Introduction to Computer Program Design

COMP 102 2020 T1

Sharon Gao, Peter Andreae (“Pondy”), David Streader

Computer Science

Victoria University of Wellington

Note: If 9am is full, lecture repeated at 4-5, KK LT301

COMP 102

Menu:

- Welcome
- Introductions
- What is COMP102 about?
- Where does COMP102 fit in your degree?
- Course organisation
- What to do NOW!

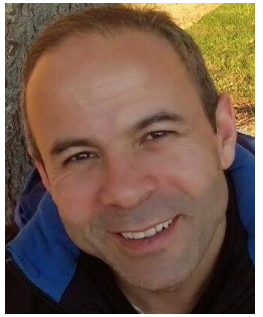
Admin:

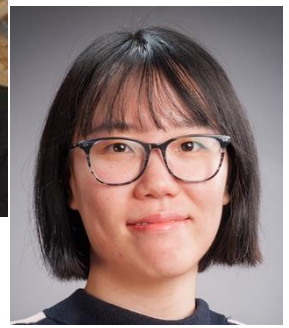
- Course Outline

The COMP 102 Team

Organiser	Xiaoying Gao (Sharon)	Office: CO 339	xgao@ecs.vuw.ac.nz
(Overall course management)		Office hours:	
Lecturer	Peter Andreae (Pondy)	Office: CO 336	pondy@ecs.vuw.ac.nz
		Office hours: Mon, Thu 1:15-2:00	
112 Lecturer	David Streader	Office: CO 260	dtsr@ecs.vuw.ac.nz
Tutor Admin	Dr. Ghassem Narimani	Office: CO 252	
(Admin issues regarding labs)			
Programmers	Dr. Monique Damito	email to report problems:	bugs@ecs.vuw.ac.n
	Betty Bai		
Tutors	Range of Undergraduates and Graduates		
School Office	(for forgotten passwords)	CO 358	
Students	You and the people around you		





What is the course about?

- COMP 102 is about learning the language and the ways of thinking required for building the software that underlies our digital world.

- Building software means writing programs: writing the instructions to make a computer behave in the way we want it to.

- In COMP102, you will design and write lots of little programs for lots of tasks.

- Give you a new set of mental tools for addressing problems
  - Different way of thinking from most disciplines
  - Creative,
  - Very precise,
  - Dealing with abstraction and complexity,.

Essential Info: Lectures

- Lectures:
  - Tues, We, Fri 9-10 (sorry – no choice)
  - First few weeks if needed:
    repeated in KKLT301:
    - Tue 4-5, Wed 2-4, Fri 1-2

	Mon	Tue	Wed	**Thu**	Fri
9		LECTURE	LECTURE		LECTURE
10					
11					
12					
1					Repeat L.
2			Repeat L.		
3					
4		Repeat L.			
5					

- COMP 112 lectures at same time as COMP 102.
  - easy to switch to those lectures if you choose

- Information:
  - ecs.victoria.ac.nz/Courses/COMP102_2020T1, also accessible via Blackboard

COMP102 vs COMP112:

Two courses taught together as different streams within a combined course.

- Both courses lead to COMP 103
- Same assessment
- Easy to switch between the streams –
  - just choose the lectures that work best for you,

COMP 102:

- Designed for students who have not done any programming
- Three lectures/week on programming in Java
- Goes slower, designed for beginning programmers
- If you have done some programming, that will obviously help

COMP 112:

- Designed for students who have done NCEA level 3 programming standards, or equivalent.
- Two lectures/week on programming in Java
- One lecture/week range of topics in Computer Science (not on the test/exam)
- Goes faster, covers wider range of issues.

