Министерство образования и науки Российской Федерации Федеральное агентство по образованию Федеральное государственное бюджетное образовательное учреждение высшего образования «Вятский государственный университет»

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Лабораторная работа №6

по курсу «Разработка программных систем»

Выполнил студент группы ИВТ-31\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Птахова А.М/

Проверил доцент кафедры ЭВМ \_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Чистяков Г.А./

Киров 2023

1. Задание

Разработать класс для генерации простых чисел. Класс  должен иметь два публичных метода:

− def getRandomPrime() – возвращает случайное простое число из  диапазона [2, 109];

− def getRandomArray(int length) – возвращает упорядоченный по  возрастанию массив простых чисел размерности length;

− и два внутренних метода:

− def getNext(int prime) – возвращает следующее после prime  простое число из диапазона [2, 109];

− def isPrime(int arg0) – определяет является ли заданный  аргумент простым числом.

2. Листинг кода

**primerandom.py**

from typing import List

\_LAST\_PRIME\_NUMBER = 999999937

def \_pollard\_p1\_test(number: int) -> bool:

from math import gcd, log, floor

b = 13

q = (2, 3, 5, 7, 11, 13)

a = 5 % number

e = lambda b, v: int(floor(log(b, v)))

aa = lambda a, v: int(pow(a, pow(v, e(b, v), number), number))

red = lambda a: a if gcd(a, number) == 1 else ((a \*\* 2) % number + 3) % number

test = lambda a: len([gcd(aa(a, v), number) for v in q if aa(a, v) != 0 and 1 < gcd(aa(a, v), number) < number]) > 0

gen = lambda a, n: [red(a)] + gen(red(a), n - 1) if n > 0 else []

return len([i for i in gen(a, 10) if test(i)]) == 0

def is\_prime(number: int) -> bool:

return number == 2 or \_pollard\_p1\_test(number)

def get\_next(number: int) -> int:

lrec = lambda n: -1 if n == 10 \*\* 9 else n if is\_prime(n) else lrec(n + 1)

return lrec(number + 1)

def get\_random\_number() -> int:

from random import randint

n = randint(2, \_LAST\_PRIME\_NUMBER)

return n if is\_prime(n) else get\_next(n)

def get\_random\_array(length: int) -> List[int]:

return sorted(get\_random\_number() for \_ in range(length))

**d.py**

from typing import Optional

from primerandom import get\_random\_number, get\_random\_array

def read\_int(min\_value: int, max\_value: int) -> Optional[int]:

try:

number = int(input())

if min\_value <= number <= max\_value:

return number

else:

return None

except ValueError:

return None

if \_\_name\_\_ == '\_\_main\_\_':

while True:

print('Выберите действие:')

print('1. Выдать случайное простое число')

print('2. Выдать массив случаных чисел')

print('3. Выйти')

op = read\_int(1, 3)

if op is None:

print('Неизвестная команда')

continue

if op == 1:

print(get\_random\_number())

elif op == 2:

print('Введите длину массива:')

while True:

ln = read\_int(1, 1000000)

if ln is None:

print('Длина массива должна быть натуральным числом в диапазоне (0; 1000000]')

else:

print('\n'.join(str(v) for v in get\_random\_array(ln)))

break

elif op == 3:

break

3. Вывод

В ходе выполнения лабораторной работы были изучены основные конструкции ЯП Python, принципы функционального программирования. В результате был реализован модуль для генерации простых чисел.