Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа № 10

По дисциплине «Основы алгоритмизации и программирования»

На тему «Рекурсивные алгоритмы»

Выполнил:

Студент 1 курса 8 группы

Статько Герман Вячеславович

Преподаватель: асс. Андронова М.В.

Минск, 2024

**В соответствии со своим вариантом выполнить задания из таблицы, представленной ниже. В некоторых заданиях имеются ошибки: не выполняется условие завершения рекурсии. Изменить условие такой задачи с тем, чтобы рекурсия выполнялась.**

**Вариант 12**

****

#include <iostream>

using namespace std;

double summa(double, int, int, int); // Прототип функции для подсчета суммы

int main()

{

setlocale(LC\_CTYPE, "rus");

int n, x;

cout << "Введите n: "; // Ввод n

cin >> n;

cout << "Введите x: "; // Ввод x

cin >> x;

cout << "y = " << summa(0, 1, n, x) << endl;

}

//sum - Сумма косинусов на данный момент

//count - Подсчет итераций на момент вызова

double summa(double sum, int count, int n, int x)

{

sum += cos(count \* x);

//Если count == n, то мы выполнили необходимое количество итераций

if (count == n)

return sum;

//Иначе увеличиваем счетчик и вызываем функцию рекурсивно с новыми параметрами

else

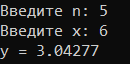
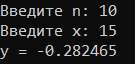
{

count++;

summa(sum, count, n, x);

}

}

**  **

**5. К номеру своего варианта прибавить число 2 и написать программу для новых исходных данных (для вариантов 15, 16 перейти к вариантам 1, 2).**

**Вариант 14**

#include <iostream>

using namespace std;

double func(double, double, double); // Прототип функции для подсчета функции

int main()

{

setlocale(LC\_CTYPE, "rus");

double x, n;

cout << "Введите x: "; // Ввод x

cin >> x;

cout << "Введите n: "; // Ввод n

cin >> n;

cout << "y = " << func(1, n, x);

}

//mul - Произведение на момент конкретного прохода рекурсии

double func(double mul, double n, double x)

{

mul \*= x / n;

if (n == 1.0)

return mul;

else

{

func(mul, n - 1, x);

}

}

**  **

**Доп. Задачи**

**Я не использовал дополнительный массив, так как считаю это не обязательным  
Но если сильно нужно, то я могу переделать**

****

****

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

void find(int, int, string, ofstream&); // Функция для записи и генерации чисел

int main()

{

setlocale(LC\_CTYPE, "rus");

int A;

do

{

cout << "Введите A: ";

cin >> A;

} while (A < 1 || A > 9);

ofstream fout("output.txt"); // Создание файла

if (!fout.is\_open())

{

cout << "Ошибка открытия файла!" << endl;

}

find(A, 0, "", fout);

fout.close();

}

//size - Контроль выхода за пределы

//num - Число

void find(int A, int size, string num, ofstream& fout)

{

if (size == A)

{

fout << num << endl;

}

else

{

//Вход в цикл и вызов рекурсивно функции, а также формирование из итераций чисел

for (int i = 1; i <= A; i++)

{

find(A, size + 1, num + to\_string(i), fout);

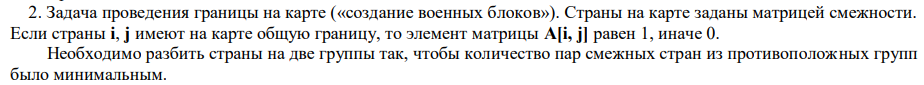
}

}

}

****

****

****

#include <iostream>

// Количество стран в графе

const int MAX\_COUNTRIES = 4;

// Матрица смежности графа

int adjacencyMatrix[MAX\_COUNTRIES][MAX\_COUNTRIES] = {

{0, 1, 0, 1},

{1, 0, 1, 0},

{0, 1, 0, 1},

{1, 0, 1, 0}

};

// Массив для хранения принадлежности стран к группам

int groups[MAX\_COUNTRIES];

// Рекурсивная функция обхода графа в глубину

void dfs(int node, int group) {

groups[node] = group;

for (int i = 0; i < MAX\_COUNTRIES; ++i) {

if (adjacencyMatrix[node][i] == 1 && groups[i] == -1) {

dfs(i, 1 - group);

}

}

}

int main() {

setlocale(LC\_CTYPE, "rus");

// Инициализация массива принадлежности к группам

for (int i = 0; i < MAX\_COUNTRIES; ++i) {

groups[i] = -1;

}

// Обход графа и разделение стран на группы

for (int i = 0; i < MAX\_COUNTRIES; ++i) {

if (groups[i] == -1) {

dfs(i, 0);

