ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Лабораторная работа

Наследование. Виртуальные функции. Полиморфизм

Выполнил студент группы РИС-23-3Б

Мазунин М.А.

Проверила доцент кафедры ИТАС

О. А. Полякова

2024 г.

**1. Постановка задачи**

1. Определить пользовательский класс.

2. Определить в классе следующие конструкторы: без параметров, с параметрами, копирования.

3. Определить в классе деструктор.

4. Определить в классе компоненты-функции для просмотра и установки полей данных (селекторы и модификаторы).

5. Перегрузить операцию присваивания.

6. Перегрузить операции ввода и вывода объектов с помощью потоков.

7. Определить производный класс.

8. Написать программу, в которой продемонстрировать создание объектов и работу всех перегруженных операций.

9. Реализовать функции, получающие и возвращающие объект базового класса. Продемонстрировать принцип подстановки.

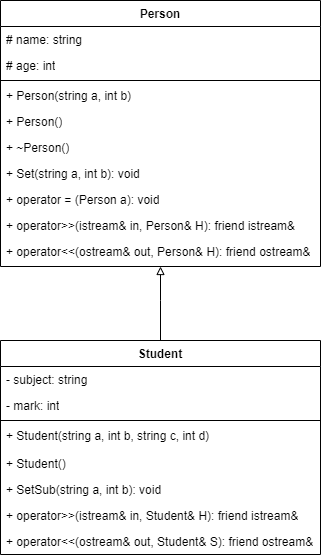
**2. Анализ задачи**

Класс Person - Класс, содержащий информацию о человеке

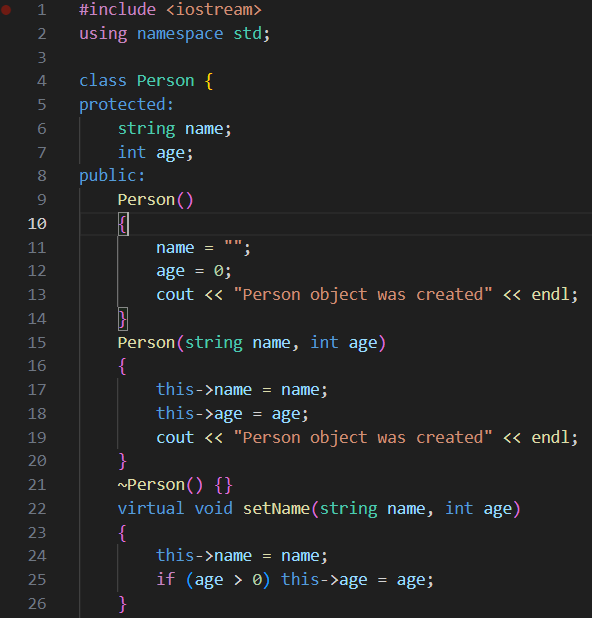
Класс Student - Класс, наследуемый от Person, содержащий дополнительно информацию об успеваемости ученика

