walker  
emma.walker@mq.edu.au

Tutor  
Joseph Hardman  
 joseph.hardman@mq.edu.au

Tutor  
Sophie Kaelin  
 sophie.kaelin@mq.edu.au

Tutor  
Neil Murphy  
 neil.murphy@mq.edu.au

Tutor  
Bradley Kenny  
 bradley.kenny@mq.edu.au

### Credit points
3

### Prerequisites
COMP115 or admission to (BActStud or BActStudSc or BAppFinBActStud or BActStudBProfPrac)

### Corequisites

### Co-badged status
Unit description
This unit studies programming as a systematic discipline and introduces more formal software design methods. Programming skills are extended to include elementary data structures and abstract data types. There is a strong emphasis on problem solving and algorithms, including aspects of correctness, complexity and computability.

Important Academic Dates
Information about important academic dates including deadlines for withdrawing from units are available at https://students.mq.edu.au/important-dates

Learning Outcomes
Apply enhanced problem solving skills to develop algorithms
Implement programs (from algorithms), showing an understanding of the underlying architecture of the computer
Adhere to standard software engineering practices (in particular documentation using Javadoc, testing using JUnit framework and debugging using Eclipse debugger)
Compare different methods available for the same problem in terms of efficiency and other criteria
Demonstrate foundational learning skills including active engagement in their learning process

General Assessment Information
Late Submission
No extensions will be granted without an approved application for Special Consideration. There will be a deduction of 10% of the total available marks made from the total awarded mark for each 24 hour period or part thereof that the submission is late. For example, 25 hours late in submission for an assignment worth 10 marks – 20% penalty or 2 marks deducted from the total. No submission will be accepted after solutions have been posted.

Assessment Tasks

<table>
<thead>
<tr>
<th>Name</th>
<th>Weighting</th>
<th>Hurdle</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programming contests</td>
<td>0%</td>
<td>No</td>
<td>Weeks 1 to 10</td>
</tr>
<tr>
<td>Practical Exam 1</td>
<td>5%</td>
<td>No</td>
<td>Week 4 practical class</td>
</tr>
<tr>
<td>Practical Exam 2</td>
<td>15%</td>
<td>No</td>
<td>Week 7 practical class</td>
</tr>
<tr>
<td>Practical Exam 3</td>
<td>30%</td>
<td>Yes</td>
<td>Week 13 practical class and week 14</td>
</tr>
<tr>
<td>Name</td>
<td>Weighting</td>
<td>Hurdle</td>
<td>Due</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------</td>
<td>--------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Assignment 1</td>
<td>5%</td>
<td>No</td>
<td>23:45, Friday ending week 6</td>
</tr>
<tr>
<td>Assignment 2</td>
<td>10%</td>
<td>No</td>
<td>23:45, Friday ending week 13</td>
</tr>
<tr>
<td>Final Examination</td>
<td>35%</td>
<td>No</td>
<td>TBA</td>
</tr>
<tr>
<td>Participation</td>
<td>0%</td>
<td>Yes</td>
<td>Week 1 to 12</td>
</tr>
</tbody>
</table>

Programming contests

**Due: Weeks 1 to 10**

Weighting: 0%

Programing tasks on [HackerRank](https://www.hackerrank.com) (released throughout the semester via iLearn announcements).

Students should create an account on HackerRank for this.

Then go to Account -> Settings and change username to **StudentID_S2_19**

For example, if your StudentID is 40404040, choose username **40404040_S2_19**

These provide excellent practice for the practical exams.

On successful completion you will be able to:

- Apply enhanced problem solving skills to develop algorithms
- Implement programs (from algorithms), showing an understanding of the underlying architecture of the computer
- Adhere to standard software engineering practices (in particular documentation using Javadoc, testing using JUnit framework and debugging using Eclipse debugger)
- Compare different methods available for the same problem in terms of efficiency and other criteria
- Demonstrate foundational learning skills including active engagement in their learning process

Practical Exam 1

**Due: Week 4 practical class**

Weighting: 5%

The first practical exam will assess you on problem solving (arrays, mostly numerical)

On successful completion you will be able to:

- Apply enhanced problem solving skills to develop algorithms
• Implement programs (from algorithms), showing an understanding of the underlying architecture of the computer
• Adhere to standard software engineering practices (in particular documentation using Javadoc, testing using JUnit framework and debugging using Eclipse debugger)
• Compare different methods available for the same problem in terms of efficiency and other criteria
• Demonstrate foundational learning skills including active engagement in their learning process

Practical Exam 2
Due: **Week 7 practical class**
Weighting: **15%**

The practical exam will assess students on topics from weeks 2 to 6.

