Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа №10

По дисциплине «Основы алгоритмизации и программирования»

На тему «Рекурсивные алгоритмы»

Выполнил(а):

Студент(ка) 1 курса 8 группы

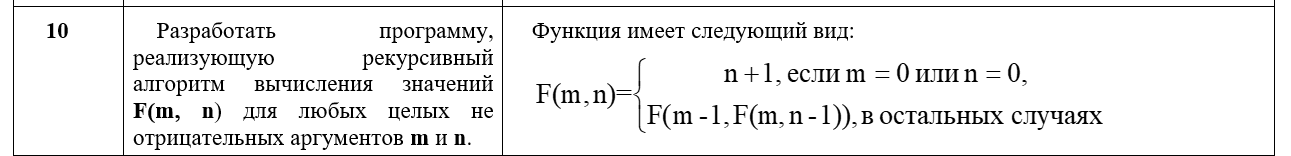
Семёнов Даниил Вячеславович

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 10



#include <iostream>

using namespace std;

int func(int m, int n) //рекурсивная функция

{

if (n == 0 || m == 0) // если выполняется условие

{

return n + 1; // возвращаем

}

else // иначе

{

return func(m - 1, func(m,n-1)); //возвращаем

}

}

int main()

{

int F = func(1, 2); //присваиваем значение

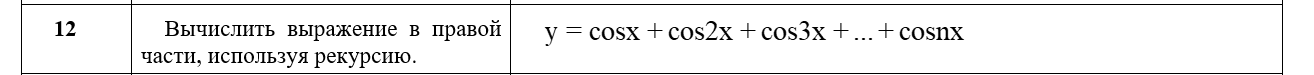
cout << F; //выводим

return 0;

}



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



#include <iostream>

#include <cmath>

using namespace std;

double func(int n, double x) // рекурсивная функция

{

if (n == 0) // если выполняется условие

{

return 0; // возвращаем

}

else //иначе

{

return cos(n \* x) + func(n - 1, x); //возвращаем

}

}

int main()

{

double y = func(1, 3.14); //присваиваем

cout << y; //выводим

return 0;

}



Доп. Задачи

3



#include <iostream>

using namespace std;

void swap(int& a, int& b) // меняем местами значение

{

int temp = a;

a = b;

b = temp;

}

void perestanovka(int nums[], int st, int en) //рекурсивная функция для обмена

{

if (st == en) //если дошли до конца массива

{

for (int i = 0; i <= en; i++) //выводим строку

{

cout << nums[i] << " ";

}

cout << endl;

}

else //иначе

{

for (int i = st; i <= en; i++) //пока не дойдём до конца массива

{

swap(nums[st], nums[i]); //меняем местами

perestanovka(nums, st + 1, en); //рекурсивно вызываем функцию для генерации перестановок подмассива

swap(nums[st], nums[i]); // возвращаем на место исходный порядок

}

}

}

int main()

{

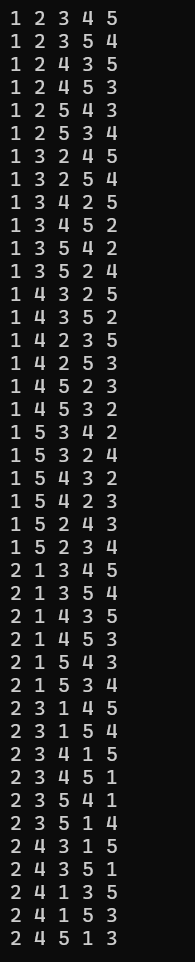
int n = 5;

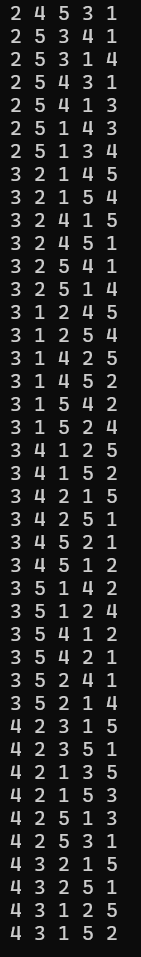
int numbers[] = { 1, 2, 3, 4, 5 };

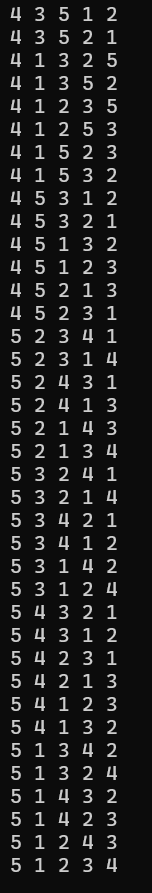
perestanovka(numbers, 0, n - 1); //вызываем функцию

return 0;

