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 3

Работу выполнил
студент группы ИТ-3,4-2024 1 курса
Сайранов Эльдар Равилевич
«02» июня 2025 г.

Работу проверил
_____ Фамилия И.О.
«___» _____ 2025 г.

Пермь 2025

СОДЕРЖАНИЕ

Постановка задачи

Напечатать в порядке возрастания первых n натуральных чисел, в разложение которых на простые множители входят только числа 2,3,5. Идея решения: введем три очереди x2, x3, x5 в которых будем хранить элементы, которые соответственно в 2, 3, 5 раз больше напечатанных, но еще не напечатаны. Рассмотрим наименьший из ненапечатанных элементов: пусть это x. Тогда он делится нацело на одно из чисел 2, 3, 5; x находится в одной из очередей и является в ней первым элементом (меньшие его уже напечатаны, а элементы очередей не напечатаны). Напечатав x, нужно изъять его из очереди и добавить в очередь кратные ему элементы. Длины очередей не превосходят числа напечатанных элементов. Изначально в очередях хранится по одному числу.

Цель проекта — разработать программу, генерирующую последовательность чисел, все простые множители которых принадлежат множеству {2, 3, 5}. Эти числа называются «уродливыми».

Пользователь должен иметь возможность ввести количество требуемых чисел n, и программа должна вернуть первые n таких чисел в порядке возрастания.

Также необходимо реализовать юнит-тесты, проверяющие корректность генерации, отсутствие дубликатов и корректную обработку невалидного ввода.

Алгоритм решения

Было реализовано классовое решение, основанное на использовании трех очередей (queue2, queue3, queue5), в которые по мере генерации добавляются новые кандидаты на следующие “уродливые” числа.

Основная идея алгоритма

- Начать с числа 1
- Для каждого текущего числа x:
    - Добавить x*2 в очередь queue2, x*3 — в queue3, x*5 — в queue5
- Выбирать наименьшее из голов трех очередей, добавлять его в результат, и при этом удалять одинаковые минимумы (если совпадают)

Таким образом все числа формируются в правильном порядке и без повторений

Выбранные структуры и типы данных

1. **Очереди** — по условию
2. **Список** — используется для хранения результатов — уже сгенерированных «уродливых» чисел, которые возвращаются пользователю. Список сохраняет порядок добавления, удобен для индексации и вывода результата

Особенности реализации

- Для каждого вызова генерации очереди сбрасываются, чтобы избежать накопления старых данных
- Используется защита от нецелочисленного и некорректного ввода через raise ValueError

Тестирование

Для проверки корректности работы программы были написаны юнит-тесты с использованием библиотеки unittest.

Проверка первых значений

Убедиться, что get_numbers(n) возвращает правильные первые n «уродливых» чисел, например:

```
self.assertEqual(self.gen.get_numbers(10), [1, 2, 3, 4, 5, 6, 8, 9, 10, 12])
```

Проверка на дубликаты

```
nums = self.gen.get_numbers(50)
self.assertEqual(len(nums), len(set(nums)))
```

Проверка на ошибки при некорректном вводе

```
with self.assertRaises(ValueError):
    self.gen.get_numbers(-3)
```

Результат тестирования: все тесты выполнены успешно

Код программы

Код программы доступен на GitHub по ссылке.