41012 Programming for Mechatronic Systems

Week 1

Overview

- Staff
- Link to Industry Needs
- Objectives
- Class Structure
- Assignments
- General Rules
  - Plagiarism
  - Late submissions
  - Peer Review
- In-class exercises

Subject Staff

- Subject Coordinators
  - Alen Alempijevic
- Teaching Staff
  - Alex Virgona

Mechatronic Systems

- 6.4B Devises Interconnected + System on
- Chip Devices Explosion
- Beyond Single Monolithic Code
- OO Paradigm – More Admissible to Systems
- C++ Essential on Many Layers
  - Android Backbone, Libraries, Applications
- Code Reuse / Testing / Documentation
- Robotics Jobs (demand)

Subject objectives

- 1. Design classes that are reusable, reliable and maintainable
- 2. Apply theoretical knowledge of sensors and control to practical programming problems
- 3. Select appropriate class structures and data handling methods for task at hand
- 4. Implement and test object-oriented applications of moderate complexity
- 5. Communicate programming design decisions, dependencies, interconnections, use cases and testing procedures in a written document

Class Structure

- Pre-readings given
- Students view readings, attempt and come to class
- We clarify concepts and push forward with examples
- We build knowledge base (stratify knowledge) towards the assignments; more complex layers of understanding

Assessment

- 1) Assessment I:
  Developing Sensor Class (5%)
- 2) Assessment II:
  Utilising Abstraction for a Range of Sensor Classes (20%)
- 3) Assessment III:
  Threading, Synchronization and Data Integrity (20%)
- 4) Individual Project: (40%)
- 5) Code Review: (9% : 3%+3%+3%)
- 6) Quizzes (6% : 6 x 1%)

Teaching Delivery

- OS: Ubuntu
- Compilation + Tools: CMake
- Documentation: Doxygen
- IDE: QTCreator
- Unit Testing: gtest
- Library : OpenCV
- CBSE / Middleware : ROS

General Rules

- Academic Conduct
  - Plagiarism
- Late submissions
  - Penalty: -20% per day, Up to 5 days
- Google Forms
  - Will be used for feedback (peer review)
  - Peer reviews will also be distributed (anonymous)
- Feedback
  - Provided on each assignment on UTS Online with associated Rubric

Compiling Samples

- Download the examples from Lynda.com
- Enable compiling of working.cpp from Chap1
- with CMake
  - Command line
  - QTCreator / Eclipse

Exercises

- Review of C
  - Pointers
  - Arrays
  - Functions

- Use material on following pages to query your understanding

Pointers & Reference

- Assign a double x of value 41012;
- Use a pointer ip to point to x
- Print the value of what ip is pointing to
- Make y reference to x
- Print the value of what y is referencing to
- Create a double z of value 1
- Use a pointer ip to point to z
- Make y to reference to z
- Print the value of what ip and y is referencing to

C arrays – MACRO

- Create an array x of doubles with 10 elements
  - Populate the elements of array on creating of array, each element [i] has value i
  - Can you create a loop to populate elements of x (each x[i] =i), how to code end of loop?)
  - Can you use a pointer and loop to initialise elements of array

C STRINGS

- Create an string array (char[]) x to value “41012”
  - do we need to specify size on initialisation?
  - Initialise with (INITIALISER LIST)
  - Can you create a loop to show elements x[i] (how to code end of loop, can we have a check for NULL, what is x[i]?)
  - Can you use a pointer and loop to print elements
- ADVANCED, Why does x[i]=NULL work for char[] for termination of loop, and will not work for double[]

Loops – Typecasting

- Create an string array (char[]) x to value “41012”
  - Can we typecast to integer each value (and what the value of each integer is?) (int)(x[i])
  - Add up all the elements (as numbers)?
  - Can we count number of elements less than 2
    - Use a for loop
    - Use a while loop
    - ADVANCED: Use a for range loop

Switch Case – Const

- Create an string array (char[]) x to value “41012”
  - Switch on items 4,1,0,2 and print word for value (what type needs to be in case clause?)

Functions

- Create a function that accepts a double value as a parameter and
  - 1. Returns a square value
  - 2. Returns a bool value if the double is greater than zero and the square value instead of initial passed value
  - 3. Returns bool value if the double is greater than zero, the square value, the cube value and the passed value incremented by one
  - 4. Loop over item 3 for 1-20

- ADVANCED: How best protect the passed value?