On successful completion you will be able to:
• Apply enhanced problem solving skills to develop algorithms
• Implement programs (from algorithms), showing an understanding of the underlying architecture of the computer
• Adhere to standard software engineering practices (in particular documentation using Javadoc, testing using JUnit framework and debugging using Eclipse debugger)
• Compare different methods available for the same problem in terms of efficiency and other criteria
• Demonstrate foundational learning skills including active engagement in their learning process

Practical Exam 3
Due: **Week 13 practical class and week 14**
Weighting: **30%**

This is a hurdle assessment task (see assessment policy for more information on hurdle assessment tasks).

The practical exam will assess students on topics discussed in the lectures from week 2 to week 11 (and thereby in the practicals from week 3 to week 12). **This is a hurdle exam** (see Grading Standards section for further details). **You must get at least 40 out of 100 to clear this hurdle.**

Each student will have two attempts at the practical exam. The **best mark** counts towards the final grade.

**Attempt 1**: Week 13 practical class
Attempt 2: Week 14 (Students can pick one of the several sessions in which exam is offered)

On successful completion you will be able to:
   • Apply enhanced problem solving skills to develop algorithms
   • Implement programs (from algorithms), showing an understanding of the underlying architecture of the computer
   • Adhere to standard software engineering practices (in particular documentation using Javadoc, testing using JUnit framework and debugging using Eclipse debugger)
   • Compare different methods available for the same problem in terms of efficiency and other criteria
   • Demonstrate foundational learning skills including active engagement in their learning process

Assignment 1
Due: 23:45, Friday ending week 6
Weighting: 5%

This will be a programming assignment that will help you to practise concepts from weeks 1 - 3. You code will be assessed via automated tests that will be provided to you. You will also be marked on code quality and completeness.

On successful completion you will be able to:
   • Apply enhanced problem solving skills to develop algorithms
   • Implement programs (from algorithms), showing an understanding of the underlying architecture of the computer
   • Adhere to standard software engineering practices (in particular documentation using Javadoc, testing using JUnit framework and debugging using Eclipse debugger)
   • Compare different methods available for the same problem in terms of efficiency and other criteria
   • Demonstrate foundational learning skills including active engagement in their learning process

Assignment 2
Due: 23:45, Friday ending week 13
Weighting: 10%

This will be a programming assignment that will help you to practise concepts from weeks 4 - 9. Your code will be assessed via automated tests that will be provided to you.
On successful completion you will be able to:

- Apply enhanced problem solving skills to develop algorithms
- Implement programs (from algorithms), showing an understanding of the underlying architecture of the computer
- Adhere to standard software engineering practices (in particular documentation using Javadoc, testing using JUnit framework and debugging using Eclipse debugger)
- Compare different methods available for the same problem in terms of efficiency and other criteria
- Demonstrate foundational learning skills including active engagement in their learning process

Final Examination

Due: **TBA**
Weighting: **35%**

This will be a two hour written invigilated examination which will cover content from the entire semester.

If you receive **Special Consideration** for the final exam, a supplementary exam will be scheduled after the normal exam period, following the release of marks. By making a special consideration application for the final exam you are declaring yourself available for a resit during the supplementary examination period and will not be eligible for a second special consideration approval based on pre-existing commitments. Please ensure you are familiar with the policy prior to submitting an application. Approved applicants will receive an individual notification one week prior to the exam with the exact date and time of their supplementary examination.