Essential Info: Labs and Assignments

- Labs: Two 1hr Labs per week:
  - Pick one TL lab (Mon/Tues)
  - Pick one AL lab (Thu/Fri)

- First lab starts next Monday!
  - (Learning to use the lab systems)

- First assignment and "real" labs:
  - Thu/Fri 2nd week + Mon/Wed 3rd week

	Mon	Tue	Wed	**Thu**	Fri
9		**LECT**	**LECT**		**LECT**
10				Assig Due	
11	Lab TL1		Lab TL5	Lab AL1	Lab AL4
12	Lab TL2		Lab TL6	Lab AL2	Lab AL5
1	Lab TL3		Lab TL7	Lab AL3	Lab AL6
2	Lab TL4				Lab AL7
3					
4					
5					

Labs in CO 219/238 (next to each other)

Sign for labs at https://www.wgtn.ac.nz/students/study/timetables/tutorial-sign-up

What kind of course is it?

- About designing and building software.
- Not about *using* computers and applications software.
- Not an “easy credits” course for most people
  - Involves higher level thinking skills than many students expect
- Women students do as well or better than men students.
- Key factors for success are
  - problem solving, not memory, not guessing
  - logical/abstract thinking,
  - attention to detail
  - being able to think about your own thinking processes
  - not getting behind
- Takes time! plan on around 10 hours / week
- Practical work is critical

Background needed for COMP 102

- We assume you have ***used*** a computer
- We do ***NOT*** assume you have done any programming
  - If you haven't, This course is for you!
  - don't worry about, or be intimidated by those who have!
- But some students have!
  - good – it is definitely helpful
- We try to meet the needs of the full range of students
  - Variety of different help and support available
  - Assignments have graduated components.
- If you are repeating the course:
  - Do the whole of the assignments, without looking at previous solutions
  - The course will be similar, but there will be changes.

Lecture #2

- Introductions
- Voting for a Class Rep
  - Put a message about yourself on the forum if you want to be class representative; the class will vote on Monday.
- Planning your courses
- Course information

Planning Ahead:

- If you are doing BE or BSc (COMP) or BSc (CGRA) or BSc (DATA)
  - then you should plan on taking COMP103 in Tri 2.

- If you are doing BE, or BSc (COMP or CGRA)
  - Don't forget the maths courses that you need for 2nd year!
  - Consider CYBR 171 this trimester

- If you are doing BSc (CGRA)
  - Need CGRA 151 in Tri 2
  - Don't forget ANFX 101

- If you are doing BBmedSc or DATA
  - COMP 132 may be more relevant.

Planning Ahead: Mathematics

	Engineering maths		Mathematics maths
• BE SWEN/CYBR:	ENGR 121, 123	*or*	MATH 161, STAT 193/MATH 177
• BE ECEN:	ENGR 121, 122	or	MATH 151, 142
• BSc COMP:	ENGR 121, 123	or	MATH 161, STAT 193/MATH 177
• BSc CGRA:	ENGR 121, 123, [122]	or	MATH 151 , 161, [142]

Which should you take?

Planning Ahead: Mathematics

Which should you take?

- Most students are better off with the Engineering maths option.
  - slower start
  - focused on application of mathematics
- Students with good mathematics should consider the Mathematics maths option:
  - Opens more options in later years
  - Better background for postgraduate study, especially in computer graphics
  - If you have the following NCEA achievement standards:
    - 3.6 (differentiation, AS91578) and 3.7 (integration, AS91579)
    - one of 3.5 (complex nos, AS91577) or 3.1 (conics, AS91573) or 3.3 (trigonometry, AS91575) or 3.13 (probability, AS91585) or 3.14 (probability distributions, AS91586)).
    - At least 2 standards must be with grades of merit or excellence.
- If you want to switch …

Course Information

Bookmark http://ecs.wgtn.ac.nz/Courses/COMP102_2020T1
(also accessible via link on BlackBoard)

- Course Outline and Course Information,
- Announcements and lecture Slides
- Lab Assignment details (times, dates, handouts, files, submission, marks...)
- Forum, for questions and discussion
- Info about doing work at home.
- Java documentation
- Other useful links

Primary administrative communication channel.

Lab assignments

- Ten assignments (roughly weekly),
  - hand out: Thursday
  - due: 10am Thursday (a week later)
- Apply material from lectures and text book to practical programming problems.

