Course Guide

CMSC 23 - Programming Paradigms *1st Semester AY 2020 - 2021*

Course Description

Major programming paradigms: imperative, functional, logic, and object-oriented programming. The first half of of the course is a tour of these paradigms. The second half of the course focuses on object-oriented programming paradigm.

Course Learning Outcomes

After completing this course, you should be able to:

1. compare and contrast multiple programming paradigms (imperative, functional, logic, and object-oriented), and identify problems in which using one paradigm is appropriate over another

2. design and implement a class.
3. use subclassing to design simple class hierarchies that allow code to be reused for distinct subclasses.
4. explain the relationship between object-oriented inheritance (code-sharing and overriding) and subtyping (the idea of a subtype being usable in a context that expects the supertype)
5. use object-oriented encapsulation mechanisms such as interfaces and private members.
6. design class relationships that adhere to object-oriented design principles
7. identify programming problems in which an object-oriented design pattern is suitable, and implement the pattern in a higher-level object-oriented programming language

Course Outline

1. *Introduction to programming paradigms*
    1. *Paradigms and the definition of programming*
    2. *Taxonomy of programming languages*
    3. *Multi-paradigm languages*
2. *Imperative programming*
    1. *Communicating imperatively*
    2. *State*
    3. *Assignment statements*
    4. *Structured program theorem*
3. *Functional programming*
    1. *Lambda Calculus*
    2. *Higher-Order functions*
    3. *Consequences of Statelesness*
4. *Logic programming*
    1. *Prolog facts, queries, and rules*
    2. *Unification*

3. *Proof Search*
4. *Advantages and disadvantages*

5. *Object-oriented programming*
    1. *OOP as a solution to the perils of state*
    2. *Classes and Objects*
    3. *The surface and the volume*
    4. *Fundamental concepts of OOP*
6. *SOLID Design principles*
    1. *Single-Responsibility Principle*
    2. *Open/Closed Principle*
    3. *Liskov Substitution Principle*
    4. *Interface Segregation Principle*
    5. *Dependency Inversion Principle*
7. *Design Patterns Introduction*
    1. *History of design patterns*
    2. *Why patterns and why not patterns*
    3. *Classification of design patterns*
8. *Creational Patterns*
    1. *Factory method pattern*
    2. *Abstract factory pattern*
9. *Behavioral Patterns*
    1. *State pattern*
    2. *Strategy pattern*
    3. *Command pattern*
    4. *Iterator pattern*
    5. *Observer pattern*
    6. *Template method patterns*

10. *Structural Patterns*
    1. *Decorator Pattern*
    2. *Adapter pattern*

Mode of Delivery

This course will be delivered asynchronously through lecture notes with accompanying lecture videos. This course can also be found online on VLE. We wont be having scheduled zoom sessions but you're free to setup consultation on the scheduled times for this class.

Most of the work for this course would be dedicated to the lab exercises.

Enrolling through VLE

Look for the course Programming Paradigms in VLE and enroll yourself to the course. Make sure you enroll using the correct enrollment key for your section:

- Section A - **sectiona2020**
- Section B - **sectionb2020**
- Section C - **sectionc2020**

Joining the Slack workspace

Make sure to join the slack workspace as soon as you can.

https://join.slack.com/t/up-4zo6452/shared_invite/zt-h0vbp2du-PAaMMZ5jr8C1NIo2odcAjg

Asking Questions and Scheduling Remote Consultations

I'll try to make myself available for questions most of the time but you'll have a better chance of reaching me during our schedules for lec and lab. You can schedule consultation as a group. Use that time to ask questions about the lecture or ask for help in answering lab exercises.

You can contact me through sms, email, slack dms, or through discord. You can find ways to reach me at the last section of this guide.

Course Materials

All of the resources in this course can be found in the course pack. The course pack includes:

- **Lecture Notes** - You'll find these in the PDF file called *textbook.pdf*.
- **Lecture Videos** - There's a link to the youtube playlist here. If you want the master copies (not really super high quality but they're uncompressed version, you can find it here.)
- **Lab Exercise guides** - Not really lab exercises since you'll be doing it at home, You can find them on the textbook as well.
- **Some sample code** - Some code that I use in class
- **PowerPoint Presentations** - will not really be useful since the lecture notes are way more detailed
- **Installs for Haskell, Prolog and Python** - Includes instructions for installation

Study Schedule

This class is asynchronous, but if you start too late, you might end up getting overwhelmed by the amount of work. Here's the recommended schedule for completing the resources and lab exercises.