On successful completion you will be able to:

- Apply enhanced problem solving skills to develop algorithms
- Implement programs (from algorithms), showing an understanding of the underlying architecture of the computer
- Adhere to standard software engineering practices (in particular documentation using Javadoc, testing using JUnit framework and debugging using Eclipse debugger)
- Compare different methods available for the same problem in terms of efficiency and other criteria
- Demonstrate foundational learning skills including active engagement in their learning process

Participation

Due: **Week 1 to 12**
Weighting: **0%**
This is a hurdle assessment task (see assessment policy for more information on hurdle assessment tasks)

You must attend and participate in at least 8 of the 10 practical classes in weeks 1-3, 5-6, and 8-12 to pass this unit. This is a hurdle requirement from the faculty of science and engineering. Participation involves actively interacting with the tutor and your peers. Your performance in a short iLearn quiz at the end of each practical class will also be considered to determine if you get participation mark for that week.

We have practical exams in weeks 4, 7 and 13 and therefore unable to satisfactorily assess student participation in those weeks.

On successful completion you will be able to:

- Apply enhanced problem solving skills to develop algorithms
- Implement programs (from algorithms), showing an understanding of the underlying architecture of the computer
- Adhere to standard software engineering practices (in particular documentation using Javadoc, testing using JUnit framework and debugging using Eclipse debugger)
- Compare different methods available for the same problem in terms of efficiency and other criteria
- Demonstrate foundational learning skills including active engagement in their learning process

**Delivery and Resources**

**CLASSES**

Each week you should attend

- three hours of lectures and
- two hour practical class - structured as one hour tutorial followed by one hour practical.

For details of days, times and rooms consult the [timetables webpage](https://unitguides.mq.edu.au/unit_offerings/103824/unit_guide/print).

Note that Lectures and Practical classes commence in week 1.

You should have selected a practical class during enrolment. **You should attend the practical class in which you are enrolled.** You won't always get the class of your choice. Check availabilities via eStudent regularly. If ALL practical classes are full, only then, contact the convenor.

Please note that you are **required** to submit work regularly. You will get the help that you need by attending your practical class. Failure to submit work may result in you failing the unit (see the precise requirements in the "Grading Standards" section) or being excluded from the final examination.
REQUIRED AND RECOMMENDED TEXTS AND/OR MATERIALS

Textbook

The first book in the following list, namely *Starting out with Java*, is the recommended text book for COMP125. The other books listed are helpful references.

  - Online edition of this book is available through MQ Library. There can be up to 5 simultaneous accesses.
  - Click on "Full text available at: 2018 eTextbooks" and login with OneID and password.
- **B. Eckel**, *Thinking in Java* (electronic book, 3rd edition available within iLearn is fine and is free but does not cover data structures)
- **A. Drozdek**, *Data Structures and Algorithms in Java* (Cengage) 2nd edition. ISBN 9780534449252 (this book will also be used in COMP225)
- **D. Carlson**, *Eclipse Distilled* (Addison-Wesley) 1st edition. ISBN 9780321288158 (extensive coverage of the software development platform eclipse)

TECHNOLOGY USED AND REQUIRED

Audio and Video Lecture

Digital recordings of lectures are available from within iLearn via Active Learning Platform.

Technology

- **Java SE** - download the latest Java SE to be compatible with the labs.
- **Eclipse** - download Eclipse IDE for Java Developers
- **Learning Management System iLearn**

Discussion Boards

The unit makes use of forums hosted within iLearn. Please post questions there, they are monitored by the unit staff.

Unit Schedule

Note that three important themes will pervade the entire unit:

1. Problem solving. A crucial skill for all of the weekly topics will be to write appropriate
code to meet a given problem specification. This theme relates to the first two learning outcomes for this unit.

2. Software development. Use of the JUnit testing framework is an important development practice which will be taught from the beginning, and used throughout the unit. This theme relates to the third learning outcome of this unit.

3. Comparing different solution methods. Very often different algorithms are available for the same problem. Another important skill to develop throughout this unit is the ability to compare different algorithms in terms of efficiency and other criteria. This theme relates to the fourth learning outcome of this unit.