  This is where your learning happens!
- Done partly in scheduled lab sessions
  - First session: Thurs/Fri (exercises and getting started on the assignment)
  - Second session: Mon/Wed (working on assignment with tutor support)
- Further work required: expect at least 5 hours outside labs
  - any of the ECS labs,
  - on your home computer

Signing up for labs

- Sign up for the labs:
  https://www.wgtn.ac.nz/students/study/timetables/tutorial-sign-up
  - choose ONE Thu/Fri Lab and ONE Mon/Wed Lab
  - Note: You need to be registered for the course
    (a) to sign up for a lab
    (b) to be able to use the school computers
  - what happens if they are full?

Where to go for Help

Depends on the kind of help needed

- Course organiser / Lecturer, Senior Tutor, tutors (in labs or helpdesk only!)
- Forum (via website)
- On-line help system (via website)
- Help desk (CO 242a)
- ECS School Office: CO 358
- Student Services: www.victoria.ac.nz/students/support
- Science Faculty office: www.victoria.ac.nz/science/student-administration
- www.wgtn.ac.nz/maori-hub/ especially Shaq in Cotton 133
- www.wgtn.ac.nz/pasifika/ especially Teneya in Cotton 133
- The Web

Getting Help.

Help Desk

- Online help:
  - Forum for general questions [don't post your assignment code!!!]
  - Email/web form for questions about your code. [only visible by staff and tutors]
- Help Desk: Tutors available at various times at CO242a: see weekly timetable, starting Tue in 3rd week.

Study groups

- We will facilitate organising study groups and tutored help sessions
- First year Engineering/CompSci tutorials/help sessions
- Excellent way of helping your learning
- Awhina and Pasifika programmes:
  - We will be working with Awhina and Pasifika support teams to help you.
- Women students support group.

Getting Help

- Ask the tutors in your lab session
- Ask a classmate, or someone else you know.
- Go to the workshops, Mondays/Wednesdays 5-7, AM106
- Go to the help desks.

Text Book and Handouts

Text Book

- *Java Foundations* Lewis, DePasquale, Chase
  - Same as for COMP103.
  - [also OK: *Java Software Solutions* (6th ed) Lewis and Loftus]
- We consider it an important resource for some people.
- The lectures complement the text, not replace it.
- Lectures will not cover all the details you need!
  - But nor will the textbook!

Resources

- Lecture slides & Assignments: On COMP102 web page.

Tests and Exams

Terms Test 1:

- 15%
- Thursday 2 April 5-6pm

Terms Test 2:

- 15%
- Tuesday 12 May 5-6 pm

Exam:

- 50%
- Date tba (between 12 and 27 June)

Note:

If a test mark is less than your exam mark, we will raise the test mark to the exam mark.

Assessment

To pass the course, you must:

- Satisfy the Mandatory Requirement.
- Get overall grade of **C-** or better.

Mandatory Course Requirement:

- Submit reasonable attempts (at least D) for at least 8 of 10 assignments.

Final Grade:

- Assignments: 20%
- Terms Test 1: 15% (mark boosted to exam mark, if better)
  Terms Test 2: 15% (mark boosted to exam mark, if better)
  Exam: 50%

Penalties for late assignments:

- 0 marks for late assignments, (Model solutions will be available)
- But you have a total of 24 "late hours" that you can use to avoid penalties.

Withdrawal dates

- Early withdrawal with refund: up do Fri 13 March
  - no consequences to early withdrawal

- Standard withdrawal without refund: up to Friday 15 May
  - Withdrawal recorded
  - No grade recorded on transcript
  - BUT, withdrawal counts as a fail for determining "Satisfactory Academic Progress"

- Late withdrawal with Associate Dean's permission: after 15 May
  - Requires permission of Associate Dean
  - Normally given when special circumstances arise after deadline.

Plagiarism (Cheating)

- You must not present anybody else's work as if it were your own work:
  - Basic principle of academic honesty.
  - applies to work by other students, friends, relatives, the web, books…
  - If you received substantial help, then you must state who helped and how much.
  - If you declare any work from someone else, then it isn't plagiarism!!!

