Object Oriented Programming with C++

Basics of OOP

By: Prof. Pandav Patel

Second Semester, 2020-21
Computer Engineering Department
Dharmsinh Desai University

What is Procedure-Oriented Programming?

- Solution to problem is viewed as sequence to tasks
  - Functions are created to accomplish those tasks
  - Primary focus is on functions
- Very little attention is given to the data and how data is manipulated by functions
  - Many data items are placed at global level
    - Global data is prone to unintentional/accidental modification by function
    - If we want to change external data structures, we need to find all functions which use it and need to change those functions. In larger s/w, its not easy to find all functions which use particular data item
- Does not model real world problems very well
  - Because functions are action oriented, do not really correspond to the elements of the problem
- Follows top-down approach in program design

What is Object-Oriented Programming?

- Treats data as critical element
  - Does not allow data to flow freely
  - Ties data closely to the functions which can operate on it and protects it from accidental modification from other functions
  - Emphasis is on data rather than procedure
- Programs are divided into objects
- Object ties together data and functions which can operate on its data
- Data can not be easily modified by functions outside object (data is hidden from outside functions)
- Objects may communicate with each other using functions
- Follows bottom-up approach in program design

Basic concepts of OOP

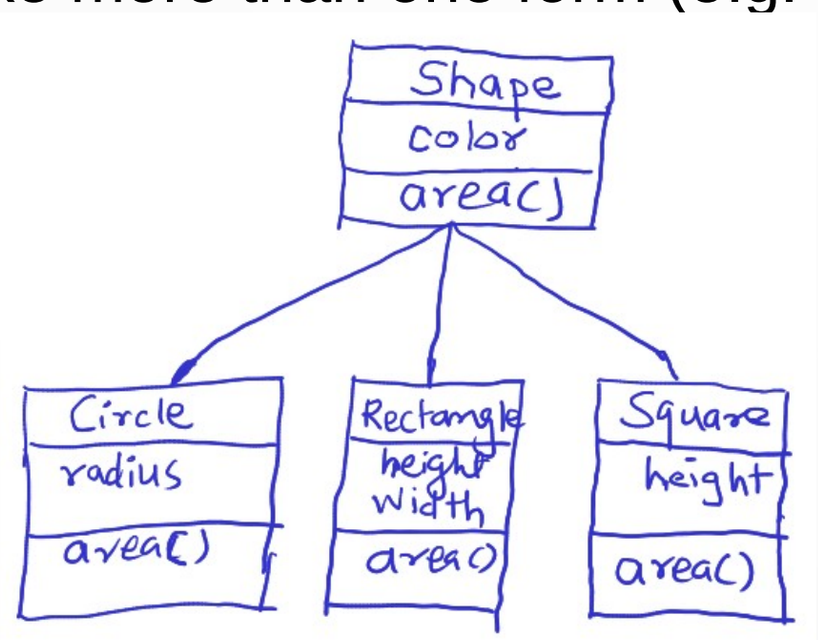
- Objects
  - Runtime entities
  - Instances (variables) of class
  - May represent real world object (e.g. a person) or concept (e.g. an account)
  - Contain data (attributes) and methods (member functions) to operate on data
  - Interact without knowing details of each other. Only interface is known
- Classes
  - User-defined data types (a.k.a. Abstract Data Types)
  - Once class is defined, multiple objects of that class can be created
  - Can be understood as collection of objects as well
- Principles of OOP
  - Encapsulation, Abstraction, Inheritance, Polymorphism

Principles of OOP

- Encapsulation
  - each object keeps its state private, other objects don’t have direct access to this state
  - Other objects can only call a list of public functions called methods
- Abstraction
  - abstraction means that each object should only expose a high-level mechanism for using it – by other objects
  - This mechanism should hide internal implementation details

Principles of OOP

- Inheritance
  - Child class derives from parent class (Reusability)
  - e.g. **User** - parent class **Buyer** and **Seller** – child classes of User
  - Child class reuses all fields and methods of the parent class (common part) and can implement its own (unique part).
- Polymorphism
  - Ability to take more than one form (e.g. operator and function overloading)



Benefits of OOP

- Reusability of code
- Software complexity can be easily managed
- Division of work can be easily done based on classes
- Easy maintainability
- More secured programs
- Programming feels closer to real world

- There are many languages which support OOP concepts

- Today, OOP concepts are used to design most of the applications