SYD366 Week 1

INTRODUCTION TO COURSE CONTENT

Today's Agenda

- Short Introduction
- Learning Outcomes
- Defining a System
- System Development Life Cycle

A Short Introduction

- Who am I?
  - Cassandra Laffan
  - Email: Cassandra.Laffan@senecapolytechnic.ca
  - PhD Student at Toronto Metropolitan University
  - Specialties include the C programming language, Python, CLISP
  - Hobbies include baking, working out, going to concerts, video games

Office Hours

- No one ever shows up to office hours properly
- Office hours are scheduled via email

Semester Timeline

- The course schedule is posted in the addendum
  - Familiarize yourselves with it!
- There are three tests in this course
  - Money test (5%, week 5)
  - Inventory test (20%, week 7)
  - Sales and Scheduling test (35%, last week of classes)

Lab Due Dates

- Labs become available on Sundays and are due the subsequent Sunday
- For example, lab 1 became available on the 3rd and is due on the 10th of December
  - There is only one exception: lab 3 is very long, so it will become available alongside lab 2
- **I do not accept late submissions, email submissions or .zip files**

Learning Outcomes

Learning By Example

- You'll be learning about businesses by reading about businesses!
  - Topics include money, inventory, accounting, sales and scheduling
- The examples you see throughout the semester will be used to frame and teach UML and OOP basics
  - You'll learn how to define classes, attributes and relationships
  - You'll create solutions using the MVC control pattern
  - You will define actions which the UI controller, Domain Controller and Entity Manager must each manage

Why UML?

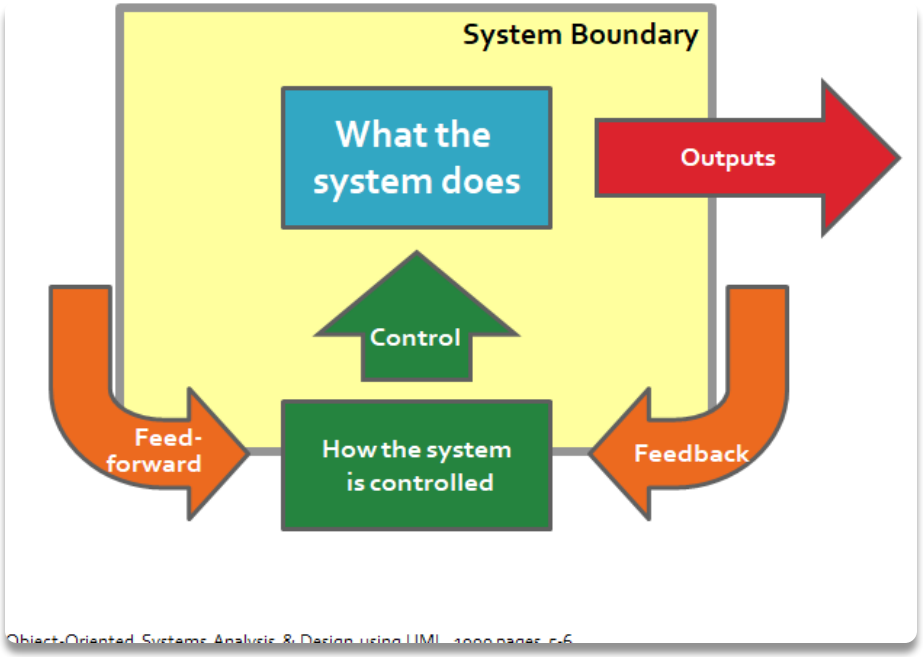
- You will learn very quickly that jumping into coding without knowing what your system will look like is a disaster waiting to happen.

Software Requirements

- Visual Paradigm Community Edition
  - Available through Seneca's Cloud Apps and on school computers
  - In the lab, I'll do a quick demo showing you how to create a class diagram in Visual Paradigm
- An IDE or text editor which can read .h files
  - I recommend VSCode

Systems

What is a System?



- A collection of inter-related components which collect, process and store input
- It then outputs the information needed to complete (business) tasks.