- **In COMP102**:
  - We encourage you to work in pairs on the core & completion parts of assignments BUT
  - You **must** put a comment at the top of your code saying that you worked with ….
  - If you use code from the *assigned text book*, or
    from the *lectures*, then you do **not** need to declare it;
    If you use any other code that wasn't yours, then declare it!

Cheating in the assignments.

Assignments are primarily for learning, not assessing

Cheating in the assignments is not worth it!

- You won't learn, so you will probably fail.
- If caught, you'll lose marks --- or worse.
- Assignments have a fairly small contribution to your grade.

Learning to Program in Java

What's involved?

- Understand what the computer can do and what the language can specify
- Problem solving:
  - program design,
  - data structuring,
- Programming language (Java):
  - syntax and semantics
  - style and common patterns
  - libraries of code written by other people
- Testing and Debugging (fixing).
- Common patterns in program design.
  - Important data structures and algorithms.

Lect #3

- Labs
- Designing a Java program

- Announcements:
- Can’t find a lab slot that works?
- Class rep: vote on Monday.
  - Put/Read candidate statements on the forum
- Lost property from labs: ID cards, USB sticks → school office
- General Lost Property → Student Union office

Lab Facilities

- All scheduled labs are in CO219/238
- Can also use other ECS labs (or other university student computing labs)
- Can also use home computers. (Details on Web Site)
- Lab Hours: 24/7
  - Need ID card to access in evenings and weekends
- The labs are for getting work done
  - Don't prevent other people from working
  - If you want to play around, go somewhere else
- We expect professional behaviour in the labs.

Read the lab rules!

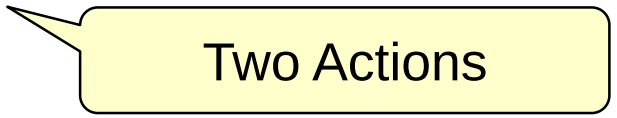
What is a Program

A program is a specification for the behaviour of a computer:

- What the computer should do when:
  - the program is started
  - the user types something
  - the user clicks the mouse
  - a message arrives over the network
  - some input from a camera/switch/sensor arrives.
  - ……
- Responses may be simple or very complex.
- A program consists of
  - descriptions of responses to events/requests
  - written as instructions
  - in a language the computer can understand:
    - Lots of different programming languages; we will use Java

A first Java Program

- Task: Write a temperature conversion program
- Step 1: Specification: what is it supposed to do?
  - Write a program that will let the user do two things:
    - print out the conversion formula
    - let user enter temperature in Fahrenheit, and print out in Celsius.
- Step 2: Design:
  - For print action:
    - Print the formula on the window
  - For calculate action:
    - Ask user for the Fahrenheit value to be converted
    - Calculate Celsius value out of given value: (F-32.0)*5.0/9.0
    - Print out the answer



Designing the Java program

Step 3: Editing

- Need to write this design in the Java language, using an editor (BlueJ)

  ➔ Need an ***object*** : a "temperature calculator"
  - all actions must be performed on some object

  ➔ Need a ***class*** to describe the object

  ➔ The class needs a name

  ➔ The class needs to specify the two actions its objects can do

  ➔ Define ***methods*** to do things.

  ➔ Give names to the methods

  ➔ specify what the methods will do

Compiling and Running

Step 4: Compiling

- If there are syntax errors (invalid Java)
  then the compiler will complain and list all the errors
  - ⇒ read the error message to work out what's wrong
  - ⇒ fixing syntax errors until it compiles without complaint
    - BlueJ makes this process easier

Step 5: Running and Testing

- Must run the program and test it on lots of different input.
  - BlueJ makes it easy to run individual methods.

