|  |  |  |
| --- | --- | --- |
| **Министерство науки и высшего образования РФ**  Федеральное государственное автономное образовательное учреждение высшего образования  «Пермский государственный национальный исследовательский университет» | | |
|  | Институт компьютерных наук и технологий | |
| **ОТЧЁТ**  по лабораторной работе №5  по дисциплине «Языки программирования»  Вариант 4 | | |
|  | | Работу выполнил  студент группы ПМИ-7,8-2023 2 курса  Семенов И.Н.  «\_\_\_» \_\_\_\_\_\_\_\_2024 г. |
| Работу проверил  Ракина В.Д  «\_\_\_» \_\_\_\_\_\_\_\_\_ 2024 г. |
| Пермь 2024 | | |

**Задание 1**

**Текст задания**

В класс Дробь, добавить интерфейс на два метода: получение вещественного значения, установка числителя и установка знаменателя.

Сгенерировать такую версию дроби, которая будет кэшировать вычисление вещественного значения. .

**Алгоритм решения**

Создаем анонимный класс дроби, с 2 доп полями - самим кэшированным числом и флагом, который определяет наличие/актуальность кэша, в геттере проверяем флаг, если у нас есть значение в кэше - берем из кэша, иначе - получаем число путем обращения к геттеру супер-класса и устанавливаем флаг на true В сеттерах всегда устанавливаем флаг на false, т.к значение в кэше теряет актуальность..

**Тестирование**

**Код программы**

<https://github.com/SemenovIlya12/Jab/tree/Lab6>

**Задание 2**

**Текст задания**

Необходимо воспользоваться классом Кот и методом принимающим всех мяукающих из задачи 2.5.4.

Необходимо таким образом передать кота в указанный метод, что бы после окончания его работы

узнать сколько раз мяукал кот за время его работы. На рисунке показан пример работы. Перед вызовом

метода создаем кота, отправляем ссылку на кота в метод, после окончания его работы выводим

количество мяуканий на экран. Кота изменять нельзя.

.**Алгоритм решения**

Создаем анонимный класс кота, добавляем поле количества мяуканий, переопределяем метод мяу(), в нем мы инкрементируем счетчик и вызываем мяу() суперкласса, переопределяем метод toString().

**Тестирование**

**Код программы**

Ссылка на githab

**Задание 3**

Составить программу, которая вставляет в список L за первым вхождением элемента E все элементы списка L, если E входит в L.

**Алгоритм решения**

**Проверяем есть ли элемент в списке, если нет - кидаем ошибку.**

**Находим индекс первого вхождения элемента, создаем новый массив. Добавляем в массив элементы от 0 до индекса нашего элемента, потом добавляем все элементы массива после индекса, добавляем оставшиеся элементы списка (от index+1 до arr.size() )**

**Тестирование**

**Код программы**

Ссылка на githab

**Задание 4**

**Текст задания**

В некотором вузе абитуриенты проходят предварительное тестирование, по результатам которого могут быть допущены к сдаче вступительных экзаменов в первом потоке. Тестирование проводится по двум предметам, по каждому предмету абитуриент может набрать от 0 до 100 баллов. При этом к сдаче экзаменов в первом потоке допускаются абитуриенты,набравшие по результатам тестирования не менее 30 баллов по каждому из двух предметов. На вход программы подаются сведения о результатах предварительного тестирования. Известно, что общее количество участников тестирования не превосходит 500.В первой строке вводится количество абитуриентов, принимавших участие в тестировании, N. Далее следуют N строк, имеющих следующий формат:

<Фамилия><Имя><Баллы>

**Алгоритм решения**

Считываем строки в файле, проходимся по строкам, разделяем по пробелу - получаем массив из 4 строк: Фамилия-Имя-Балл1-Балл2 для каждой строки(студента), используем мапу <String, int[]>, ключ - Фамилия + имя, значение - вектор {бал1, бал2}, добавляем в мапу. Получаем множество ключей нашей мапы, проходимся по этим значениям (по каждому студенту - Фамилии + Имя), получаем значение, убеждаемся что один из баллов ниже 30.

**Тестирование**

**Код программы**

Ссылка на githab

**Задание 5**

**Текст задания**

Файл содержит текст на русском языке. Напечатать в алфавитном порядке все глухие согласные буквы, которые входят в каждое нечетное слово.

**Алгоритм решения**

Используем множество. Текст может быть в несколько строк, так что просто пройтись по строке с шагом 2 не получится. Создаем флаг, который определяет, было ли последнее слово предыдущей строки четной или нечетной(чет = false). Создаем цикл while, пока у нас не кончатся строки - считываем строку, избавляемся от ненужных символов, разделяем строку по пробелу, получаем массив строк,проходим по массиву, определяем начало цикла с помощью тернарного оператора (false - 0, true -1 ), с шагом 2, добавляем каждый нечетный элемент. После, проверяем длину массива, если остаток при делении на 2 - значит последнее слово четное, флаг - false. Создаем массив с каждой глухой буквой. Используем TreeSet, он поддерживает множество в натуральном порядке (если объект имплементирует интерфейс comparable),

**Тестирование**

**Код программы**