Characteristics of a System

- Exists in an environment
- Separated from its environment by a boundary
- Has inputs and outputs which come from, or are sent to the environment
- Has interfaces
  - These allow communication between two systems
- Can have sub-systems
- Has a control mechanism

What is a Software System?

A combination of hardware and developed software which create the solution to solve a problem or meet business requirements

What is Software Development?

- "Software Development" implies developing some sort of software
  - Does not involve simply coding programs
- Software is developed:
  - to turn manual processes into automated processes.
  - to improve/enhance existing automated processes.

What is Software Development?

- Software Development entails understanding:
  - how a business operates
  - the problem to be solved
  - that the solution to be developed will be of value to the business

Why do we develop software?

- Environments are rapidly changing
  - New Operating Systems, changes to office workspaces, etc.
- New technologies are frequently introduced
  - Haptics, voice-to-text, AR, VR, etc.
- Companies merge and need to combine their systems
  - Combine two separate database systems (a nightmare)
- Governments pass new legislation or make changes to it
  - New tax codes (also nightmarish)

Typical Software Development Solutions

- Each approach has pros and cons
- Customized
  - Developed in-house or contracted out
- Off-the-Shelf
  - Turnkey
- Combination
  - Off-the-Shelf software with custom components
  - Customized Software with Off-the-Shelf components

Learning Outcome: Systems Analysis

- What goes into a system?
- How do we determine what needs to be built?
- Who is involved with these decisions?

Learning Outcome: System Design

- How will our system work?
- What is the ‘flow’ of input and output?
- How does the user’s interaction affect the system’s behaviour?
- What will the system look like?
- How will data be stored?

Learning Outcome: System Development

BUILD A SYSTEM

CREATE ITS DOCUMENTATION

TEST THE SYSTEM

MAINTAIN THE SYSTEM

Quick Aside: Systems Stream Subjects

- Gives CPA students the experience of completing the SDLC using OOP project management design techniques
- SYD366 (this course!)
  - Experience the difference between Predictive and Agile Project Management methodologies
  - Follows a small business through selection of software
  - Build Class and Sequence Diagrams by reviewing User Stories and Systems Use Case Specifications. (Agile artifacts)
  - Research off the shelf software solutions and determine the best fit.

Quick Aside: Systems Stream Subjects

- Completion of the aforementioned courses will give you the tools to:
  - Understand the system development life cycle (SDLC)
  - Use and follow agile methodologies and artifacts

Potential Career Path: Systems Analyst

- Not a meme (I swear)!
- What is a Systems Analyst?
  - An IT professional involved in the development of a computerized solution to a business problem
  - Requires extensive technical, business and people knowledge, communication, business and technical skills
  - Focuses on understanding the business problem
  - Focuses on the approach to be taken to solve the business problem

Systems Analyst Skills

- Technical Knowledge and Skills
- Business Knowledge and Skills
- People Knowledge and Skills
- Integrity & Ethics

Speaking of Integrity and Ethics...

- Don't cheat.
- What is cheating?
  - Handing an assignment in you bought off a website
  - Handing in an assignment which you did not write yourself
  - Copying others' tests while writing your own

