Министерство высшего образования и науки Российской Федерации

Федеральное государственное образовательное автономное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет» (ПНИПУ)**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

ОТЧЁТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ

Дисциплина: Информатика

Тема: Использование конструкторов

Выполнил

Студент группы РИС-22-2б

Худеньких В.Д.

Проверил

доц. кафедры ИТАС

Полякова О.А.

Пермь 2023

**Содержание**

Введение **3**

Разработка программы вычисления значений функции с использованием языка программирования C++**9**

Заключение**11**

**Введение**

Целью данной лабораторной работы является изучение принципа подстановки на языке программирования C++ и его применение в программировании. Принцип подстановки является одним из принципов SOLID, описывающим возможности использования производных классов вместо базовых классов без изменения корректности программы.

В процессе выполнения лабораторной работы будет создана программа на языке C++, демонстрирующая применение принципа подстановки. В результате выполнения работы будут получены знания и навыки, необходимые для применения принципа подстановки в реальных программах.

Основные этапы работы будут связаны с изучением теоретических аспектов принципа подстановки, созданием базового класса и производных классов, а также с написанием программного кода, использующего принцип подстановки. В процессе работы будут использованы языковые конструкции C++, такие как наследование, полиморфизм, абстрактные классы и виртуальные функции.

В результате выполнения лабораторной работы студент получит знания и навыки для правильного применения принципа подстановки в своих проектах и сможет создавать более гибкие и масштабируемые программы на языке C++.

**Разработка программы с использованием языка программирования C++.**

Теоретические аспекты принципа подстановки

Принцип подстановки является одним из принципов SOLID (Single responsibility, Open-closed, Liskov substitution, Interface segregation, Dependency inversion). Он описывает, что производный класс должен быть способен использоваться вместо базового класса без изменения корректности программы. Таким образом, принцип подстановки требует соблюдения правил наследования: поведение производного класса не должно противоречить поведению базового класса. Производный класс может расширять или специализировать поведение базового класса, но не переопределять его. В случае необходимости изменения поведения базового класса, следует создать новый класс на его основе, а не изменять существующий.

Создание базового класса и производных классов

Для демонстрации принципа подстановки будет создан базовый класс "Фигура", содержащий методы для вычисления периметра и площади фигуры. На основе базового класса будут созданы производные классы "Треугольник", "Круг" и "Прямоугольник".

Применение принципа подстановки

Будет создан программный код, позволяющий работать с объектами разных классов через указатель на базовый класс "Фигура". В процессе работы программы производные классы будут использоваться вместо базового класса без изменения корректности программы.

Результаты выполнения работы

Результатом выполнения работы станет рабочая программа, демонстрирующая применение принципа подстановки и возможности использования производных классов вместо базовых. В результате работы студент получит знания и навыки для правильного применения принципа подстановки в своих проектах и сможет создавать более гибкие и масштабируемые программы на языке C++.

**Заключение**

В ходе выполнения лабораторной работы были изучены теоретические аспекты принципа подстановки, создан базовый класс и производные классы, а также разработан и протестирован программный код, демонстрирующий применение принципа подстановки.

Применение принципа подстановки позволяет создавать более гибкие и масштабируемые программы. На основе базового класса можно создавать неограниченное количество производных классов, расширяя или специализируя поведение объектов. В то же время, изменения в базовом классе могут привести к нарушению работоспособности программы и повлечь за собой большие затраты на исправление ошибок.

Таким образом, принцип подстановки должен соблюдаться при проектировании и разработке программного обеспечения. В результате выполнения работы студент убедился в важности соблюдения принципов SOLID и научился применять их на практике при разработке программ на языке C++.

**ПРИЛОЖЕНИЕ А**

**UML-схемы программ**

