

## ACOM1900 Intro. to Programming:

### Bolting it all Together

Roy Ruddle

with thanks to Tony Jenkins & Nick Efford  
(CR11 and previous ACOM1900 material)

1

University of Leeds

Roy Ruddle

## Objectives

- So far
  - Professionalism
  - Data types
  - Basic I/O
  - Control flow
- How to develop programs for “real” problems
  - Software development lifecycle
    - Methodology
    - When (not) to use a computer
  - Algorithms & efficiency
  - Coursework 1!!!

2

University of Leeds

Roy Ruddle

## Part 1: The idea

- Programming is
  - Engineering? Maths? Artistic?
  - Structured, methodological process
- Software development lifecycle
  - Analyse: Understand the problem
  - Design alternative solutions
  - Build: Implement chosen solution
  - Test: Verify it works correctly

... and then do it all again?

3

University of Leeds

Roy Ruddle

## Analyse

- User’s requirements?
  - Loosely specified; ambiguous; contradictory
- Assumptions?
- Input/output?
- What might go wrong?

4

University of Leeds

Roy Ruddle

## Design

- Language independent
- In increasing detail
- System architecture
  - User interface
  - Database
  - Algorithm details & efficiency
- Document
  - Prototype (low vs. high fidelity)
  - Flow chart
  - Pseudo code (structured English)
    - [http://www.csc.calpoly.edu/~jdalbey/SWE/pdl\\_std.html](http://www.csc.calpoly.edu/~jdalbey/SWE/pdl_std.html)

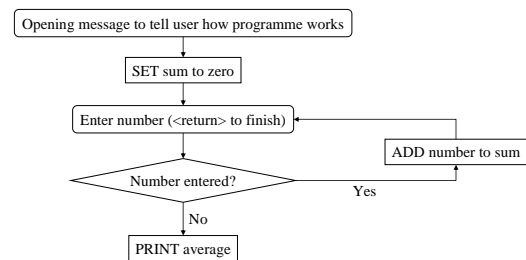
5

University of Leeds

Roy Ruddle

## Design: Flow chart

- Calculate average of a set of numbers



6

University of Leeds

Roy Ruddle

## Design: Pseudocode

```
PRINT message to tell user how programme works
SET sum to zero
SET counter to zero
```

```
REPEAT
  PRINT message "Enter number (<return> to finish)"
  READ number
  IF number was entered THEN
    ADD number to sum
    INCREMENT counter
  ENDIF
UNTIL no number entered

IF at least one number was entered THEN
  CALCULATE average
  PRINT average
ENDIF
```

7

University of Leeds

Roy Ruddle

## Build & Test

- **Build**
  - Implement the programme
  - Often trivial (if designed!)
- **Test**
  - Verify program works correctly
    - What needs to be tested?
    - How?
  - Test harness for each component (function)
  - Automate testing

8

University of Leeds

Roy Ruddle

## Example: Control flow Q5

- **User's requirements**
  - Write a program that prompts the user to enter a string consisting of digits. The program should examine the string and then display how many 1s, 2s, etc. were found.
- **Input: Enter string: 0113 343 5430**
- **Output might resemble:**
  - 0s: 1
  - 1s: 2
  - etc.
- **Assumptions?**
- **What might go wrong?**

9

University of Leeds

Roy Ruddle

## Control flow Q5: Design

- **Overall programme structure**
  - At high level
  - Then add detail
  - Data structures and algorithms
- **Write down your design**
- **Make it more efficient?**

10

University of Leeds

Roy Ruddle

## Summary

- **Software development lifecycle**
- **Algorithm efficiency**
- **Coursework 1**
  - Due 9am Mon 25 Feb
  - 30% of module mark

11

University of Leeds

Roy Ruddle