Object Oriented Programming

Lecture 1

Engr. Sara Rehmat, MS(CS)

Contents to be covered today

- Introduction (mine and yours)
- Introduction to the course
    - Course Objectives
    - Course Policies - Attendance, Grading, Plagiarism Tolerance
    - Evaluations to be used (type, frequency and weightage)
    - Books and Resources
    - Course Outline
- Object Oriented Paradigm
- Why learn C++?

Introduction

- Sara Rehmat, MS (CS)
- Email-ID: sara.rehmat@nu.edu.pk; sara.rehmat@pwr.nu.edu.pk;
- Slack Channel: Workspace Channel: #oop-f21
- Office no. 33, First Floor

Introduction to the course

- Course code: **CS217** (Core Course - Prerequisite to Data Structures Course)
- Credits: 3+1
- **Course Objectives:**
  - **Primary Objectives**
    - Understand principles of object oriented paradigm.
    - Identify the objects & their relationships to build object oriented solution
    - Model a solution for a given problem using object oriented principles
    - Examine an object oriented solution
  - **Secondary Objectives**
    - Learning to program in C++

Introduction to the course - Policies

- **Attendance Policy**
  - 80% attendance (even in online classes)
  - Attendance will be marked on the basis of class participation in online classes.
- **Grading Policy**
  - Absolute grading (at least 50% total score required to pass the course)
- **Plagiarism Policy**
  - Tools are available to check plagiarism from the internet as well as from classmates.
  - Zero tolerance (zero marks to any assignment found plagiarized)

Introduction to the course - Evaluations

- Quizzes (On-Campus)
  - Frequency depending on the on-campus duration of the course
  - Weightage - **0% to 4%**
  - Both announced and unannounced
- Assignments (Online and On-Campus)
  - 6 (at least)
  - Weightage - **8%** to **12%**
- Semester Project
  - Different Milestones
  - Weightage - **10%**
- Class Participation
  - Weightage - **3%**
- Sessional Exams
  - 1st Sessional Weightage - **15%**
  - 2nd Sessional Weightage - **15%**
- Final Exams
  - Weightage - **45%**

Semester Project

- Demonstration of the OOP concepts taught in the course
- Developed in groups of 2-3
- Grading Criteria
  - Understanding of the code
  - Modular Structure with OOP concepts applied
  - Coding style (clarity achieved by indentation and comments)
  - Uniqueness
  - User Interface

Semester Project

Milestone	Due Date	Weightage
Team Formation	End of Week 2	0.5
Project Proposal Submission	End of Week 4	1
Identification of Classes with the description of their data members and functions	End of Week 10	2.5
Demo of the First Prototype (at least 60 percent functionality delivered)	End of Week 10	4
Demo of Final Working Version	Week 15	4

Books

- Text Books
  - **Thinking in C++: Standard Libraries and Advanced Topics** by **Bruce Eckel,** Volume 1, 2nd Edition (soft copy uploaded on SLATE)
  - **C++ The Complete Reference** by **Herbert Schildt,** Fourth Edition
- Reference Books
  - **The Art of Computer Programming** by Donald Knuth
  - **C++ How to Program** by **Deitel and Deitel**, Tenth Edition
  -

Course Outline

https://docs.google.com/spreadsheets/d/1VmmjS069MIzUfM6tzAAp-VfkiSFIAi2hvpIegjGAPnk/edit?usp=sharing

Primary Objective: Learn Object Oriented Paradigm

- Paradigm - pattern or model
- Programming Paradigms
  - Procedural
  - Object Oriented
- Procedural Paradigm:
  - You’ve followed so far in Programming Fundamentals course
  - Divide the program into units called procedures or functions or routines.
  - Each procedure (or function or routine) simply consists of a series of computational steps to be carried out.
  - During a program’s execution, any given procedure might be called at any point, including by other procedures or itself.
  - Suitable for small-scale applications that require low maintenance.

Primary Objective: Object Oriented Paradigm

- Based upon the concept of objects rather than functions/procedures.
- Closer to the real-world.
- Objects contain data in the form of attributes and code in the form of methods.
- Computer programs are designed using the concept of objects that interact with real world.
- Main Concepts of OOP:
    - Abstraction
    - Encapsulation
    - Inheritance
    - Polymorphism
- Makes programs more
    - Reusable
    - Secure
    - Easy to design as closer to real world
- Examples: C++, Java, Python, R,

Secondary Objective: Learning C++

- Why learn C++?
    - Mostly all general purpose programming languages are equivalent (i.e. a functionality implemented in one language can be implemented in another : see Turing Completeness if you are interested)
    - Differences between C++ and Python
        - Python is more high level language than C++.
            - Programs in Python are easier to write, debug and maintain.
            - Programs in C++ are more memory efficient.
        - Python is interpreted language while C++ is compiled.
            - C++ is faster than Python
        - Python is weakly typed and C++ is strongly typed language.

Secondary Objective: Learning C++

- C++ is used to develop all kinds of embedded systems like smartwatches, multimedia systems in automobiles, IoT devices, etc.
- C++ also allows you to develop the servers and the high-performance microcontroller programs.
- Game development is the key to C++. That's why C++ is becoming more popular among game developers.
- Python is mostly used in Machine Learning, web development and data analytics.
- On TIOBE (the language popularity index), Python is ranked at 2 and C++ is ranked at 4.

First Program (Hello World) in C++

```
// Your First C++ Program


#include <iostream>


int main() {
    std::cout << "Hello World!";
    return 0;
}
```