МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ государственное БЮДЖЕТНОЕ

образовательное учреждение

высшего образования

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра защиты информации



**ЛАБОРАТОРНАЯ РАБОТА №5**

«Многопоточность»

**по дисциплине: « Программирование »**

|  |  |
| --- | --- |
| Выполнил:  *Студент гр. «АБ-321», «АВТФ»*  *Морозов Даниил Иванович*  «21» февраля 2025г  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | Проверил:  *Ассистент кафедры ЗИ*  *Исаев Глеб Андреевич*  «\_\_\_» \_\_\_\_\_\_ 2024г  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |

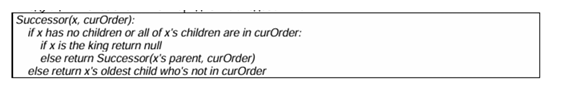
Новосибирск 2024

****Задание к работе:****

**Написать программу с использованием принципов ООП и паттернов программирования.**

****Задание 11.****

**Королевство состоит из короля, его детей, внуков и так далее. Время от времени кто-то из членов семьи умирает или рождается ребенок. В королевстве существует четко определенный порядок наследования, в котором король является первым членом. Определим рекурсивную функцию Successor(x, curOrder), которая, учитывая человека x и порядок наследования, возвращает, кто должен быть следующим человеком x в порядке наследования.**

**Рис.1. Функция Successor.**

**Например, предположим, что у нас есть королевство, состоящее из короля, его детей Alice и Bob (Alice старше Bob) и, наконец, сына Alice - Jack. В начале curOrder будет ["king"]. Вызов Successor(king, curOrder) вернет Алису, поэтому мы добавляем к curOrder, чтобы получить ["king", "Alice"]. Вызов Successor(Alice, curOrder) вернет Jack, поэтому мы добавляем к curOrder, чтобы получить ["king", "Alice", "Jack"]. Вызов Successor(Jack, curOrder) вернет Боба, поэтому мы добавляем к curOrder, чтобы получить ["king", "Alice", "Jack", "Bob"]. Вызов Successor(Bob, curOrder) вернется null. Таким образом, порядок наследования будет таким ["king", "Alice", "Jack", "Bob"]. Используя вышеуказанную функцию, мы всегда можем получить уникальный порядок наследования.**

**Необходимо реализовать ThroneInheritance класс:**

**1. Конструктор ThroneInheritance(string kingName) инициализирует объект класса ThroneInheritance. Имя короля указано в конструкторе.**

**2. void birth(string parentName, string childName) указывает на то, что parentName является родителем childName.**

**3. void death(string name) Указывает на смерть name. Смерть человека не влияет ни на Successor функцию, ни на текущий порядок наследования.**

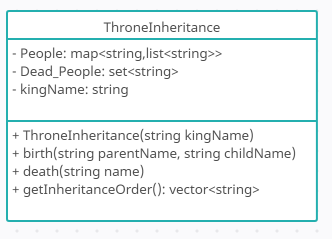
**4. string[] getInheritanceOrder() возвращает список, представляющий текущий порядок наследования, исключая умерших людей.**

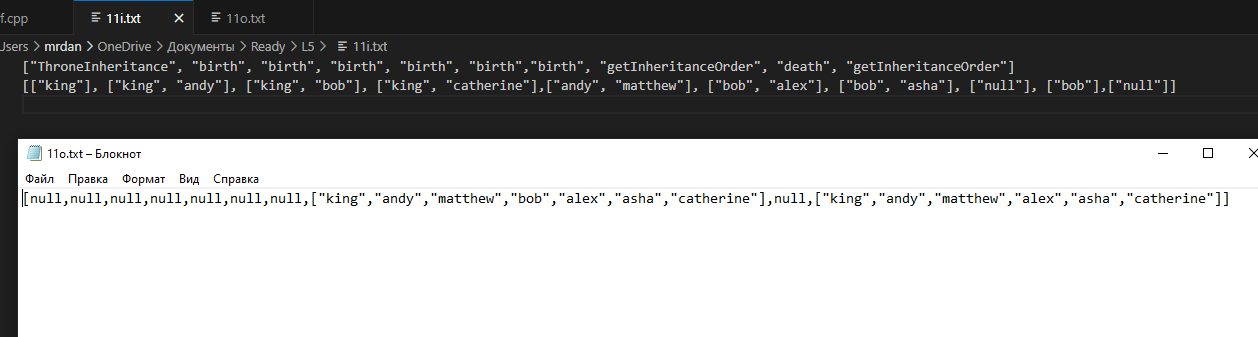
**Ввод: ["ThroneInheritance", "birth", "birth", "birth", "birth", "birth", "birth", "getInheritanceOrder", "death", "getInheritanceOrder"] [["king"], ["king", "andy"], ["king", "bob"], ["king", "catherine"], ["andy", "matthew"], ["bob", "alex"], ["bob", "asha"], [null], ["bob"], [null]]**

**Вывод: [null, null, null, null, null, null, null, ["king", "andy", "matthew", "bob", "alex", "asha", "catherine"], null, ["king", "andy", "matthew", "alex", "asha", "catherine"]]**

**В моей программе задание реализовано в точности, имеет место инкапсуляция и абстракция над способом хранимых данных. Предоставляемый классом интерфейс скрывает всевдодревовидную структуру от пользователя, позволяет влиять на дерево и получать текущий порядок наследования. Существует две вариации — 11c (консольный вариант) и 11f – файловый вариант.**

**Программа доступна по ссылке** <https://github.com/Mirmordan/S3L3-6>

**Рис.2. UML класса.**

**Рис.3. Вывод программы.**

**Вывод**

**В ходе выполнения лабораторной работы я ознакомился с принципами ООП и частью паттернов проектирования. Данные знания позволяют более эффективно писать легко поддерживаемый и понимаемый, читаемый код.**