<table>
<thead>
<tr>
<th>Week</th>
<th>Topic</th>
<th>Assessments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Java and Eclipse</td>
<td>Diagnostic test (does not contribute towards final mark)</td>
</tr>
<tr>
<td>2</td>
<td>Problem solving, JUnit testing</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Classes and Objects - 1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Classes and Objects - 2</td>
<td>Practical exam 1</td>
</tr>
<tr>
<td>5</td>
<td>Recursion - 1</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Binary search</td>
<td>Assignment 1 Due</td>
</tr>
<tr>
<td>7</td>
<td>Case study</td>
<td>Practical exam 2</td>
</tr>
<tr>
<td>8</td>
<td>List Interface</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Custom-built ArrayList</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Self-referencing classes (Node class)</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Custom-built LinkedList</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Recursion - 2</td>
<td>Assignment 2 Due</td>
</tr>
<tr>
<td>13</td>
<td>Revision</td>
<td>Practical exam 3</td>
</tr>
</tbody>
</table>

**Policies and Procedures**

Macquarie University policies and procedures are accessible from [Policy Central](https://staff.mq.edu.au/work/strategy-planning-and-governance/university-policies-and-procedures/policy-central). Students should be aware of the following policies in particular with regard to Learning and Teaching:

- Academic Appeals Policy
- Academic Integrity Policy
- Academic Progression Policy
- Assessment Policy
Undergraduate students seeking more policy resources can visit the Student Policy Gateway (https://students.mq.edu.au/support/study/student-policy-gateway). It is your one-stop-shop for the key policies you need to know about throughout your undergraduate student journey.

If you would like to see all the policies relevant to Learning and Teaching visit Policy Central (https://staff.mq.edu.au/work/strategy-planning-and-governance/university-policies-and-procedures/policy-central).

Student Code of Conduct

Macquarie University students have a responsibility to be familiar with the Student Code of Conduct: https://students.mq.edu.au/study/getting-started/student-conduct

Results

Results published on platform other than eStudent, (eg. iLearn, Coursera etc.) or released directly by your Unit Convenor, are not confirmed as they are subject to final approval by the University. Once approved, final results will be sent to your student email address and will be made available in eStudent. For more information visit ask.mq.edu.au or if you are a Global MBA student contact globalmba.support@mq.edu.au

Disruption to Studies

The University recognises that students may experience disruption that adversely affects their academic performance in assessment activities. Support services are provided by the University, and it is the student’s responsibility to access such services as appropriate. For information concerning the Disruption policy and procedures please visit http://students.mq.edu.au/student_admin/exams/disruption_to_studies/

Learning Skills

Learning Skills (mq.edu.au/learningskills) provides academic writing resources and study strategies to improve your marks and take control of your study.

- Workshops
- StudyWise
- Academic Integrity Module for Students
- Ask a Learning Adviser
Our graduates will take with them the intellectual development, depth and breadth of knowledge, scholarly understanding, and specific subject content in their chosen fields to make them competent and confident in their subject or profession. They will be able to demonstrate, where relevant, professional technical competence and meet professional standards. They will be able to articulate the structure of knowledge of their discipline, be able to adapt discipline-specific knowledge to novel situations, and be able to contribute from their discipline to inter-disciplinary solutions to problems.

This graduate capability is supported by:

**Learning outcomes**

- Apply enhanced problem solving skills to develop algorithms
- Implement programs (from algorithms), showing an understanding of the underlying architecture of the computer
- Adhere to standard software engineering practices (in particular documentation using Javadoc, testing using JUnit framework and debugging using Eclipse debugger)
- Compare different methods available for the same problem in terms of efficiency and other criteria
- Demonstrate foundational learning skills including active engagement in their learning process

**Assessment tasks**

- Programming contests
Problem Solving and Research Capability

Our graduates should be capable of researching; of analysing, and interpreting and assessing data and information in various forms; of drawing connections across fields of knowledge; and they should be able to relate their knowledge to complex situations at work or in the world, in order to diagnose and solve problems. We want them to have the confidence to take the initiative in doing so, within an awareness of their own limitations.

This graduate capability is supported by:

Learning outcomes

- Apply enhanced problem solving skills to develop algorithms
- Adhere to standard software engineering practices (in particular documentation using Javadoc, testing using JUnit framework and debugging using Eclipse debugger)
- Compare different methods available for the same problem in terms of efficiency and other criteria

Assessment tasks

- Programming contests
- Practical Exam 1
- Practical Exam 2
- Practical Exam 3
- Assignment 1
- Assignment 2
- Final Examination
- Participation

Creative and Innovative

Our graduates will also be capable of creative thinking and of creating knowledge. They will be imaginative and open to experience and capable of innovation at work and in the community. We want them to be engaged in applying their critical, creative thinking.