SDLC

Systems Development Life Cycle

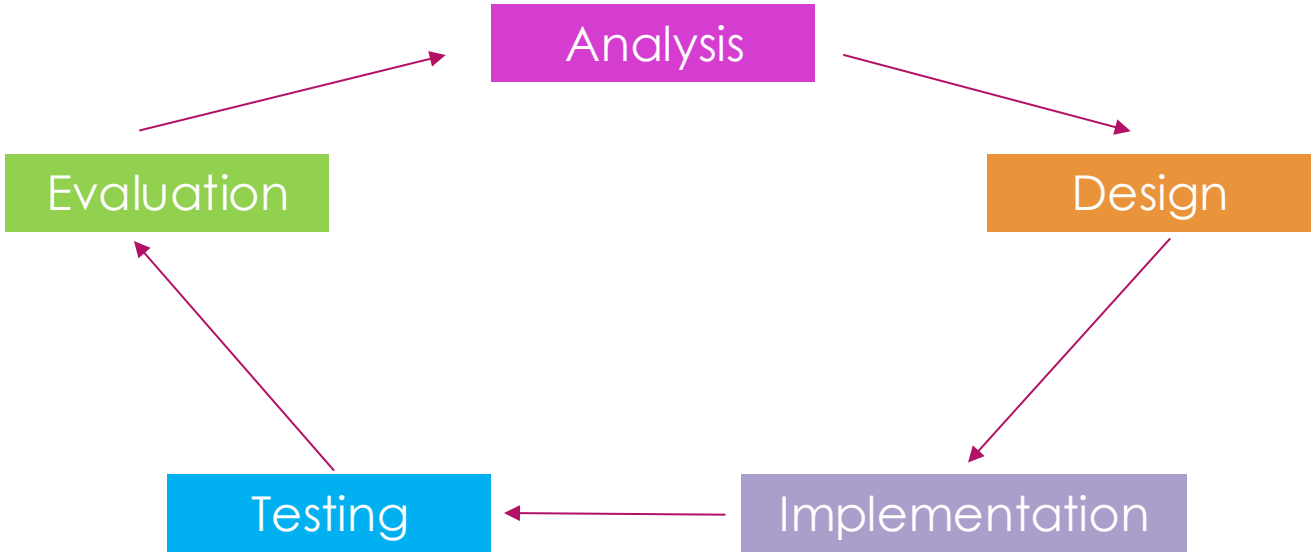
Software Development Projects are developed according to a definite methodology called the SDLC:

- Waterfall, Interactive and Incremental
- Organizes the activities of a project
- Followed by professionals involved in software development

Problem Solving Approach

1. Research and understand the problem
2. Verify that the benefits of solving the problem outweigh the costs
3. Develop a set of possible solutions (alternatives)
4. Decide which solution is best and make a recommendation
5. Define the details of the chosen solution
6. Implement the solution
7. Monitor to make sure that you obtain the desired results

Activities in SDLC



Analysis

- Systems requirements are determined, defined and documented
- Looks at functions (at a high level) and the data that will be used
- Defines what the system will do

Analysis

Understanding business needs includes:

- Researching and understanding the problem
- Identifying Stakeholders
- Identifying Business Needs
- Documenting Business Processes

Design

Conceptualizing computer-system solutions:

- Developing a set of possible solutions (alternatives)
- Deciding which solution is best and make a recommendation
- Defining the details of the chosen solution

Implementation

- Coding
- Creating data constructs
- Installation
- Deployment to testing environments

Testing

- Finding bugs, glitches and other abnormalities in your program
- Error tracking

Evaluation

Determining the success of a solution:

- Monitoring to ensure the desired results are obtained
- Determining what changes, if any, need to be made

What is a Methodology?

- A body of methods, rules and postulates employed by a discipline: a particular procedure or set of procedures.
- Within the context of systems, it's a set of comprehensive guidelines to follow while completing every SDLC activity:
  - Structured (traditional)
  - Object-oriented

Methodology: Waterfall (Structured)

- Worked well for centralized processing applications and procedural languages
- Linear methodology, difficult to evolve project
- Rigid Development, minimal reusability
- Uses Data Flow Diagrams and Entity Relationship Diagrams

Methodology: Agile

- Pretty much ever modern company uses this methodology!
- Effective for most solutions, but particularly user-facing ones such as GUIs and websites
- Useful when constantly making many small changes
- You'll repeat the SDLC cycle until your boss tells you to stop

Variances in SDLC

Developers encounter many variations of SDLC in practice:

- Phases may vary
- Number of iterations may change
- Emphasis on people
- Speed of development (agile cycles can be a week to a month!)

BUT you must understand the basic methodology before you can vary it!

Why might Software Projects Fail?

- Common causes of failure:
  - Deadlines that cannot be met
  - Budgets that have been exceeded
  - Solutions that don't work as defined or required
  - Systems too complex to maintain
  - Customer's requirements not fully understood or captured correctly
  - Customers continually change their requirements
  - Customers are not committed to the project
- SDLC helps minimize failure by adding details in successive iterations and incremental releases of software.

Any questions?