# Введение

Для выполнения курсовой работы требуется написать программу, с использованием механизмов объектно-ориентированного программирования. Объектно-ориентированное программирование - это парадигма разработки программных систем, в которой приложения состоят из объектов, то есть сущностей, со своими свойствами и поведением. Обычно объектами являются экземпляры классов, а связанные с этим объектом данные называют свойством. Поведение объекта определяется методами, которые можно вызывать из разных частей программы.

Использование этих свойств и методов, ускоряет разработку и делает код ассоциативно понятным. Так же это помогает программисту создавать код, основываясь на объекте и его отдельных компонентов.

# **1. Объектно-ориентированный анализ предметной области**

Объектно-ориентированное программирование основано на представлении программы в виде совокупности объектов, каждый из которых реализовывается в определенном классе. В свою очередь классом является ссылочный тип данных, который описывает объект - конкретный экземпляр класса, хранящийся в памяти компьютера как переменная соответствующего типа. Классы имеют поля, свойства и методы. Метод – это процедура или функция, которая работает с полями данных. Поэтому с помощью методов можно управлять действиями классов. Свойства похожи на поля, но имеют дополнительные описатели, определяющие механизмы записи и считывания данных.

Класса называют ссылочным типом данных, потому что при копировании переменной через присваивание копируется интерфейс, а не данные. Поэтому классы разрабатывают таким образом, чтобы данные соответствовали объекту и решаемой задаче. Благодаря наследованию можно спроектировать и реализовать целостность природы объекта и его интерфейс.

Объектно-ориентированное программирование состоим из четырёх основных принципов: наследование, абстракция, инкапсуляция и полиморфизм. Использование каждого из этих принципов помогает в разработке программ и экономит большое количество времени.

Наследование — механизм, который позволяет описать новый класс на основе существующего. При этом свойства и функциональность родительского класса частично или полностью заимствуется новым классом.

Класс, от которого производится наследование, называется родительским или базовым, а новый класс потомком или производным классом. Основными плюсами этого принципа, является повторное использование кода и возможность перенять поля и методы, описанные в родительских классах, наследником.

Абстракция означает выделение наиболее значимых характеристик предмета и отбрасывание незначительных. В ООП рассматривается только набор наиболее значимых характеристик объекта, доступных остальной программе.

Инкапсуляция означает ограничение доступа к данным и возможностям их изменения. Так же это позволяет объединить данные и методы, для использования их в классе. Основной целью инкапсуляции является обеспечение согласованности внутреннего состояния объекта.

Полиморфизм позволяет работать с объектами, у которых одинаковый интерфейс, но разное поведение. Поведение меняется в зависимости от типа, к которому принадлежит объект.

При проектировании программы возникает необходимость в предварительной разработке информационной модели предметной области - концептуальной схемы, которая отражала бы взаимосвязи между объектами предметной области и поддержания целостности данных. В этом поможет объектно-ориентированный анализ. ООА – это методология анализа предметной области, при которой требования к проектируемой системе воспринимаются с точки зрения классов и объектов.

В объектно-ориентированном анализе основное внимание уделяется созданию моделей, более близких к реальности, с использованием объектно-ориентированного подхода.

Выделяют три этапа ООА:

* Построение информационной модели, абстрагирование реальных сущностей в терминах объектов и атрибутов.
* Построение модели состояний для формализации жизненных циклов объектов и отображение этой модели диаграммами и таблицами переходов, взаимодействие между объектами осуществляется путем передачи сообщений о происходящих с ними событиях.
* Разработка модели процессов, в которой действия в моделях состояний расчленяются на фундаментальные и многократно используемые процессы.

Предметная область - это мысленно ограниченная область реальной действительности, подлежащая описанию или моделированию и исследованию. Она состоит из объектов, различаемых по свойствам и отношениям между собой. В предметной области «Программное обеспечение» можно разделить три класса:

* Системное ПО - системные программы, выполняющие различные вспомогательные функции, например, управление ресурсами компьютера, проверка работоспособности устройств компьютера;
* Прикладное ПО - прикладные программы, непосредственно обеспечивающие выполнение необходимых пользователям работ;
* Системы программирования - инструментальные программные системы, облегчающие процесс создания новых программ для компьютера.

Программное обеспечение - это совокупность программ обработки данных и необходимых для их эксплуатации документов. Программное обеспечение является очень широким понятием, оно включает в себя системное программное обеспечение, которое отвечает за работоспособность компьютеров, прикладное программное обеспечение, предназначенное для решения задач любой предметной области в виде пакетов прикладных программ и инструментарий технологии программирования.

Все виды программного обеспечения имеют общие черты, такие как: режим эксплуатации, функционал, требования надежности, исходный язык, вычислительная система и среда, способ использования и распространения. Все эти критерии можно объединить в класс-интерфейс, который согласует работу с основными полями всех ПО для дальнейшей полиморфной обработки. Также нужен базовый класс, который будет содержать в себе основные поля для всего программного обеспечения.

Так как ПО делиться на три класса, а каждом классе есть свои программы, то с помощью наследования можно реализовать это.

В результате проведенного объектно-ориентированного анализа была построена контекстная диаграмма классов (рис. 1)

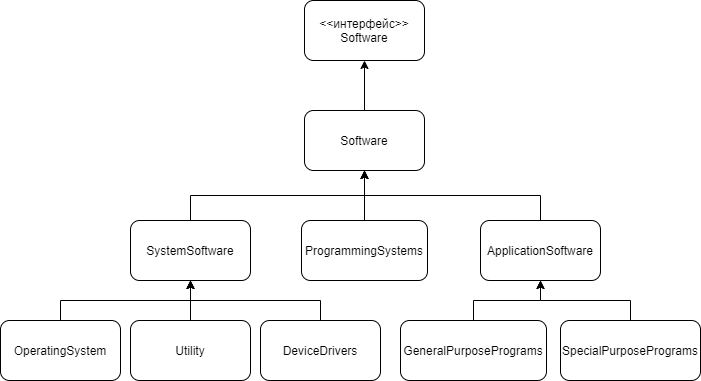


Рис. 1. Контекстная диаграмма классов