

COMPI400

Programming for Designers

INFSI609/2609

Business Programming

Claude Sammut

Contacts

Lecturer in Charge: Claude Sammut

Web pages: <http://blogs.unsw.edu.au/infsl609/>

Web pages: <http://blogs.unsw.edu.au/compl400/>

Email: isl609@cse.unsw.edu.au

Email: cs1400@cse.unsw.edu.au

Subject admin: Tim Wiley

Tutor: Tim Wiley, Sim Mautner, Troy Lenger

Two classes in one

There are two classes:

- COMPI400
- INFSI609/2609

Class announcements

Announcements will be made in lectures and posted on the [class web pages](#).

It is your responsibility to keep up to date with announcements.

Labs

Programming can only be **learned by doing**.

Lab exercises will be posted on the web pages for each subject.

Labs: **Leaf Lab Mech Eng Undercroft (K-J17-G04)**

Labs start in **Week 2**.

Java and BlueJ

This course will be taught using the **Java** programming language and the **BlueJ** interactive development environment.

Java and BlueJ can be downloaded for free at: <http://www.bluej.org/>

Assessment

- Lab mark (10%)
- Three assignment tasks (10% each)
- Mid-semester practical exam (10%)
- A practical exam in Week 13 (20%)
- A written exam (30%)

Textbook

**Objects First with Java:
A Practical Introduction using BlueJ**

David J. Barnes & Michael Kölling

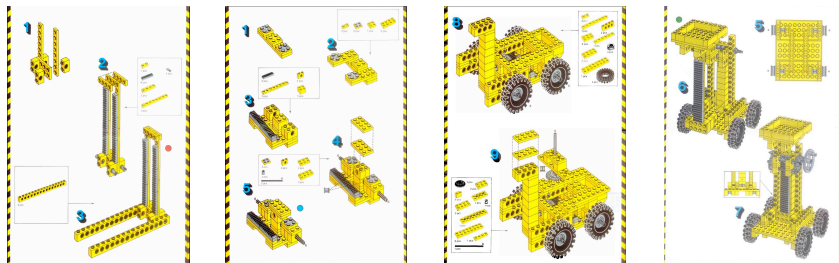
Pearson, 5th edition 2012

<http://www.bluej.org/objects-first>

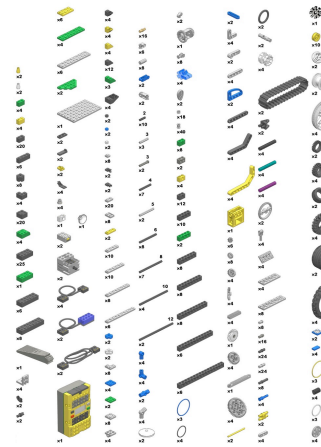
What is a computer?

- It is an incomplete machine
 - computer hardware only provides the building blocks for a working machine
 - A *program* assembles the blocks and says how they will be used and in what order

A program is a sequence of instructions that make a machine



What's the difference?



What does this do?

```
private static int min(int[] nums)
{
    int m = nums[0];
    for (int i = 1; i < nums.length; i++)
    {
        int n = nums[i];
        if (n < m)
            m = n;
    }
    return m;
}
```

What does this say?

ΣΩΚΡΑΤΗΣ· κατέβην χθὲς εἰς Πειραιᾶ μετὰ
Γλαύκωνος τοῦ Ἀρίστωνος προσευξόμενός τε τῇ θεῷ
καὶ ἅμα τὴν ἐορτὴν βουλόμενος θεάσασθαι τίνα
τρόπον ποιήσουσιν ἅτε νῦν πρῶτον ἄγοντες. καλὴ μὲν
μέντοι ἦττον ἐφαίνετο πρέπειν ἢν οἱ Θραῖκες ἔπεμπον.
προσευξάμενοι δὲ καὶ θεωρήσαντες ἀπήμεν πρὸς τὸ
ἄστυ. κατιδὼν οὖν πόρρωθεν ἡμᾶς οἴκαδε ὠρμημένους
Πολέμαρχος ὁ Κεφάλου ἐκέλευσε δραμόντα τὸν παῖδα
περιμεῖναί ἐ κελεῦσαι. καὶ μου ὅπισθεν ὁ παῖς
λαβόμενος τοῦ ἱματίου, κελεύει ὑμᾶς, ἔφη,
Πολέμαρχος περιμεῖναι.

