Part 1: Introduction

This software project aims at creating an application that displays an interface with two buttons, each with their own functionality. The first button generates six shapes that can be squares, rectangles, or circles. The second button sorts the six shapes based on their surface and displays the shapes on the interface.

The challenges associated to the software project include innovating an efficient design for the software project, using appropriate sorting algorithms in terms of space and time complexity, and presentation of various design patterns of the software project via UML diagrams.

There are many concepts used throughout the software project. The first one is object-oriented design (OOD). Object-oriented design allows defining software object’s states and behaviours, and the way these objects interact to achieve the desired achievements. In other words, Object-Oriented Design specifies software objects and the way they collaborate to satisfy requirements. OOD utilizes objects which comprise data and methods (features of an object) ,and enables reusability and maintainability being grouped into classes. Another concept used in the software project is Object-Oriented Design Principles (OOD Principles), which consists of abstraction, encapsulation, polymorphism, inheritance. Abstraction prevents all the detailed data and methods of a class to be known by other classes to ease their interactions. Encapsulation keeps an object’s state and methods inside a class, disabling direct interaction with other classes’ objects. Inheritance consists of creating a child and a parent class where the child class can reuse the parent state and methods without changing them. Polymorphism gives the ability of a method to take different shapes (having several methods with the same name but different parameters). Lastly, the concept of design patterns is used in the software project. The design pattern identifies the classes and instances, their roles, the dependencies between them, and the distribution of classes responsibilities. There are 23 design patterns in three categories, creational, structural, and behavioural design patters. Creational design patterns focus on instantiation of objects and allow decoupling a system from the way its objects are instantiated, composed, and represented. The Creational design patterns used in this software project include Singleton and Factory. The Structural design patterns focus on the composition of classes and objects to design larger structures and achieve new functionality. Lastly, the behavioural design pattern focusses on the communication between objects and on the assignment of responsibilities between objects through composition.

This report is structured in parts, following the introduction, the report explains in further detail the design of the software project, the implementation of the software project and a conclusion of the report.

Part 2: Design

Diagram

Description automatically generated

In the design of this software system, I have used Object Oriented Design Principles which include inheritance, abstraction, and encapsulation. Throughout the design, encapsulation has been used almost everywhere, since most classes have their own set of states and methods. It disables the direct interaction among other classes, only assessable through mutator methods. These methods are the ‘get’ methods utilized to get the attributes of the shape. The concept of having child classes like circle, square, and rectangle extend from the shape abstract class also follow the design principle of inheritance. These child classes inherit all the methods and states of the Shape abstract class. The principle of abstraction is demonstrated using shape abstract class. The abstract class allows inner implementation of the class to be hidden from other classes, and it prevents the class from being instantiated.

Part 3: Implementation

Part 4: Conclusion