}

}

// Вывод информации о принадлежности стран к группам

std::cout << "Группы стран:" << std::endl;

for (int i = 0; i < MAX\_COUNTRIES; ++i) {

if (groups[i] == 0) {

std::cout << "Страна " << i + 1 << ": Группа 1" << std::endl;

}

else {

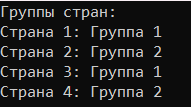
std::cout << "Страна " << i + 1 << ": Группа 2" << std::endl;

}

}

return 0;

}

****

****

#include <iostream>

using namespace std;

const int n = 5; // Количество чисел

int numbers[n] = { 1, 2, 3, 4, 5 }; // Массив чисел

int permutation[n]; // Массив для хранения текущей перестановки

bool used[n]; // Массив для отслеживания использованных чисел

void generatePermutations(int index) {

if (index == n) {

for (int i = 0; i < n; ++i) {

cout << permutation[i] << " ";

}

cout << endl;

}

else {

for (int i = 0; i < n; ++i) {

if (!used[i]) {

used[i] = true;

permutation[index] = numbers[i];

generatePermutations(index + 1);

used[i] = false;

}

}

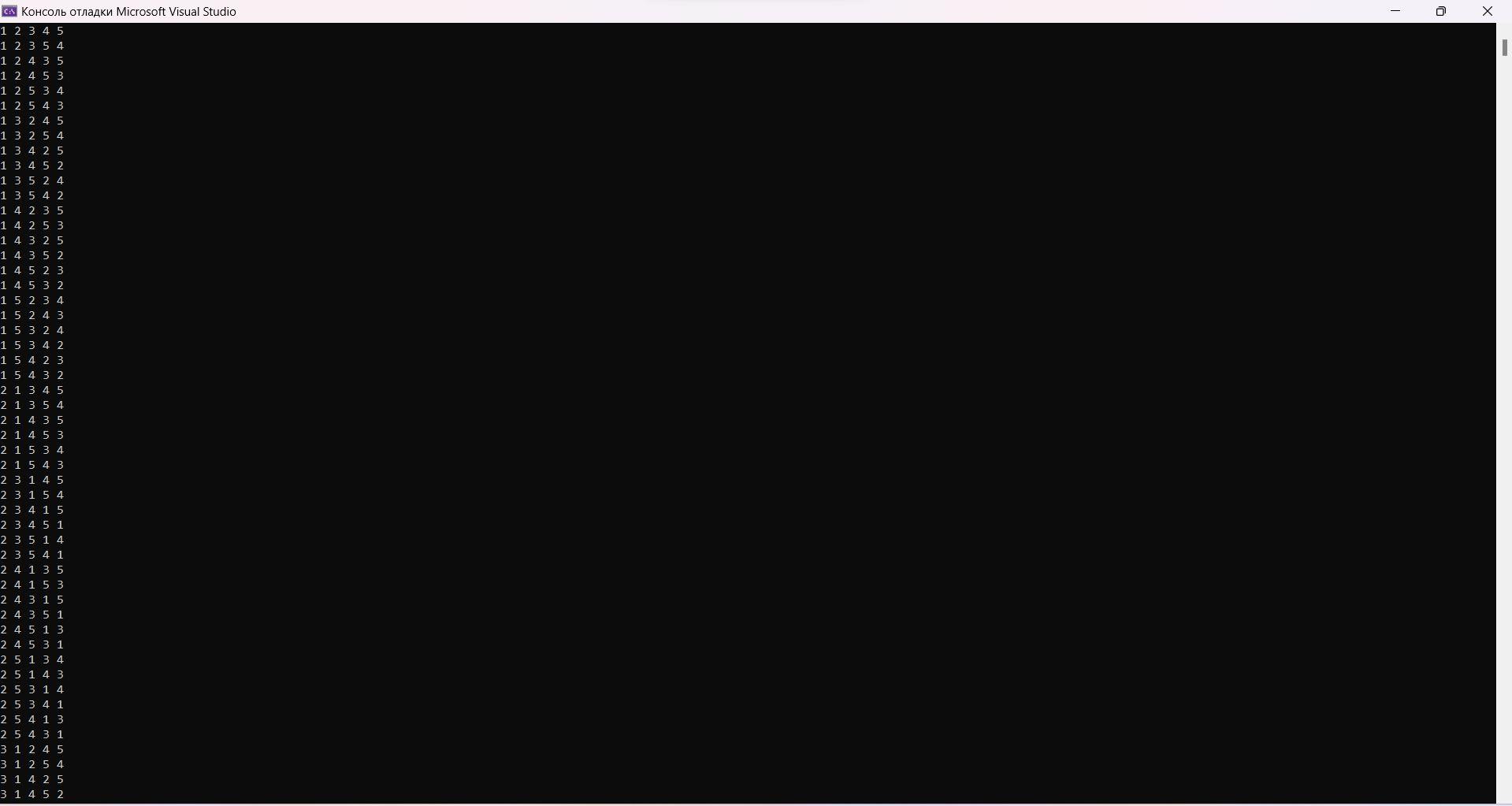
}

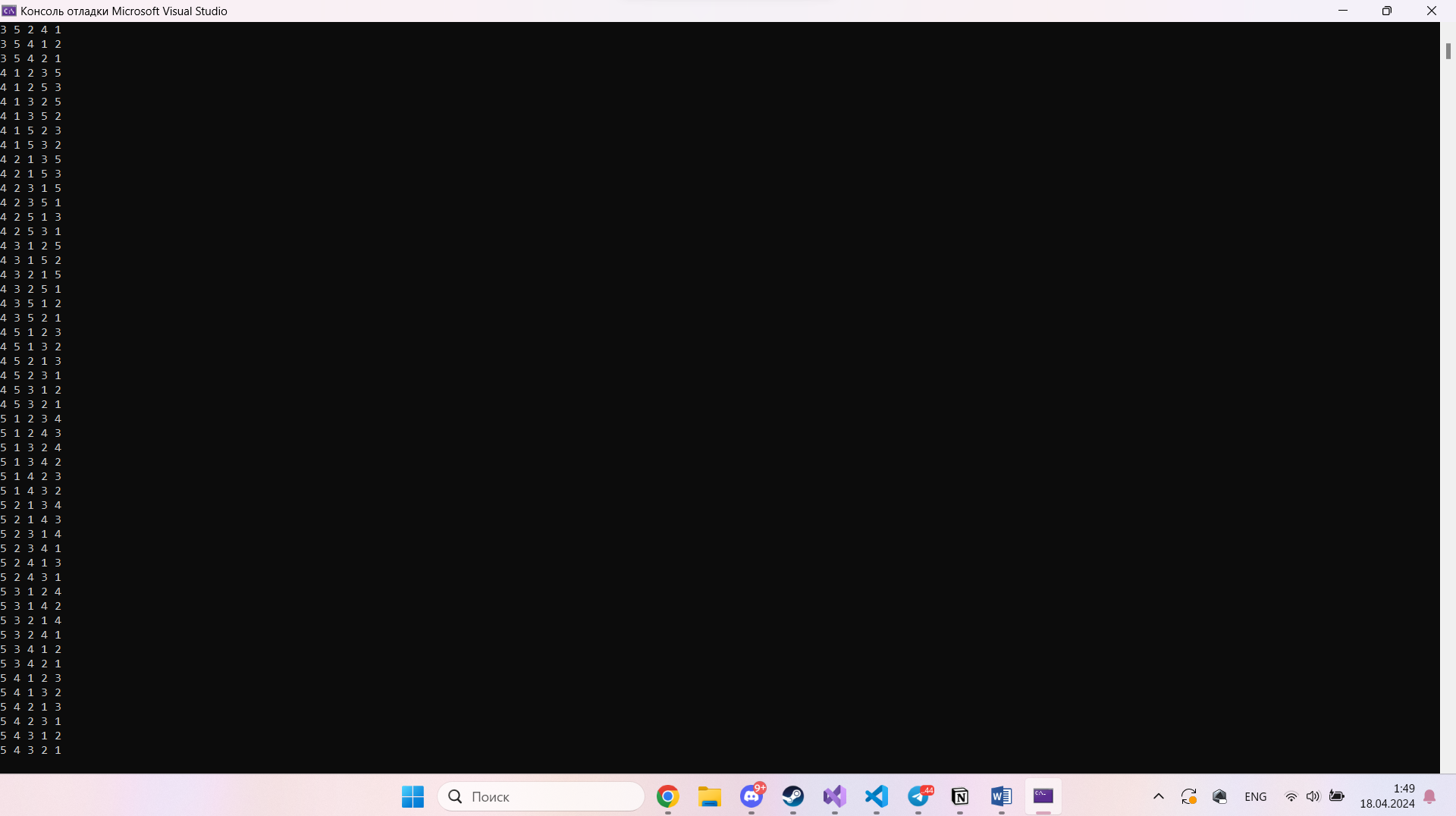
}

int main() {

generatePermutations(0);

}

****

****