This graduate capability is supported by:
Assessment tasks

- Programming contests
- Practical Exam 1
- Practical Exam 2
- Practical Exam 3
- Assignment 1
- Assignment 2
- Final Examination
- Participation

Effective Communication

We want to develop in our students the ability to communicate and convey their views in forms effective with different audiences. We want our graduates to take with them the capability to read, listen, question, gather and evaluate information resources in a variety of formats, assess, write clearly, speak effectively, and to use visual communication and communication technologies as appropriate.

This graduate capability is supported by:

Learning outcome

- Demonstrate foundational learning skills including active engagement in their learning process

Assessment tasks

- Programming contests
- Final Examination
- Participation

Engaged and Ethical Local and Global citizens

As local citizens our graduates will be aware of indigenous perspectives and of the nation's historical context. They will be engaged with the challenges of contemporary society and with knowledge and ideas. We want our graduates to have respect for diversity, to be open-minded, sensitive to others and inclusive, and to be open to other cultures and perspectives: they should have a level of cultural literacy. Our graduates should be aware of disadvantage and social justice, and be willing to participate to help create a wiser and better society.

This graduate capability is supported by:

Learning outcome

- Demonstrate foundational learning skills including active engagement in their learning process
Assessment tasks

• Assignment 1
• Assignment 2
• Participation

Socially and Environmentally Active and Responsible

We want our graduates to be aware of and have respect for self and others; to be able to work with others as a leader and a team player; to have a sense of connectedness with others and country; and to have a sense of mutual obligation. Our graduates should be informed and active participants in moving society towards sustainability.

This graduate capability is supported by:

Assessment task

• Participation

Capable of Professional and Personal Judgement and Initiative

We want our graduates to have emotional intelligence and sound interpersonal skills and to demonstrate discernment and common sense in their professional and personal judgement. They will exercise initiative as needed. They will be capable of risk assessment, and be able to handle ambiguity and complexity, enabling them to be adaptable in diverse and changing environments.

This graduate capability is supported by:

Learning outcome

• Demonstrate foundational learning skills including active engagement in their learning process

Assessment tasks

• Final Examination
• Participation

Critical, Analytical and Integrative Thinking

We want our graduates to be capable of reasoning, questioning and analysing, and to integrate and synthesise learning and knowledge from a range of sources and environments; to be able to critique constraints, assumptions and limitations; to be able to think independently and systemically in relation to scholarly activity, in the workplace, and in the world. We want them to have a level of scientific and information technology literacy.

This graduate capability is supported by:
Learning outcomes

- Compare different methods available for the same problem in terms of efficiency and other criteria
- Demonstrate foundational learning skills including active engagement in their learning process

Assessment tasks

- Programming contests
- Practical Exam 1
- Practical Exam 3
- Assignment 1
- Assignment 2
- Final Examination
- Participation

Commitment to Continuous Learning

Our graduates will have enquiring minds and a literate curiosity which will lead them to pursue knowledge for its own sake. They will continue to pursue learning in their careers and as they participate in the world. They will be capable of reflecting on their experiences and relationships with others and the environment, learning from them, and growing - personally, professionally and socially.

This graduate capability is supported by:

Learning outcome

- Demonstrate foundational learning skills including active engagement in their learning process

Assessment tasks

- Final Examination
- Participation

Changes from Previous Offering

Changes since S1, 2019

1. Discussion on visibility (public vs. private) removed from COMP125. All instance members (variables and methods) are now assumed to be public. Visibility will be covered in COMP229.
2. Practical exam 3 moved from week 12 to week 13.
3. Threshold required to sit the second attempt removed.
4. Best of two attempts counts towards the final mark for practical exam 3.

Changes since S2, 2018

Faculty ruling that the second attempt marks should be capped to be enforced.

Changes since S1, 2018

1. Practical exam 3 weight increased from 25% to 30%
2. Final exam weight reduced from 40% to 35%
3. More programming activities added to provide students required practice.

