Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики

и радиоэлектроники»

Специальность «Программная инженерия»

Учебная дисциплина «Основы алгоритмизации и программирования»

Отчет

по лабораторной работе №8

«Функции»

Подготовила: Студент гр. 410901

Гуцева Т. Г.

Проверил: Усенко Ф. В.

Минск 2024

Цель: сформировать навыки и умения обработки структурированных типов данных, организованных в виде функций.

Задание 1: Дано натуральное число Р. Напишите программу нахождения всех натуральных чисел, не превосходящих Р, которые можно представить в виде произведения двух простых чисел.

Листинг кода:

#include <iostream>

using namespace std;

int\* sieveOfEratosthenes(int p, int &prime\_count)

{

int\* arr = new int[p + 1];

for (int i = 0; i <= p; i++) arr[i] = i;

for (int i = 2; i \* i <= p; i++)

{

if (arr[i])

{

for (int j = i \* i; j <= p; j += i)

{

arr[j] = 0;

}

}

}

for (int i = 2; i <= p; i++)

{

if (arr[i]) prime\_count++;

}

int\* primes = new int[prime\_count];

int idx = 0;

for (int i = 2; i <= p; i++) {

if (arr[i]) primes[idx++] = arr[i];

}

delete[] arr;

return primes;

}

int\* findSemiprimes(int\* primes, int prime\_count, int p, int& semiprime\_count)

{

int\* semiprimes = new int[p + 1];

for (int i = 0; i < prime\_count; i++)

{

for (int j = i; j < prime\_count; j++)

{

int semiprime = primes[i] \* primes[j];

if (semiprime <= p)

{

semiprimes[semiprime\_count++] = semiprime;

}

else

{

break;

}

}

}

return semiprimes;

}

void sortAndRemoveDuplicates(int\* arr, int& count)

{

for (int i = 0; i < count - 1; i++)

{

for (int j = i + 1; j < count; j++)

{

if (arr[i] > arr[j])

{

int swap = arr[i];

arr[i] = arr[j];

arr[j] = swap;

}

}

}

int unique\_count = 0;

for (int i = 0; i < count; i++)

{

if (i == 0 || arr[i] != arr[i - 1])

{

arr[unique\_count++] = arr[i];

}

}

count = unique\_count;

}

int main()

{

setlocale(LC\_ALL, "ru");

int p;

cout << "Введите P:" << endl;

cin >> p;

cout << endl;

int prime\_count = 0;

int\* primes = sieveOfEratosthenes(p, prime\_count);

cout << "Все простые числа до P:" << endl;

for (int i = 0; i < prime\_count; i++)

{

cout << primes[i] << " ";

}

cout << endl << endl;

int semiprime\_count = 0;

int\* semiprimes = findSemiprimes(primes, prime\_count, p, semiprime\_count);

sortAndRemoveDuplicates(semiprimes, semiprime\_count);

cout << "Все полупростые числа до P:" << endl;

for (int i = 0; i < semiprime\_count; i++)

{

cout << semiprimes[i] << " ";

}

cout << endl;

delete[] primes;

delete[] semiprimes;

return 0;

}

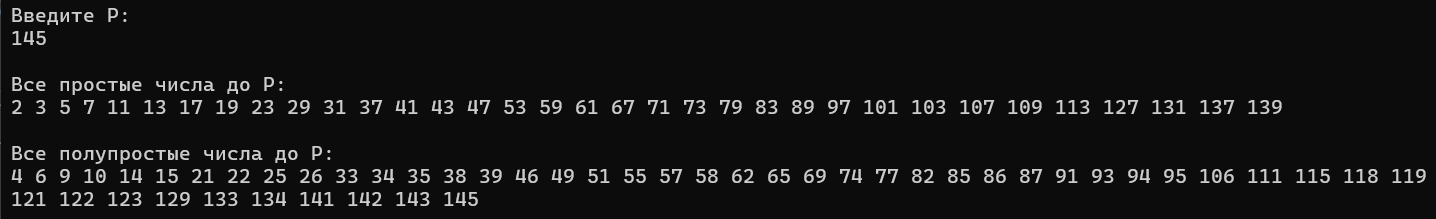
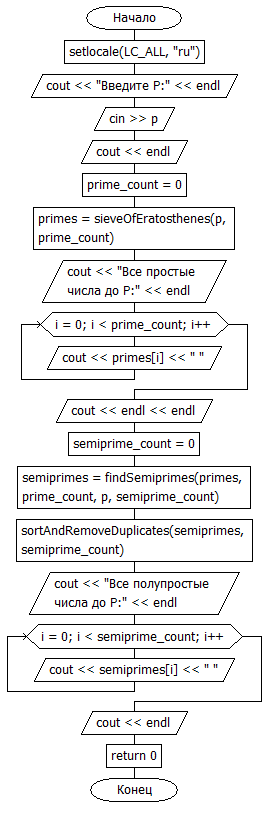


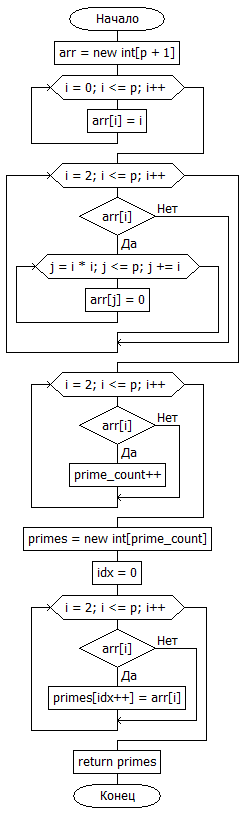
Рисунок 1 – Результат выполнения программы

