INFO1103: Introduction to Programming

School of Information Technologies, University of Sydney

Week 1: First Steps

We will cover: Introduction to the unit, fundamental concepts, your first program

You should read: Sections 1.1 - 1.2, 2.1 of Savitch
Java: An Introduction to Problem Solving and Programming
7th Edition, Walter Savitch, ISBN 9781292018331

Lecture 1: Introduction to Programming

Course overview and general guidelines

INFO1103

- This unit is *Introduction to Programming*.
- You will, I hope, by the end of this Semester, be able to write simple programs in Java and understand the basics of programming in general.
- This is a standard unit (i.e., not Advanced) but there will be extension material to keep everyone challenged.
- You will be expected to come to lectures and take notes, ask questions and engage with the subject!
- We will do our best to teach you all — but *you* have to take responsibility for your own learning.

Where am I?

If you're wondering what you're doing in this unit, maybe something here
will help:

- This unit is the lead-in to ALL the technical units in the School of IT
- This unit *does* require you to program!
- This unit has *no prerequisites*[1]
- INFO1103 is not a simple version of INFO1903 – these two are
  completely independent.

- (or possibly. . .) Actually yes, you've walked in to the wrong room.

---

[1]Except you know, having finished High School reasonably well and such. But note: we
don't assume you can already program.

This unit

The unit is delivered in 13 weeks of lectures, with labs that you should attend. A register will be taken, and your attendance will be monitored.

Most of the *assessments* will be held during labs: missing them will not be a good idea as you'll miss out on the marks!

If you engage with the material, come to lectures, and *practice*, you will probably pass.☺

If you think you can learn it all by just reading the book and not programming, you will probably fail.☹

☠ *Plagiarism — submitting someone else's work as your own — will not be tolerated. You MUST read and understand the University's policy documents on* plagiarism*. We use electronic means to identify potentially plagiarised work. You have been warned.*

What You Will Learn

Learning to program takes *work*!

There are some tricky concepts to get your head around:

- How does my code turn into a program?
- How does information get moved around in the computer?
- How do I turn this problem into a set of instructions to solve it?
- Why is the computer so stupid / why doesn't it know what I mean?

Most of you are in the same boat. You will learn a *lot* this semester — particularly if you are new to programming. You can do it!

Lectures are recorded

Audio is recorded and the screen. The screen images are only those shown on the display that is to the presenters left hand side.

Don't just listen or simply attend, take good notes. The better your notes the easier it is to revise.

Some, but not all, solutions to lab exercises will be provided.

Lecture slides are not full in detail

Blackboard / eLearning

This semester Blackboard will be used for:

- Presenting your marks
- Access to Lecture recordings
- Web links to other important places

elearning.sydney.edu.au

We are using ed: edstem.com.au for discussions. Keep in mind:

When you post a question choose the most appropriate category.

Anyone posting code solutions to assessments will be banned, and may face further disciplinary action.

Any person doing anything inappropriate will face disciplinary action

The first lab will involve a sign up process for Ed - check with your tutor if you enrolled late.

Associate Professor Masahiro Takatsuka (Masa)
Information Visualisation, Computer Graphics, Computer Vision, Human Computer Interaction...

Teaches courses, supervises postgraduate research students, and conducts his own research work.

Please be considerate of his time.

The teaching assistant Mr. Tyson Thomas
Prepares and delivers tutorial materials and conducts general course duties.
Bachelor of Computer Science and Technology (Adv),
Strong interest in computational creativity and embedded systems

About our teaching team

We have a fantastic team of tutors. Each are talented in their own regard.

- Abdul Zreika
- Gladys Wong
- Jonathan Du
- Madeleine Wagner
- Chris Hyland
- David Byrne
- Rachel Dowavic

- Frank Zhu
- Joel Aquilina
- Hamish Ivison
- Joel Aquilina
- Olivia Nemes-Nemeth
- Shane Arora
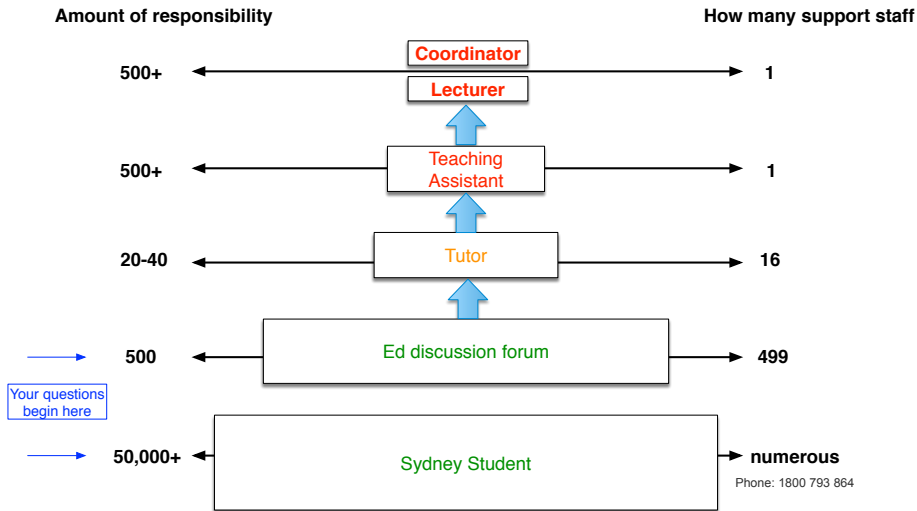- Tristan Spiteri
- Yixing Zheng

If you encounter any problems with our tutors, please contact the coordinator directly.