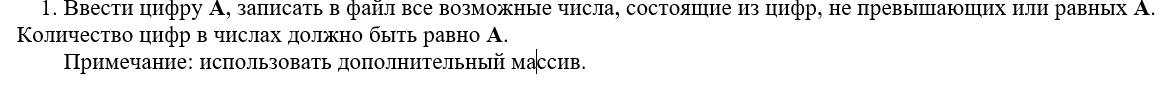
}







1



#include <iostream>

#include <fstream>

using namespace std;

void generateNumbers(int arr[], int n, int A, ofstream& file, int k = 0) // рекурсивная функция для генерации всех возможных чисел

{

if (k == n) // елси достигнута нужная длина числа

{

for (int i = 0; i < n; i++) // выводим число в файл

{

file << arr[i];

}

file << endl;

}

else

{

for (int i = 1; i <= A; i++) // для каждой цифры от 1 до A

{

arr[k] = i; // записываем цифру на текущую позицию

generateNumbers(arr, n, A, file, k + 1); // Рекурсивно генерируем оставшиеся цифры

}

}

}

int main()

{

setlocale(LC\_ALL, "rus");

int A;

cout << "Введите число A: ";

cin >> A;

int \*arr = new int[A]; // Массив для хранения текущего числа

ofstream file("A.txt");

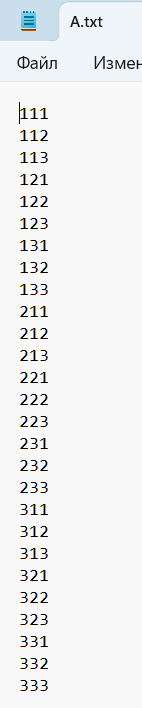
generateNumbers(arr, A, A, file); // Генерируем все возможные числа

file.close(); // Закрываем файл

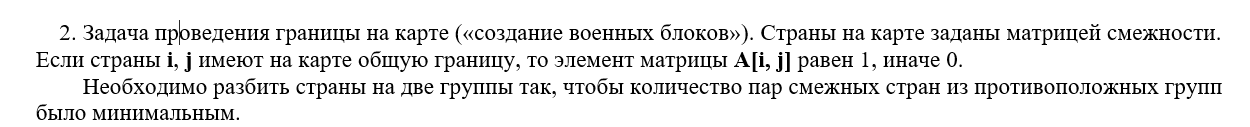
return 0;

}





2



#include <iostream>

const int MAX = 100;

using namespace std;

bool isSafe(int country, int graph[MAX][MAX], int group[], int currentGroup, int N) // проверяем можно ли добавить страну в группу

{

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

if (graph[country][i] && group[i] == currentGroup)

return false;

}

return true;

}

bool partitionUtil(int graph[MAX][MAX], int group[], int country, int N) // Рекурсивная функция для разбиения стран на две группы

{

if (country == N) // Если все страны разделены на группы, возвращаем true

return true;

//пробуем добавить в одну группу

group[country] = 1;

if (isSafe(country, graph, group, 1, N) && partitionUtil(graph, group, country + 1, N))

return true;

// Если не удалось, попробуем добавить во вторую

group[country] = 0;

if (isSafe(country, graph, group, 0, N) && partitionUtil(graph, group, country + 1, N))

return true;

// Если ни в одну группу добавить не удалось, возвращаем false

return false;

}

void partition(int graph[MAX][MAX], int N) // Функция для разбиения стран на две группы

{

int group[MAX] = { 0 };

if (!partitionUtil(graph, group, 0, N))

{

cout << "Невозможно разбить страны на две группы" << endl;

return;

}

cout << "Группа 1: ";

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

if (group[i] == 1)

cout << i << " ";

}

cout << endl;

cout << "Группа 2: ";

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

if (group[i] == 0)

cout << i << " ";

}

cout << endl;

}

int main()

{

setlocale(LC\_ALL, "rus");

int graph[MAX][MAX] = {

{0, 1, 0, 0},

{1, 0, 0, 0},

{0, 0, 0, 1},

{0, 0, 1, 0}

};

int N = 4; // Количество стран

partition(graph, N);

return 0;

}