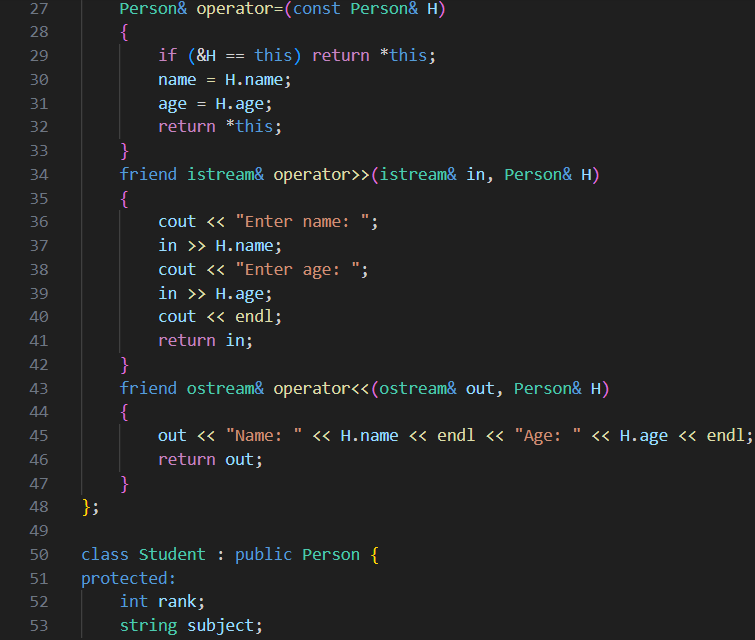
**3. Разбор работы алгоритма**

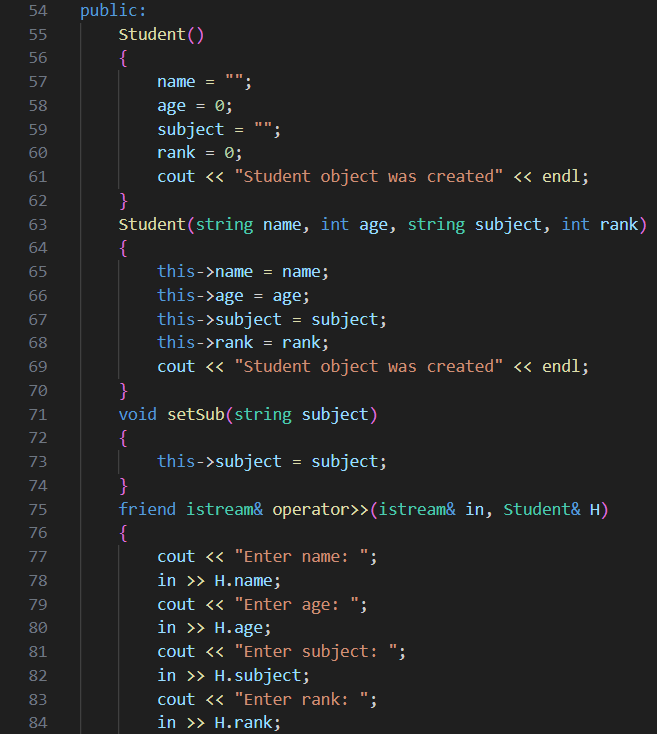
UML диаграмма

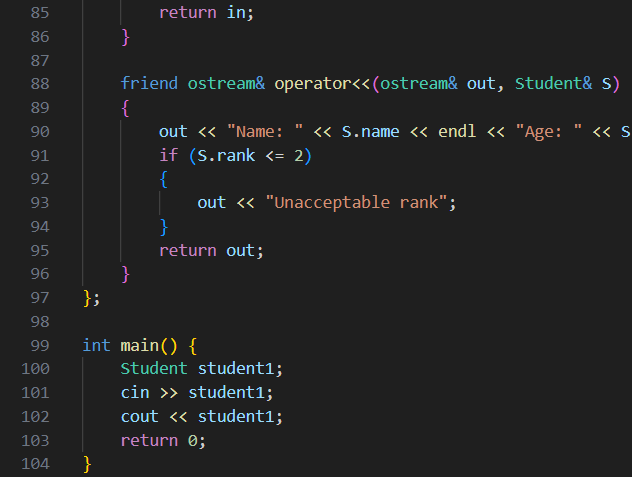


Код программы

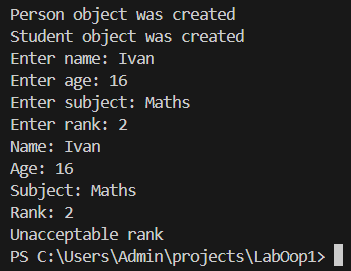








**4. Результаты работы программы**



**5. Контрольные вопросы**

1. Наследование используется для расширения функционала класса-родителя с сохранением обратной совместимости.

2. Наследуются.

3. Наследуются.

4. Не наследуются.

5. Производный класс описывается как обычный класс, с тем различием, что после его названия через двоеточие указываются класс(ы)-родитель(и) со спецификаторами доступа.

6. Конструкторы не наследуются.

7. Деструкторы также не наследуются.

8. Сначала родительский класс, затем производный.

9. Сначала производный класс, затем родительский.

10. Виртуальные функции и механизм позднего связывания – это способ поддерживать общий интерфейс, повысить гибкость наследования.

11. Конструкторы не могут быть виртуальными. Виртуальные деструкторы обязательно должны быть в абстрактном классе.

12. Спецификатор virtual наследуется.

13. Открытое наследование позволяет соблюдать принцип подстановки.

14. Закрытое наследование — это наследование реализации, при котором не соблюдается принцип подстановки.

15. Принцип подстановки состоит в том, что любой наследник класса может использоваться вместо его родителя.

16. stage, post, name, age.

17. Student(){ name = “”;}

Employee(){post = ””; name = “”;}

Teacher(){stage = 0; post=””; name=””;}

18. Student(string name){ name = name;}

Employee(string post, string name){post = post; name=name;}

Teacher(int stage, string post, string name){stage = stage; post = post; name = name;}

19.

Student(Student& S){name = S.name;}

Employee(Employee& E) { name = E.name; post = E.post; }

Teacher(Teacher& T) { name = T.name; post = T.post; stage = T.stage;}

20.

Operator=(Student& S) {name = S.name;}

Operator=(Employee& E) { name = E.name; post = E.post; }

Operator=(Teacher& T) { name = T.name; post = T.post; stage = T.stage;}

**6. Вывод**

Алгоритм выполняет поставленную задачу.

Данный проект можно найти на GitHub



