COMP125

Fundamentals of Computer Science

S2 Day 2018

Dept of Computing

Contents

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.

General Information

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

Tutor
Scott Buckley
scott.buckley@mq.edu.au
Contact via via dialogue utility (contact teaching staff)

Tutor
Daniel Sutantyo
daniel.sutantyo@mq.edu.au
Contact via via dialogue utility (contact teaching staff)

Tutor
Michael Lay
michael.lay@mq.edu.au
Contact via via dialogue utility (contact teaching staff)

Tutor
Terry Bui
terry.bui@mq.edu.au
Contact via via dialogue utility (contact teaching staff)

Tutor
Julius Lu
jianchao.lu@mq.edu.au
Contact via via dialogue utility (contact teaching staff)

Lecturer
James Zheng
james.zheng@mq.edu.au
Contact via via dialogue utility (contact teaching staff)

Tutor
Kym Haines
kym.haines@mq.edu.au
Contact via via dialogue utility (contact teaching staff)

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://www.mq.edu.au/study/calendar-of-dates

Learning Outcomes

On successful completion of this unit, 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

General Assessment Information

Late Submission

Approval for late submissions will be done on a case by case basis. By default, late submissions are not accepted unless you experience unavoidable disruption, in which case, please submit a Disruption to Studies notification with appropriate evidence.

Assessment Tasks

Name	Weighting	Hurdle	Due
Programming contests	0%	No	Weeks 1 to 10
Practical Exam 1	5%	No	Week 4 workshops
Practical Exam 2	15%	No	Week 7 workshops
Practical Exam 3	30%	Yes	Week 12 workshops
Assignment 1	5%	No	23:45, Friday ending week 6
Assignment 2	10%	No	23:45, Friday ending week 12
Final Examination	35%	No	TBA
Participation in workshops	0%	Yes	Week 1 to 12

Programming contests

Due: **Weeks 1 to 10**
Weighting: **0%**

Programming tasks on HackerRank

Students should create an account on HackerRank for this.

Then go to Account -> Settings and change username to **StudentID_S2_18**

For example, if your StudentID is 40404040, choose username **40404040_S2_18**

These provide excellent practice for the practical exams!

Week Content 1
COMP115 content

2
Simple java programs

3
Classes, Objects

5
Classes Objects

6
Searching, Sorting

8
Recursion

9
ArrayList

10 LinkedList
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 workshops**
Weighting: **5%**

The first practical exam will assess you on two topics -

1. problem solving (numbers and arrays)

2. classes and objects - 1

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 workshops**
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 12 workshops**
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 from week 2 to week 11. This is a hurdle exam. In order to pass the unit, you must have,

1. at least 40% in this assessment (12 out of 30)
2. a raw mark of 50 or more
3. satisfactory participation (not just attendance) in at least 7 out of 9 workshops (described below)

If you get more than or equal to 9 and less than 12 marks in this exam, you will be given a second chance. This exam will be scheduled in week 13 Friday between 4pm and 8pm.

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 12**
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 in the interval between the regular exam period and the start of the next session. 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. You can check the supplementary exam information page on FSE101 in iLearn (bit.ly/FSESupp) for dates, and 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 in workshops

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 7 of the 9 workshops in weeks 1-3, 5-6, and 8-11 to pass this unit. This is a hurdle requirement.

We have practical exams in weeks 4, 7 and 12 and therefore unable to satisfactorily assess student participation in those weeks.

Participation involves actively interacting with the tutor and your peers.

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 workshop - structured as one hour tutorial followed by one hour practical.

For details of days, times and rooms consult the timetables webpage.

Note that Lectures and Workshops commence in week 1.

You should have selected a workshop during enrolment. **You should attend the workshop you are enrolled in**. If you do not have a class, or if you wish to change one, you should see the enrolment operators in the E7B courtyard during the first two weeks of the semester. Thereafter you should go to the Science and Engineering Student Services Centre.

Please note that you are **required** to submit work regularly. You will get the help that you need by attending your workshop. 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.

HELP101

A help desk where tutors are available for consultation on 100-level units.

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.

- **T. Gaddis, Starting out with Java: From control structures through**

objects (Pearson), Global edition (6th). ISBN 9781292110653

   - 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.

- W. Savitch, Absolute Java (Pearson) 5th edition. ISBN 9780132830317
- 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 9780534492526 (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 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
- http://hackerrank.com/ and http://codingbat.com/ for programming exercises.

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.

1	Welcome, refresher
2	Problem solving, debugging, JUnit testing
3	Classes and Objects - 1
4	Classes and Objects - 2
5	Binary search, selection sort, insertion sort
6	Case Study - 1
7	Recursion - 1
8	ArrayLists - 1
9	LinkedLists - 1
10	LinkedLists - 2
11	Recursion - 2
12	Case Study - 2
13	Revision

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
- Fitness to Practice Procedure
- Grade Appeal Policy
- Complaint Management Procedure for Students and Members of the Public
- Special Consideration Policy *(**Note:** The Special Consideration Policy is effective from 4 December 2017 and replaces the Disruption to Studies 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 shown in *iLearn*, 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.

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/

Student Support

Macquarie University provides a range of support services for students. For details, visit http://students.mq.edu.au/support/

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

Student Services and Support

Students with a disability are encouraged to contact the Disability Service who can provide appropriate help with any issues that arise during their studies.

Student Enquiries

For all student enquiries, visit Student Connect at ask.mq.edu.au

IT Help

For help with University computer systems and technology, visit http://www.mq.edu.au/about_us/offices_and_units/information_technology/help/.

When using the University's IT, you must adhere to the Acceptable Use of IT Resources Policy. The policy applies to all who connect to the MQ network including students.

Graduate Capabilities

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 in workshops

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 in workshops

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 in workshops

Discipline Specific Knowledge and Skills

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

- Practical Exam 1
- Practical Exam 2
- Practical Exam 3
- Assignment 1
- Assignment 2
- Final Examination
- Participation in workshops

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 in workshops

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 in workshops

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 in workshops

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 in workshops

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 in workshops

Changes from Previous Offering

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.

Chances 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 excutes 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.
- **Credit (Cr)**: provides evidence of learning that goes beyond replication of content knowledge or skills relevant to the learning outcomes. There is demonstration of substantial understanding of fundamental concepts in the field of study and the ability to apply these concepts in a variety of contexts; plus communication of ideas fluently and clearly in terms of the conventions of the discipline.
- **Distinction (D)**: provides evidence of integration and evaluation of critical ideas, principles and theories, distinctive insight and ability in applying relevant skills and concepts in relation to learning outcomes. There is demonstration of frequent originality in defining and analysing issues or problems and providing solutions; and the use of means of communication appropriate to the discipline and the audience.
- **High Distinction (HD)**: provides consistent evidence of deep and critical understanding in relation to the learning outcomes. There is substantial originality and insight in identifying, generating and communicating competing arguments, perspectives or problem solving approaches; critical evaluation of problems, their solutions and their implications; creativity in application.

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.

Concretely, **you will pass the unit**, if

- your total mark is at least 50 out of 100; and
- you get at least 40% (10 out of 25) in the third practical exam.
- 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.

Changes since First Published

Date	Description
30/07/2018	1. Added online textbook link that was communicated to me today by the library. 2. Corrected HackerRank username requirements under Programming contests. (HackerRank only allows 16 or less characters in username).
25/07/2018	minor fix of java version (changed from 9 to "latest").