Week Number	Topic	Videos to watch	Lecture Notes to read	Lab Exercises
1	Introduction to Programming Paradigms	Introduction to Programming Paradigms	Introduction to Programming Paradigms	None
2	Imperative Programming Paradigm	Imperative Programming Paradigm	Imperative Programming Paradigm	(Optional) Lab Exer 1
3	Functional Programming Paradigm	Functional Programming Paradigm, Lab Exercise 2, Lab Exercise 3	Functional Programming Paradigm, Haskell Cheat Sheet	Lab Exer 2, Lab Exer 3
4	Logic Programming Paradigm	Logic Programming Paradigm	Logic Programming Paradigm	Lab Exer 4,
5	Object Oriented Programming Paradigm	Object Oriented Programming Paradigm	Object Oriented Programming Paradigm, Python Introduction	Lab Exer 5,
6	OOP Principles: Fundamentals of OOP, Class Relationships, Exceptions,	Fundamentals of OOP, Class Relationships, OOPython	Fundamentals of OOP, Class Relationships, OOPython, Exceptions	Lab Exer 6,
7	Solid Design Principles, Design Patterns Introduction	Solid Design Principles, Design Patterns Introduction	UML for Class Relationships, Solid Design Principles, Design Patterns Introduction, Extra Stuff	Lab Exer 7,

Week Number	Topic	Videos to watch	Lecture Notes to read	Lab Exercises
8	Creational Patterns	Creational Patterns	Creational Patterns	Lab Exer 8, Lab Exer 9
9	Behavioral Patterns I: State and Strategy	Behavioral Patterns I: State and Strategy	Behavioral Patterns I: State and Strategy	Lab Exer 10, Lab Exer 11
10	Behavioral Patterns II: Command and Observer	Behavioral Patterns II: Command and Observer	Behavioral Patterns II: Command and Observer	Lab Exer 12, Lab Exer 13,
11	Behavioral Patterns III: Template and Iterator	Behavioral Patterns III: Template and Iterator	Behavioral Patterns III: Template and Iterator	Lab Exer 14, Lab Exer 15
12	Structural Patterns	Structural Patterns	Structural Patterns	Lab Exer 16, Lab Exer 17

Course Requirements

There will be no exams in this course, 100% of your grades will come from the lab exercises.

Lab Exercises

The deadlines written on the lab exercise guides are all for November 30, 2020, which is the end of the semester. But please do not start this at the end of the semester. You can refer to the study schedule above for the recommended pace.

- Lab Exercise 1 (Imperative Programming Review)
- Lab Exercise 2 (Introduction to Haskell)
- Lab Exercise 3 (Higher Order Functions in Haskell)
- Lab Exercise 4 (Drama in the Clue Mansion)
- Lab Exercise 5 (Snakes)
- Lab Exercise 6 (Library System)
- Lab Exercise 7 (Designing an OOP System from Scratch)
- Lab Exercise 8 (Factory Method)
- Lab Exercise 9 (Abstract Factory)
- Lab Exercise 10 (Strategy)
- Lab Exercise 11 (State)
- Lab Exercise 12 (Commmand)
- Lab Exercise 13 (Observer)
- Lab Exercise 14 (Template)
- Lab Exercise 15 (Iterator)
- Lab Exercise 16 (Decorator)
- Lab Exercise 17 (Adapter)

About the Instructor

You know me. Its your CMSC 56 or 57 teacher. I've developed and taught this course for the first time last year (enjoy some of the ugly drawings I made just for this course). This is the second time I'm delivering this course. This course might be a little rough around the edges sometimes and it may lack some refinement, but I love teaching this course. For the previous run I saw my students enjoy the lab exercises in some form or another.

To be honest I'm kind of disappointed that I wont be physically present while your working on the lab exercises. But, don't worry I will try my best to help you remotely if you have issues in your code or something.

Let's be understanding of each other in these strange times.

How to reach me

Don't share these info please

- SMS +63 921 031 2455
- email rrabella@up.edu.ph
- discord ruberoni#5100