Построение блок-схемы:

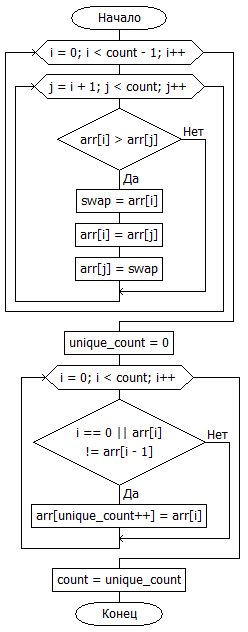
Функция main:



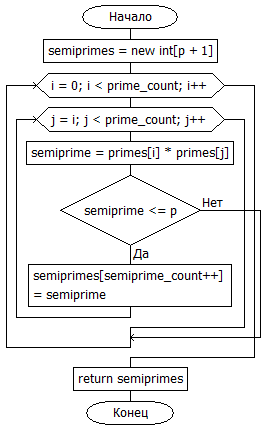
Функция sieveOfEratosthenes:



Функция sortAndRemoveDuplicates:



Функция findSemiprimes:



Задание 2: Используя перегрузку методов, создайте программу:

а) для сложения десятичных дробей;

б) для сложения обыкновенных дробей.

Листинг кода:

#include <iostream>

using namespace std;

float Sum(float a, float b)

{

return a + b;

}

int NOD(int num, int denom)

{

int remain = num;

while (remain = denom % num)

{

denom = num;

num = remain;

}

return num;

}

void Sum(int m1, int n1, int m2, int n2)

{

int m3 = m1 \* n2 + m2 \* n1;

int n3 = n1 \* n2;

int nod = NOD(m3, n3);

m3 /= nod;

n3 /= nod;

cout << m3 << "/" << n3 << endl;

}

int main() {

setlocale(LC\_ALL, "ru");

float a, b;

int m1, n1, m2, n2;

cout << "Задайте две десятичные дроби:" << endl;

cout << "Введите первое число:" << endl;

cin >> a;

cout << "Введите второе число:" << endl;

cin >> b;

cout << "Сумма десятичных дробей:" << endl;

cout << a << " + " << b << " = " << Sum(a, b) << endl;

cout << "Введите числитель и знаменатель первой дроби:" << endl;

cin >> m1 >> n1;

cout << "Введите числитель и знаменатель второй дроби:" << endl;

cin >> m2 >> n2;

cout << "Сумма обыкновенных дробей:" << endl;

cout << m1 << "/" << n1 << " + " << m2 << "/" << n2 << " = ";

Sum(m1, n1, m2, n2);

return 0;

}

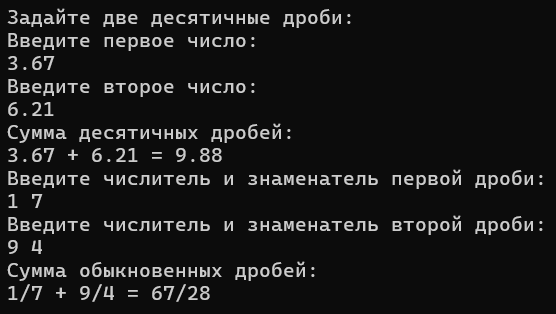
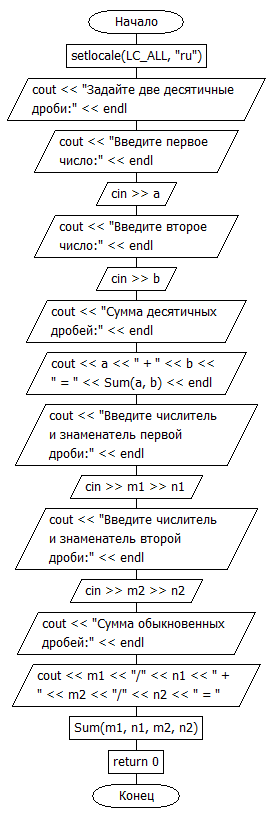


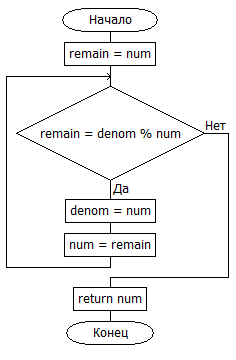
Рисунок 2 – Результат выполнения программы

Построение блок-схемы:

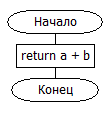
Функция main:



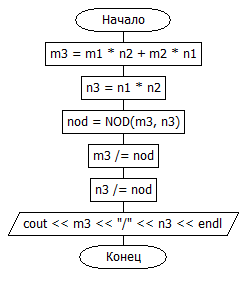
Функция NOD:



Функция Sum (для сложения десятичных дробей):



Функция Sum (для сложения обыкновенных дробей):



Вывод: В ходе выполнения задания успешно созданы две программы. Первая находит все натуральные числа, не превосходящие Р, которые можно представить в виде произведения двух простых чисел Вторая с помощью перегрузки методов считает сумму десятичных или обыкновенных дробей.