Writing the Java code

```
import ecs100.*;

/** Program for converting between temperature scales */
public class TemperatureCalculator{

    /** Print conversion formula */
    public void printFormula ( ) {
       UI.println("Celsius = (Fahrenheit - 32) *5/9");
    }

    /** Ask for Fahrenheit and convert to Celsius */
    public void doFahrenheitToCelsius(){
        double fahrenheit = UI.askDouble("Farenheit:");
        double celsius = (fahrenheit - 32.0) * 5.0 / 9.0;
        UI.println(fahrenheit +  " F -> " + celsius + " C");
    }
}
```

Comments

Keywords

Identifiers

Strings

Types

Numbers

Operators

Punctuation

Improving the Program

```
import ecs100.*;                    /** Program for converting between temperature scales */
public class TemperatureCalculator{
    /** Print conversion formula */
    public void printFormula ( ) {
       UI.println("Celsius = (Fahrenheit - 32) *5/9");
    }
    /** Ask for Fahrenheit and convert to Celsius */
    public void doFahrenheitToCelsius(){
        double fahrenheit = UI.askDouble("Farenheit:");
        double celsius = (fahrenheit - 32.0) * 5.0 / 9.0;
        UI.println(fahrenheit +  " F -> " + celsius + " C");
    }
    /** Setup the buttons */
    public void setupGUI(){
        UI.addButton("Formula", this::printFormula);
        UI.addButton("Convert", this:: doFahrenheitToCelsius);
    }
}
```

BlueJ

- BlueJ is an IDE for Java
  (Integrated Development Environment)
  - Class manager, for keeping track of the files in your program
  - Editor for entering and modifying the program
  - Built-in compiler interface to help compile and fix the syntax errors
  - Special interface to make it easy to construct objects and call methods on them.

- Let's do it… editing in BlueJ

Elements of the program

Program Structure:

- Import
  - list the "libraries" you will use (We always use ecs100, and usually java.awt.Color and java.util.*)
- Class
  - Top level component of program
  - Describes a class of objects
    - Specifies the set of actions this kind of object can perform
    - (Can also specify information the objects can hold)
  - Note name, and conventions for naming.

- Methods
  - Main elements of a class
  - Each method describes an action that objects of this class can perform

Elements of the program

- Comments *vs* Code
- Keywords / Identifiers / Strings / Types / numbers / operators and punctuation
  - **Keywords** : words with special meaning in the Java Language
    eg: **public**, **class**, **if**, **while**, …
    mostly to do with the structure of the program
  - **Identifiers** : other words, used to refer to things in the program.
    mostly made up by the programmer,
    some are predefined.
  - Strings : bits of text that the program will manipulate.
    always surrounded by " and "
  - Types : names for kinds of values.
  - **numbers**
  - **operators** and **punctuation** : + - * / = % . ; , ( ) { } [ ] ' "
    all have precise meanings and rules for use

Actions in a program

- Method calls *object* . *method* (*arguments*)
  - telling an object to do one of its methods, passing the necessary information as arguments:

```
UI.println("Celsius = (Fahrenheit - 32) *5/9");
UI.drawRect(100, 200, 50, 75);
UI.addButton(“Draw",  this::doDraw);
```

  - What are the possible objects? what are the possible methods.
    - **UI** object has methods for
      - Printing, asking, drawing, buttons, ….
    - **this** object – the one we are defining – has the methods being defined in the class
- Assignment statements *place* = *value*
  - putting a value in a place

```
double celsius = (fahren – 32.0) * 5.0 / 9.0;
double fahren= UI.askDouble(“Fahrenheit:");
```

Using BlueJ for Java Programs

Simple use of BlueJ for simple programs:

1. Edit the class file(s) to define the methods
2. Compile the class (and fix the syntax errors)
3. Create an object of the class
   - right click on the rectangle representing the class
   - select “new…..”

     ⇒ a red square representing the object
4. Call methods on the object
   - right click on the square representing the object
   - select the method.

Compiling

Compiling

- Translating the Java programme into a simpler language the computer can run directly.
- If there are syntax errors (invalid Java)
  then the compiler will complain and list all the errors
  - ⇒ read the error message to work out what's wrong
  - ⇒ fix syntax errors until it compiles without complaint
    - BlueJ makes this process easier

Let’s do it…