МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образование «Белорусский государственный технологический университет»

**«Основы теории чисел и их использование в криптографии»**

Студент: Высоцкий Я.А.

ФИТ 3 курс 5 группа

Вариант 4

Преподаватель: Савельева М. Г.

Минск 2023

**1) Найти все простые числа в интервале [2, *n*]. Подсчитать количество простых чисел в указанном интервале.**

Для нахождения простых чисел было разработана функция   
GetPrimeList (int start, int end). Функция принимает два параметра: начала и конец интервала.

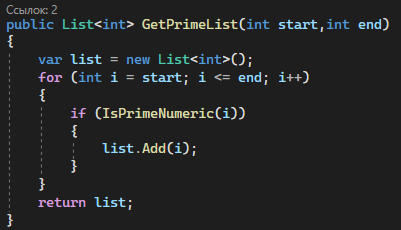


Рисунок 1.1 – Функция нахождения простых чисел на интервале

Для работы данной функции было разработана функция   
IsPrimeNumeric (int numeric). Функция принимает число и возвращает булевскую переменную, которая описывает является ли число простым.

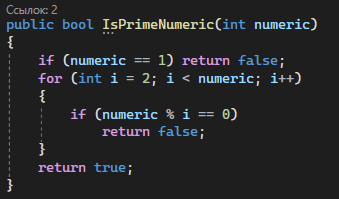


Рисунок 1.2 – Функция проверки числа на натуральность

В итоге выполнения функции подсчета натуральных чисел, результат является следующим:



Рисунок 1.3 – Результат выполнения первого задания

**2) Повторить пункт 1 для интервала [*m*, *n*]. Сравнить полученные результаты с «ручными» вычислениями используя «решето Эратосфена»**

Для нахождения простых чисел на интервале [*m*, *n*] используется таже функции что и ранее: GetPrimeList (int start, int end). В итоге результат выполнения второго задания является следующим:

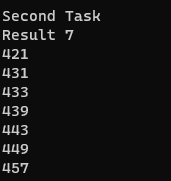


Рисунок 2.1 – Результат выполнения второго задания

**3) Записать числа m и n в виде произведения простых множителей.**

Для записи чисел в канонической форме была разработана функция GetCannonFrom (int numeric). В качество параметра функция принимает число, которое требуется привести в каноническую форму.

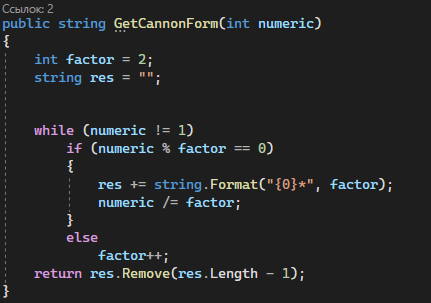


Рисунок 3.1 – Функция приведения числа к канонической форме

В итоге выполнения функции подсчета натуральных чисел, результат является следующим:

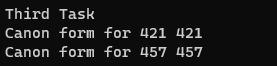


Рисунок 3.2 – Результат выполнения третьего задания

Числа в канонической форме не отличаются от исходной, так как они является простыми.

**4) Проверить, является ли число, состоящее из конкатенации цифр *m* || *n* простым.**

Для выполнения данного задания используется функция, описанная ранее: IsPrimeNumeric (int numeric). В данную функцию передается заранее конкатенированное число.

В итоге выполнения функции проверки числа на натуральность, результат является следующим:

****

Рисунок 4.1 – Результат выполнения четвертого задания

**5) Найти НОД (*m*, *n*).**

Для нахождения наименьшего общего делителя была разработана функция GetNod (int a, int b). В качестве параметров принимает два числа, для которых подсчитывается наименьший общий делитель.

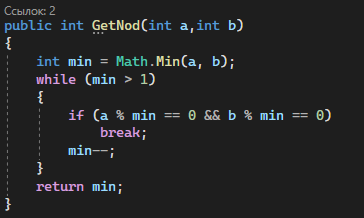


Рисунок 5.1 – Функция подсчета НОД для двух чисел

Так же у данной функции есть перегрузка, принимающая 3 параметра. Она используется для подсчета НОД для трёх чисел

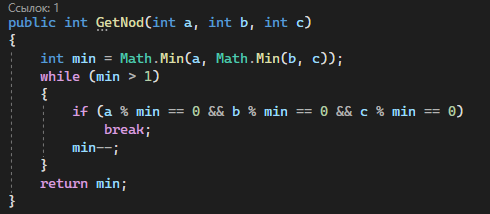


Рисунок 5.2 - Функция подсчета НОД для трёх чисел

В итоге выполнения функции подсчета НОД, результат является следующим:

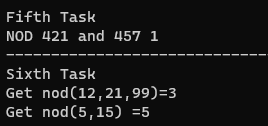


Рисунок 5.3 – Результат выполнения пятого задания

Так как оба числа являются простыми то их НОД равен единице.

**Вывод**

При выполнении лабораторной работы были закреплены знания по высшей арифметике. Также приобретены практические знания в решении задач с использованием простых и взаимно простых чисел, вычислений по правилам модулярной арифметики и нахождению обратных чисел по модулю.