МИНИСТЕРСТВО СВЯЗИ И ИНФОРМАТИЗАЦИИ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«БЕЛОРУССКАЯ ГОСУДАРСТВЕННАЯ АКАДЕМИЯ СВЯЗИ»

ФАКУЛЬТЕТ ЭЛЕКТОСВЯЗИ

КАФЕДРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ СЕТЕЙ ТЕЛЕКОММУНИКАЦИЙ

Написание программ с использованием пользовательских классов

Курсовая работа

по учебному предмету

«ОБЪЕКТИВНО-ОРИЕНТИРОВАННОЕ ПРОШРАММИРОВАНИЕ»

|  |  |
| --- | --- |
| Выполнила учащийся гр. ТП312 | Рудский Г.В. |
| Руководители | Малолеткин А.Б. |

Минск 2025

ВВЕДЕНИЕ

1 ТЕОРИТИЧЕСКИЕ ОБОСНОВАНИЯ

Введение:

Теоретические обоснования:

1. Классы и методы

Класс является фундаментальным понятием объектно-ориентированного программирования, представляющим собой абстрактный тип данных, который инкапсулирует состояние (в виде полей или свойств) и поведение (в виде методов). Класс служит шаблоном или blueprint для создания объектов - конкретных экземпляров этого класса. С точки зрения философии ООП, класс представляет собой абстракцию реальной сущности, моделируя её ключевые характеристики и функциональность.

Методы представляют собой функции, ассоциированные с классом, которые определяют поведение объектов данного класса. Они могут быть классифицированы на методы-аксессоры (получение и установка значений полей), методы-мутаторы (изменяющие состояние объекта) и методы, реализующие бизнес-логику. Важным аспектом методов является их способность работать с данными конкретного объекта через неявную ссылку this/self.

2. Агрегация и композиция

Агрегация представляет собой тип ассоциации между классами, характеризующийся отношениями "часть-целое" с слабой формой владения. При агрегации время жизни составных частей не зависит от времени жизни содержащего их объекта. Это отношение предполагает, что компоненты могут существовать независимо и могут быть разделены между несколькими объектами. Агрегация обычно реализуется через ссылки или указатели.

Композиция является более строгой формой отношения "часть-целое", предполагающей исключительное владение и сильную зависимость времени жизни. При композиции составные части не могут существовать независимо от содержащего их объекта - они создаются и уничтожаются вместе с ним. Композиция реализуется через непосредственное включение объектов и обеспечивает более тесную связь между компонентами.

3. Перегрузка операций

Перегрузка операций - это механизм, позволяющий определять пользовательскую реализацию для стандартных операторов языка программирования при работе с объектами пользовательских типов. Этот механизм обеспечивает возможность использования привычного синтаксиса для пользовательских классов, что повышает читаемость и естественность кода. Перегрузка операций основывается на принципе ad hoc полиморфизма, где один и тот же оператор может иметь различную семантику в зависимости от типов операндов.

4. Наследование

Наследование является механизмом, позволяющим создавать новые классы на основе существующих, заимствуя и расширяя их функциональность. Это отношение "is-a" (является) устанавливает иерархическую связь между классами, где производный класс наследует атрибуты и методы базового класса. Наследование поддерживает принцип иерархической классификации и способствует созданию логически организованных систем классов. Различают одиночное и множественное наследование, каждый из которых имеет свои преимущества и ограничения.

5. Полиморфизм

Полиморфизм представляет собой фундаментальную концепцию ООП, позволяющую объектам различных классов обрабатываться через общий интерфейс. Полиморфизм времени выполнения обеспечивает возможность вызова методов производных классов через ссылки или указатели на базовый класс. Эта концепция реализует принцип подстановки Барбары Лисков, где объекты производных классов могут использоваться вместо объектов базовых классов без изменения корректности программы.

6. Виртуальная функция

Виртуальная функция - это функция-член класса, объявленная с спецификатором virtual, вызов которой разрешается динамически во время выполнения программы. Механизм виртуальных функций реализуется через таблицу виртуальных методов (vtable), которая содержит указатели на реализации виртуальных функций для каждого класса. Чисто виртуальные функции (abstract methods) не имеют реализации в базовом классе и должны быть переопределены в производных классах, делая класс абстрактным и не допуская его инстанцирования.

7. Обработка исключительной ситуации

Обработка исключительных ситуаций - это механизм управления ошибками, позволяющий отделить нормальный поток выполнения программы от обработки ошибочных ситуаций. Исключения представляют собой объекты, содержащие информацию об ошибке, которые "бросаются" (throw) при возникновении исключительной ситуации и "ловятся" (catch) в специальных блоках обработки. Этот механизм обеспечивает структурированную обработку ошибок, предотвращая распространение ошибок через несколько уровней вызовов и улучшая надежность и сопровождаемость кода.

8. Шаблонные функции и классы

Шаблонные функции и классы представляют механизм параметрического полиморфизма, позволяющий создавать обобщенные алгоритмы и структуры данных, независимые от конкретных типов. Шаблоны обеспечивают статический полиморфизм, где конкретизация происходит на этапе компиляции. Шаблонные функции позволяют писать алгоритмы, работающие с произвольными типами, удовлетворяющими определенным концепциям или требованиям. Шаблонные классы позволяют создавать обобщенные контейнеры и структуры данных, параметризованные типами элементов.

Список литературы:

Гамма, Э., Хелм, Р., Джонсон, Р., Влиссидес, Дж. Приемы объектно-ориентированного проектирования. Паттерны проектирования. — СПб.: Питер, 2001. — 368 с.

Страуструп, Б. Язык программирования C++. — М.: Бином-Пресс, 2011. — 1136 с.

Мейер, Б. Объектно-ориентированное конструирование программных систем. — М.: Русская редакция, 2005. — 1192 с.

Лафоре, Р. Объектно-ориентированное программирование в C++. — СПб.: Питер, 2004. — 928 с.

Буч, Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++. — М.: Бином, 1998. — 560 с.

Эккель, Б. Философия C++. Введение в стандартный C++. — СПб.: Питер, 2004. — 572 с.

Александреску, А. Современное проектирование на C++. Обобщенное программирование и паттерны проектирования. — М.: Вильямс, 2008. — 336 с.

Дьюхерст, С. C++. Священные знания. — СПб.: Питер, 2017. — 240 с.

Саттер, Г. Решение сложных задач на C++. — М.: Вильямс, 2005. — 400 с.

Дотч, Ш. Путь C++. Том 1. Основы программирования. — М.: ДМК Пресс, 2021. — 498 с.

ISO/IEC 14882:2020 Programming languages — C++. — International Organization for Standardization, 2020.

cplusplus.com The C++ Resources Network. — URL: https://www.cplusplus.com/

CppReference.com — URL: https://en.cppreference.com/

Microsoft Docs C++ documentation. — URL: https://docs.microsoft.com/en-us/cpp/

GCC GNU Compiler Collection C++ Standard Library Implementation. — URL: https://gcc.gnu.org/