

# CS1013 - Programming Project

Dr. Gavin Doherty

ORI LG.19

Gavin.Doherty@cs.tcd.ie

## Review - direct

- How to produce output on the screen and have it change.
- Dealing with lots of objects using arrays and lists.
- Structuring larger programs using a number of classes.
- How to take input from the user and use it to change what appears on the screen (updating variables, invoking methods).
- Structure of visualisation programs: load in data, handle user input, run queries, display results, navigate within results.

## Review - indirect

- Important to produce readable and comprehensible code, commenting code, using consistent naming conventions.
- Using a revision control system to manage group software projects.
- Working as a team on software projects is a skill in itself.
- Planning in advance makes teamwork on software projects run a lot smoother.

# The tar pit...



# The mythical man month

- Fred P. Brooks. 1979, 1995.
- Based on IBM OS 360 experiences
  - 5000 person-years of effort
  - Introduced 1963, completed in 1968
- No body of knowledge, no professionals, no mass market, no high level languages.
- Large system development is a tar pit
- A multitude of small problems slow you to a crawl.
- Q: How does a system get to be a year late?
  - A: *One day at a time*

# Myths and fallacies

- Poor estimation
  - Assumes nothing will go wrong
  - Hard to know all in advance
  - Probability of success in *every* step is small.
  - Most measures confuse effort with progress.
- Person-month
  - Throwing more people at a task which is behind schedule will often make progress slower.
  - Communication and training.
- Not planning to test
  - Many projects on schedule until testing phase. Not budgeted.
- Gutless estimating.
  - Need to learn how to give bad news
  - Need to learn when to tell the client “no”

## Programming teams

- Cost does indeed vary as the product of the number of people and the number of months. Progress does not!
- The unit of the person-month implies that people and months are interchangeable
- However, this is only true when a task can be partitioned among many workers with no communication among them!
- When a task is sequential, more effort does not necessarily improve schedule.
- Many tasks in software engineering have sequential constraints.

# Programming in teams

- Most applications are much too big to tackle alone
  - Too complex to analyse, too big to design, too much programming
  - One person doesn't have the monopoly on good ideas
    - however talented they are
- Increasing scope for confusion
  - Decisions aren't fully shared, people aren't notified of changes, ...
  - Not everyone understands the issues or ramifications of a decision
  - Difficult to achieve unanimity of design or coding styles
  - "Experts" will disagree on the "right" approach



## Communication

- “How, then, shall teams communicate with one another? In as many ways as possible”.
- Informally
- Meetings
- Logs & Tools

## Project - Code

- Comments
- Indicate authorship and changes at the top of every source file.
- Our project: Must have everything needed to run. Check this by checking out the repository to a lab machine and trying to run it.

# Project presentation

- Demonstrations will be 4pm-7pm Thursday in Regent House. BE THERE ON TIME.
- You have 5 minutes to present the features of your program (what you did, what is good about your design) in the demonstration.
- The final version for marking will be downloaded from subversion at 5pm Friday.
- You should have your presentation and demo on a USB key as a precaution.
- If your group does not have a laptop to present with TALK TO ME AFTER THIS LECTURE.
- As your slot approaches please have the demonstration machine ready to run, and make sure you know how to connect it to a projector - I suggest doing this after this lecture.

# Project report

- Outline of design
- How you split up the work and organised the team
- Features implemented
- Problems encountered
- Your report should be in PDF or DOC (word) format, 5 pages *MAXIMUM*, and should be uploaded to subversion as  
    CS1013-report-x.pdf
- Have until Friday 4pm to submit report.

## Demonstration

- We have a VERY tight schedule, so your demo must take no more than 5 minutes total.
- Move to the front in advance of your presentation.
- Decide in advance who is speaking + demonstrating.
- Have your laptop ready and make sure it can connect to the projector beforehand.
- If your program takes a while to start, please start it running when you begin your talk.