Changes since S2, 2017

1. Total weight for practical exams increased from 35% to 45%.
2. Written tests are now in the workshops and do not contribute towards final marks.
3. Hurdles have been changed. Final exam is no longer the hurdles. Workshop participation and performing satisfactorily in practical exam 3 are hurdles.
4. Textbook changed from required to recommended.

Changes since S1, 2017

1. Number of practical exams increased from 2 to 3. Total weight for practical exams increased from 25% to 35%.
2. Final exam will be shortened to 2 hours and will be worth 40% (down from 50% in S1, 2017).
3. Use of codingbat.com for programming exercises.
4. Minor change in schedule of written test 5.
5. User-defined lists back in the assessed topics.

Changes since S2, 2016

None

Changes since S1, 2016

1. Weekly exercises replaced by 5 tests and marks decreased from 15 to 10, marks for end of semester practical exam increased from 10 to 15.
2. Introduction of the final exam as a hurdle assessment. Please refer to assessment tasks for further details.

Changes since S2, 2015

1. Change of textbook from "Absolute Java" to "Starting out with Java". Please refer to "Delivery and resources" for more details about the textbook.
2. Classes-first approach will be adopted this semester. We will try to familiarize students
with the object-oriented environment that is Java, and then move on to more complex problem solving.

3. Exceptions and queues have been removed from the course content.
4. In-class tests replaced by practical examinations.
5. Weekly submissions replaced by in-class oral assessments.
6. Quizzes introduced as a pre-requisite to assignments.
7. Number of assignments reduced to two.

Grading Standards

Three standards, namely Developing, Functional, and Proficient, summarize as many different levels of achievement. Each standard is precisely defined to help students know what kind of performance is expected to deserve a certain mark. The standards corresponding to the learning outcomes of this unit are given below:

| L.O. #1 | Limited ability to solve problems. Limited knowledge of basic data structures. | Ability to write simple algorithms and solve simple problems using OOD. Know basic data structures (queues, stacks, linked lists) and how to manipulate them. | Ability to write complex algorithms and solve complex problems using OOD and recursion. Ability to select the most appropriate data structures to solve a problem. |
| L.O. #2 | Show poor programming skills. Limited ability to write code that compiles or executes properly. | Show basic programming skills. Understand notions of compiler and virtual machine. Know types, how to implement simple conditions, simple loops, simple data structures, simple objects. | Show advanced programming skills. Understand notions of compiler and virtual machine. Know types, how to implement conditions, loops, data structures, objects. Understand inheritance and polymorphism. |
| L.O. #3 | Unability to follow specifications. Poor coding style. Poor documentation. Submission of incorrect programs showing no sign of testing/debugging skills. | Follow simple specifications. Document code (e.g. pre-post conditions). Test and debug a simple program. Understand the notion of modularity/object file. | Understand the importance of specifications. Neat code/consistent programming style. Clear and insightful comments. Design test cases and debug programs. |

At the end of the semester, you will receive a grade that reflects your achievement in the unit.

- **Fail (F):** does not provide evidence of attainment of all learning outcomes. There is missing or partial or superficial or faulty understanding and application of the fundamental concepts in the field of study; and incomplete, confusing or lacking communication of ideas in ways that give little attention to the conventions of the discipline.

- **Pass (P):** provides sufficient evidence of the achievement of learning outcomes. There is demonstration of understanding and application of fundamental concepts of the field of study; and communication of information and ideas adequately in terms of the conventions of the discipline. The learning attainment is considered satisfactory or adequate or competent or capable in relation to the specified outcomes.
In this unit, your final grade depends on your performance for each component of the assessment. Indeed, for each task, you receive a mark that captures your standard of performance regarding each learning outcome assessed by this task. Then the different component marks are added up to determine your total mark out of 100. Your grade then depends on this total mark and your overall standard of performance.

You will pass the unit, if

- your total mark is at least 50 out of 100; and,
- you get at least 40% in the third practical exam, and,
- you actively participate in at least 7 out of 9 workshops (weeks 1,2,3,5,6,8,9,10,11)

In order to obtain a higher grade than a Pass, you must fulfil the conditions for a Pass and have a total mark of at least

- 85 for High Distinction;
- 75 for Distinction;
- 65 for Credit.