Where to get help?

1. Student admin → `https://sydneystudent.sydney.edu.au` or contact the Student Centre. Please check these first. E.g. timetable, passwords, payments, enrolments etc.
2. Ed discussion forum → `edstem.com.au`
3. Your tutor in your designated laboratory
4. The teaching assistant Tyson Tyson Thomas
5. course e-mail: for succinct messages only sit.info1103@sydney.edu.au
6. Consultation with Lecturer/Coordinator time/place:
   In between lectures!
   Room 212 in School of IT. 3pm Monday.

Do you need Help?

Amount of responsibility   **How many support staff**

500+ ← [**Coordinator**] [**Lecturer**] → **1**

500+ ← Teaching Assistant → **1**

20-40 ← Tutor → **16**

500 ← Ed discussion forum → **499**

Your questions begin here

50,000+ ← Sydney Student → **numerous**

Phone: 1800 793 864

Points That Sometimes Need Clarification

We are not out to get you: We want you to pass, but *you* have to do the work. If you feel that something has gone wrong, it might have, but *most things are fixable*.

"It's not fair": *This is as fair as you will ever get, ever again.* Everyone gets the same information and has access to the same resources. Everyone has the same requirements.

Reminders: At University you won't be reminded of when things are due: you have to organise your time and plan accordingly.

Assignments take time: The amount of time is not proportional to how many marks you can get from it, because assignments and quizzes and exams serve very different purposes.

"Introductory" does NOT mean "easy": No, it really doesn't. (Would you think "introductory quantum mechanics" was going to be simple?)

Points That Sometimes Need Clarification (cont.)

Attendance is not enough. We have had students, after they get their exam results, saying they deserved to pass, because they

1. tried really hard
2. went to every lecture
3. really need to pass

None of these are sufficient.

You are responsible. Now you're at University, you need to understand that, while we do care, and we do want you to do well, it's not just up to us. Our role is to help you to learn, to present you with all the resources you need, or the opportunities to get them: there's no spoon-feeding.

Points That Sometimes Need Clarification (cont.)

Language. If you're having trouble understanding written or spoken instructions, get help as soon as you can. Student Services has many resources to help you improve your understanding of English, organise your time, study, write reports, etc. Use them!

Time. You're probably enrolled full time, doing four 6cp units, or equivalent. Expect to spend about 10 hours per week for each 6cp unit.

Seriously, commit yourself to do 10 hours of *actual study*.

Assignments and Tasks are *individual* work.

You must get ≥40% of each major component to be permitted a pass: the major components are the *progressive mark* (Tasks, Quizzes, and Assignments combined), and the *exam mark*.

You must also get a combined mark of at least 50% in total, of course[2].

Here are some examples of how this works, in case the above isn't clear:

ProgMark 44%, Exam Mark 50%, total 48%: FAIL
ProgMark 75%, Exam Mark 35%, total 55%: FAIL
ProgMark 22%, Exam Mark 80%, total 51%: BUT FAIL

---

[2] If you are having trouble working this out, you are already in trouble.

Contacting staff

You should *definitely*

- post questions on ed if you're having problems,
- contact your tutor if you'd like more challenging things to do
- be really specific in your questions
- for serious/complex course issues, first email the coordinator and visit the consultation time.

You should *not*

- expect an immediate response
- expect all questions on ed to be answered by teaching staff
- post this question → "`my thing doesn't work`"
- expect to pass just by showing up.
- beg for more marks at any time.
- be late on the day of an assessment. **You won't be able to take it later**
- expect help on *beginning* of the course at the end
- writing this email → "`hi im in ur class can i get sum help`"

About your tutors

- Your tutors have been hand-picked and quality selected to give you the best possible help in learning to program.

- You *must* take advantage of having them around.

- Make sure you learn your tutor's name.

About you

- You're enrolled in INFO1103 (I hope)

- or you're just interested in programming

- You have probably not done (much) programming before

- You are highly motivated!

When you want help

1. Look at your notes and course text referred to as Savitch.
2. Look on ed *before* asking the same question someone else has asked! ☺
3. Ask your friends (don't copy their code! you won't LEARN!)
4. Ask your tutor directly
5. The consultation time is a good chance to get a direct response on administration matters. Remember, this is not a tutorial, you cannot receive help for your assignments or the course materials explained in great detail.

