CO1005 Data structures and development environments

Dr Tom Ridge, tr61@le.ac.uk
https://campus.cs.le.ac.uk/teaching/resources/CO1005/

This version compiled 2018/01/17 at 12:02:38

Announcements

- No sessions on Friday 26th (I am in London on University business)

CO1005 in a nutshell

- In CO1003 you learnt some things about Java e.g. how to define a class, declare an attribute and declare a method.
- But to be a good (or at least, halfway reasonable) programmer you need to know much more than this.
- I'm particularly keen that you should learn how to program simple functions such as "reverse a string", or "reverse the words in a string" (e.g. as in CO1003 exam paper from Jan 2012).
- Actually, I want you to know that you can program these kinds of functions quickly, easily, and without making any mistakes.
- The exercises we will do **come up all the time in job interviews**
- There is a lot more interesting stuff, which we will come to in a bit, but the main point is this: *I will try to make you a better programmer*

Module webpage

- The module webpage is https://campus.cs.le.ac.uk/teaching/resources/CO1005/
- Please make sure you know where the module webpage is, and that you read it regularly.
- It is worth reading the study guide (available on the module webpage).
- Look at the timetable, and make sure you know when the lectures are (and where they are!).
- Look at the timetable, and familiarize yourself with the class test deadlines.
- I will now discuss what is on the webpage

CO1005 in a nutshell

- Datastructures: strings, lists, collections, sorting, trees, recursion
- Advanced Java features: inheritance and polymorphism; interfaces and abstract classes; exceptions
- Development environments: command line compilation and tools, packaging programs as jar files, IDEs and testing

CO1005 concept map

See concept map here

Concept map

- This is a nice clear picture with a clear modular structure but...
  - we don't do each block separately, instead...
    - each week we do bits of each of the main three parts of the course material.
- This allows us to see the connections, means we don't get bored just doing one thing, and allows us to apply things from one part to topics in another part.
- **BUT** this may mean that you perceive that we are doing a lot of different unrelated things each week. If you start to think that, just look at the concept map, and see how it all fits together.

CO1005 weekly sessions

- See the webpage for full details.
- Tue: Lecture
- Thu: Lecture, computation class (attend one only), surgery
- Fri: Lecture

Sessions this week

These **will** happen:

- labs on Thu
- lecture on Fri

These **will not** happen:

- surgery on Thu

Computation classes (labs)

- See the webpage for full details.
- These classes are compulsory and you must attend.
- They are supervised by the teaching assistants (TAs).
- You must complete the work on your own.
- The TAs are there to help you, but they are not there to do the work for you.
- You may have to submit your work. 10% of the module mark is awarded for making an effort during the labs (see next slide).
- The lab allocation (which student attends which lab) will be made available shortly

Lab assessment

- We want you to participate in labs.
- During the two hours that you are in the lab, you work on lab exercises.
- At the end of the lab, you upload a log of what you have done.
- The log includes a copy of your code at various points
- If you have made a reasonable effort, you pass that lab (1 point). Otherwise you fail the lab (0 points).
- Your lab mark is released periodically (say, every 3 or 4 labs).
- The first lab is a trial run through of the assessment process, so you can see how it works (it is not part of the assessment)

Lab feedback

- We have constructed a "Feedback machine", which runs tests on your code, and tells you when the code you have written is not correct.
- TAs and the convenor can also give feedback in labs.
- Lab solutions are released after the lab, and discussed in the surgery/problem class immediately after the lab (although there is no surgery in week 1)

Class tests, and assessed labs

- See the webpage for full details.
- Class tests are named CT1, CT2, and CT3 (or CW1 CW2 and CW3 in older documentation)
- Each class test lasts roughly 1 hour, under exam conditions (actually about 40 mins, but then the scripts have to be collected etc)
- The tests have weights (see the study guide for details)
- In addition, the labs contribute to the overall mark (10% of overall mark)

Class test marking and feedback

- See the study guide for details.
- The tests are marked partly by me and partly by the teaching assistants, based on marking schemes that I provide.
  - I look at what they have done, and make sure everything is OK.
- Typically marks will be returned electronically, and scripts will be returned so you can inspect where things went wrong. This should happen within 10 days.
- Feedback on individual questions will be given in a surgery, or in class.
- Additional feedback may be given explicitly in lectures, or I may alter the lecture material to address mistakes that are being made (so you won't necessarily realize that feedback is happening).

The relationships between the parts

A student who messed up has this to say:

> *The fact that the labs were not assessed lead to a grand failing on my behalf. I had no motivation to complete them. Some may have found this useful as a time to learn rather than churning out solutions, but for me I would often spend the time doing other, more important, assessed work.*

We have changed the labs so that they **are** assessed, but the point is: *you need to attend lectures, but the only way to learn to code is by actually coding (in the lab).*

Some comments on coding

Some small observations I have made over the years:

- **Programming is difficult**; it is in many ways very unnatural for humans
- Some people are more suited to programming than others; patience helps; being methodical and systematic helps
- Working with computers is **extremely frustrating** even if you know what you are doing; if you don't, then everything is almost impossible.
- In order to cope with the frustration, you need to be **motivated**. This is not something I can really give you. If you find programming boring, you will find the course very difficult. If you are motivated, and learn each bit as we go along, you will find the course very easy.

(continued)

- It is **very important that you learn each bit as we go along**. Programming is a field where later techniques build on earlier techniques (like science). You need to learn the basics before tackling the more advanced stuff. I am still learning new things all the time about (very advanced) programming.
- At the end of the module, there are essentially two classes of student: those who understand almost everything, and those that understand almost nothing. There are relatively few in-between. (It is difficult to recover if you slack off early in the module, or if you slacked off in CO1003)

How I like to think about programming

- Programming is a "game"; to win the game, you have to write your code so the computer does what you want
- The **"rules of the game"** are the rules that control how the computer behaves (**eg how a for-loop works**); the computer has to do exactly what you tell it, according to these rules
- The rules may not be clear, but you can find them out (hopefully); part of my role is to tell you what the rules are
- **if you don't know what the rules are, the game is impossible to win**
- The computer can be made to tell you exactly what it is doing so you can find out what is going wrong and correct it (this is "debugging")
- So the "game" is extremely fair, and ultimately you should always win the game (ie you should always end up with perfect code).

Questions?

- I'm happy to answer any questions you may have about the module.
- Please can I have the sign-in sheet back!

Older slides

To pass the course. . .

. . . what you should do is:

- Turn up on time. Behave reasonably.
- Take **written** notes during lectures:
- Most of the material is in the electronic slides etc., but I will give further explanation and motivation during lectures
- So you should: turn up to lectures; take notes
- Getting someone else to take notes and then copying them is not good enough (disabilities excepted)
- Please bring your notes, and other relevant material, to lectures
- Supplement the lectures with study/lab work in your own time (the study guide says 10 hours/week **additional** work).
- Put in effort to understand the material and the feedback.
- Manage your own time, accept the consequences of your actions etc.