This...

```
private static int min(int[] nums)
{
    int m = nums[0];
    for (int i = 1; i < nums.length; i++)
    {
        int n = nums[i];
        if (n < m)
            m = n;
    }
    return m;
}
```

...is the same as this...

```
def min(nums):
    m = nums[0]
    for i in range(0, length(nums)):
        if (nums[i] < m):
            m = nums[i]
    return m
```

...is the same as this...

```
def min(nums):
    m = nums[0]
    for i in range(0, length(nums)):
        if (nums[i] < m):
            m = nums[i]
    return m
```

...and this...

```
min([X], X).
min([X | Xs], X) :-
    min(Xs, Y),
    X < Y, !.
min([_ | Xs], Y) :-
    min(Xs, Y).
```

...is the same as this...

```
def min(nums):  
    m = nums[0]  
    for i in range(0, length(nums)):  
        if (nums[i] < m):  
            m = nums[i]  
    return m
```

...and this...

```
min([X], X).  
min([X | Xs], X) :-  
    min(Xs, Y),  
    X < Y, !.  
min([_ | Xs], Y) :-  
    min(Xs, Y).
```

...and this.

```
LDA $0,X  
LOOP: CMP $0,X  
      BCC SKIP  
      LDA $0,X  
SKIP: DEX  
      BNE LOOP  
      RTS
```

Programming is like Pancakes

1. Put ingredients in bowl.
2. Beat until smooth.
3. Heat frying pan.
4. Pour 1/4 cup of mixture into pan.
5. Wait for bubbles.
6. Flip.
7. Wait 30s or until brown underneath.
8. Transfer to plate.
9. Repeat steps 4-8 until all mixture is used.

Procedural thinking

Programming is about thinking in terms of **step-by-step procedures** to achieve an outcome.

Mastering procedural thinking is more important than any particular programming language.

Computers are stupid

To put ingredients in bowl:

1. Take bag of flour from cupboard
2. Take bottle of milk from fridge
3. Take carton of eggs from fridge
4. For each person served:
 1. Transfer 1 cup of flour from bag to bowl
 2. Transfer 1 cup of milk from bottle to bowl
 3. Remove 1 egg from carton
 4. Crack egg and pour contents into bowl
 5. Put eggshell in compost bin

Computers are picky

```
void addIngredients(Bowl bowl, int numberOfPeople)
{
    Flour flour = theCupboard.get("Flour");
    Milk milk = theFridge.get("Milk");
    List<Egg> eggs = theFridge.get("Eggs");

    for (int i = 0; i < numberOfPeople; i++)
    {
        flour.transfer(1, bowl);
        milk.transfer(1, bowl);
        Egg egg = eggs.remove(0);
        egg.crackInto(bowl);
        egg.dispose();
    }
}
```

The essential skills of programming

Procedural thinking: knowing how to write a sequence of instructions to achieve an outcome.


Language literacy: knowing how to read and write code.

Testing and debugging: knowing how to test your code and track down bugs.

The 6 Stages of Programming

1. Requirements **What do they want?**
2. Specification **What should it do?**
3. Design **How will it work?**
4. Implementation **How is it made?**
5. Testing **Does it actually work?**
6. Debugging **What went wrong? How to fix?**

The 7th Stage of Programming

1. Requirements
 2. Specification
 3. Design
 4. Implementation
 5. Testing
 6. Debugging
 7. Documentation
- 

Good code

A well-written computer program:

1. Tells the computer what to do,
2. Tells another human being what the computer is going to do.

Bad Style

```
void AddIng(Bowl b, int n) {
    Flour f=c.Get("Flour");
    Milk m = fr.Get("Milk");
    List<Egg> e = fr.Get("Eggs");
    for (int i=0; i<n; i++)
    {
        f.Put(1,b);m.Put(1,b);
        Egg e1=e.rem(0);
        e1.cr();e1.Put(b);
        e1.Disp();
    }
}
```

Good style

```
// Add pancake ingredients to bowl:
//   bowl - the bowl to put them in
//   numberOfPeople - the number of people to serve
// Makes 2-3 pancakes each

void AddIngredients(Bowl bowl, int numberOfPeople)
{
    // 1 cup of SR flour per person
    Flour flour = theCupboard.get("Flour");
    flour.transfer(numberOfPeople, bowl);

    // 1 cup of milk per person
    Milk milk = theFridge.get("Milk");
    milk.transfer(numberOfPeople, bowl);

    // 1 egg per person
    List<Egg> eggs = theFridge.Get("Eggs");
    for (int i = 0; i < numberOfPeople; i++)
    {
        Egg egg = eggs.remove(0);
        egg.crackInto(bowl);
        egg.dispose();
    }
}
```

Style Guide

<http://www.bluej.org/objects-first/styleguide.html>