Assessments

What you'll have to do to do well

Important Dates

- Week 1-13 one or more challenges due
- Week 5 write a short Java program (45 minutes)
- Week 7 write a short Java program (45 minutes)
- Week 8 Quiz 1 (50 minutes)
- Week 10 write a short Java program (45 minutes)
- Week 12 Quiz 2 (50 minutes)

Quizzes 20%

- Week 8 worth **20%**
- Week 12 worth **0%**

These will assess your knowledge and skills for the course.

Closed book. 40-50 minutes time.

Covers all material from lectures and lab

You may write actual programming code, be neat.

These are on scheduled on Mondays of specific weeks in the semester and are visible on your timetable.
Absence or Lateness on the day will result in zero marks.[3]

---

[3] exception is Special Consideration

Practical assessment 0%

The practical assessment in this course is similar to an assignment. You will be given a problem to solve, allowed 40 - 50 minutes to work on the solution.
Complete this task with a lab computer.

These are on scheduled on Mondays of specific weeks in the semester and are visible on your timetable.

There is no Internet to be used during this assessment.

Practical assessment 0%

- Week 5 worth **0%**
- Week 7 worth **0%**
- Week 10 worth **0%**

This shows us that you know how to use a computer, but more importantly complete a programming problem given a specification.

Each assessment builds on the previous knowledge. It is important that you stay on top of the course!

Absence or Lateness on the day will result in zero marks.[4]

---

[4]exception is Special Consideration

Challenges

Small programming exercises to be completed online with edstem (about 16)

There will be both easy and hard questions. Most will take 5 - 30 minutes to complete.

Due dates are throughout the semester

No late submissions

Why so many assessments?

It is a reflection of how involved you are in the course.

Your study time, used correctly, will make these relatively easy. This is reasonable assumption of any University Student.

Most of your preparation time is practice tasks and tutorial exercises with *actual programming*

Final exam.

- The exam will be 2.5 hours long with 10 minutes reading time
- It will contribute 60% to your final grade.
- You will be permitted 1 A4-size sheet of paper with notes on it (yes, both sides)[5].

---

[5] unless you have Möbius paper but that's your own fault if you do

Automatic Marking

We use several ways to test your program. Generally, input vs output.

We give your program input data and compare the output with what we expect.

The *kind* of testing that is done by automatic marking is made known to you. Not everything is known to you.

Pay attention to the assignment specification to derive your own test data before you submit.

Automatic Marking (cont.)

For example, suppose you have a program called `MyNameIs`, which should print out a friendly message when run with the input "Bruce", and a question otherwise:

```
> java MyNameIs Bruce
Gday mate.
> java MyNameIs Bill
Is your name not Bruce?
```

If, when we run your program with the same input, we get this, then you would fail a test:

```
> java MyNameIs Bruce
Like, dude!
```

The above output would *fail* the test.
Similarly this would also fail:

```
> java MyNameIs Bruce
Is your name not Bruce?
```

Automatic Marking (cont.)

We have many tests for each program you write. To get full marks you must pass each test.

We also keep tests *hidden*: you won't know until after the deadline how you've gone on those.

That means you'll have to really understand your code and think carefully about all kinds of possible inputs to ensure your program will handle them properly.

More details on submission

- The program *must be written in Java* and must compile and run on the lab computers or other platform that is specified in the assignment
- The proportion of tests your code passes will be used for your automatic mark for each assessment.
- You may be required to *explain* your code to your tutor, or to the Unit Coordinator (i.e., me). If you can't explain what it does, you won't get a mark.[6]
- Don't get other people to do your assignments for you. Really.

---

[6] If you can't understand it, why would you submit it?

Plagiarism

☠ ***We run tests on your submissions to see how similar they are to those of other students.***

The software we use is very good at helping us detect plagiarism, so don't do it.

☠ ***We have failed students the course on the basis of plagiarism.***

Sharing is caring...is learning

Outside assessments we want you to pursue the discussion of programming problems, ideas, approaches, suggestions with your peers

If you can discuss the problem and receive feedback, then debug and correct your own code, you would have the equivalent of mining skill +1

If you can resolve someone else's programming problem without rewriting their code, you would have repair skill +1

There are many different kinds of programming skills!

Recommendation: survey as much as you can of this landscape and shift your character from explorer to navigator!

Times and places

INFO1103 lectures weeks 1-13.

- 1 - 3pm Monday ABS Lecture Theatre B2010

INFO1103 Tutorials weeks 1-13.

- Shown on your timetable. One 2 hour session.

INFO1103 practical assessments. See your timetable.

Sydney University Information Technology Society (SUITS)

This is not affiliated with this course.

They run various events throughout semester, both fun and professional.

Free pizza

Meet people

...Free pizza

They have a shiny website: https://suits.org.au

Welcome to the beginning!