|  |  |  |
| --- | --- | --- |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное автономное  образовательное учреждение высшего образования  «Пермский государственный национальный  исследовательский университет» | | |
|  | Институт компьютерных наук и технологий | |
| **ОТЧЁТ**  по индивидуальной работе №2  по дисциплине «Язык программирования Python»  Вариант 1 | | |
|  | | Работу выполнил  студент группы ИКНТ-ИТ-7,  2024-2025 1 курса  Вавилина Анастасия Юрьевна  «\_\_\_» \_\_\_\_\_\_\_\_2024 г. |
| Работу проверил  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Фамилия И.О.  «\_\_\_» \_\_\_\_\_\_\_\_\_ 2024 г. |
| Пермь 2025 | | |

СОДЕРЖАНИЕ

[Постановка задачи 3](#_Toc153130027)

[Алгоритм решения 3](#_Toc153130028)

[Тестирование 3](#_Toc153130029)

[Код программы 3](#_Toc153130030)

[Инструкция по применению стилей и оформлению работы 4](#_Toc153130031)

# Постановка задачи

Реализуйте с помощью двусвязного циклического списка следующую игру: Ученики школы встают в круг. Один ученик пишет программу, генерирующую случайное целое число из промежутка [–10;10]. Если выпало положительное число, то отсчет ведется «по часовой стрелке», если отрицательное, то «против часовой». Ученик, на котором остановился счет, не выбывает из круга, а делает «доброе дело» и уровень его рейтинга увеличивается на 1. Следующий отсчет начинается с ученика, стоящего в круге рядом с тем, который только что делал «доброе дело», с правого (если отсчет ведется по часовой стрелке) или с левого (против часовой). После того, как игра всем надоела, нужно вывести список учеников в порядке не возрастания рейтинга. Если имеется несколько учеников с одинаковым рейтингом, то их нужно выводить в том порядке, в котором они поступили в список. Исходный список фамилий учеников для игры находится в текстовом файле. В начале игры рейтинг каждого ученика равен 0. Во время игры нужно выводить протокол: какое случайное число выпало, какой ученик делал доброе дело и рейтинг ученика после совершения доброго дела. Количество раундов игры вводится пользователем с клавиатуры.

# Алгоритм решения

1. Создание структуры данных.

Класс StudentNode (хранение данных между узлами):

1. name(str) – фамилия участника.
2. rating(int) – текущий рейтинг(начинается с 0).
3. prev(StudentNode) – ссылка на предыдущий узел.
4. next(StudentNode) – ссылка на следующий узел.
5. Реализация двусвязного циклического списка.

Класс StudentCircle (управление кругом учеников):

1. add\_student(self, name) – добавляет нового ученика в круг.
   1. Создаёт новый узел.
   2. Если круг пустой делает узел головой и замыкает ссылки на себя.
   3. Иначе вставляет узел перед головой, сохраняя цикличность.
2. find\_student(self, name).
   1. Ищет ученика по имени (без учёта регистра).
   2. Линейный обход круга с проверкой node.name.lower() == name.lower().
3. get\_student\_steps(self, start\_node, steps) - перемещается на steps узлов от start\_node.
   1. Определяет направление (знак steps).
   2. Делает abs(steps) % size шагов, чтобы избежать лишних оборотов.
4. print\_circle(self) - выводит текущее состояние круга.
   1. Проверяет не пуст ли круг.
   2. Начиная с головы, выводит каждого ученика.
   3. Завершает обход при возврате к голове.
5. Чтение данных и настройка игры.

Функция read\_students\_from\_file(filename) - загружает список учеников из файла.

1. Открывает файл, читает строки, пропускает пустые.
2. Обрабатывает ошибки (например, отсутствие файла).
3. Игровой процесс.

Функция main()

1. Инициализация
   1. Чтение файла: создание круга (StudentCircle).
   2. Выбор начального ученика (select\_starting\_student).
2. Раунды (для каждого раунда):
   1. Запрос числа (get\_number\_input).
   2. Поиск ученика (get\_student\_steps).
   3. Обновление рейтинга.
3. Итоги
   1. Сортировка (get\_sorted\_list).
   2. Вывод результатов.
4. Сортировка и вывод.

Метод get\_sorted\_list():

1. Собирает список кортежей (name, rating, original\_index).
2. Сортирует по убыванию rating, затем по original\_index.

# Тестирование

# Код программы

Код

# Инструкция по применению стилей и оформлению работы

Для оформления частей отчёта следует использовать заранее созданные стили. Все стили, которые могут пригодиться, начинаются с «ЛР. 1 …».



**ЛР. 1 Обычный** – для оформления текста задания и алгоритма решения.

**ЛР. 1 Текст программы** – для оформления кода программы.

**ЛР. 1 Заголовок 1** – заголовок первого уровня (для того, чтобы озаглавить основные разделы отчета).

**ЛР. 1 Заголовок 2** – заголовок второго уровня (для того, чтобы озаглавить подразделы).

Для того, чтобы перенести текст следующего блока на другую страницу, необходимо воспользоваться инструментов «Разрыв страницы» в разделе «Вставка».

