Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа 10

По дисциплине «Основы алгоритмизации и программирования»

На тему «Рекурсивные алгоритмы»

Выполнил:

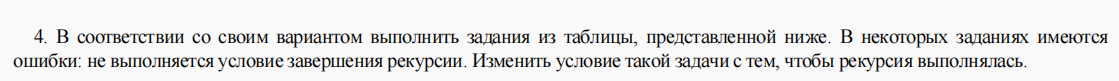
Студент 1 курса 6 группы

Романов Игорь Вячеславович

Преподаватель: асс. Андронова М.В.

2024, Минск

Вариант 12



#include <iostream>

#include <cmath> //Для правильной работы косинуса

using namespace std;

double y(double x, double n) {

if (!n) {

return cos(x); //Если множитель = 0

}

else {

return cos(x \* n) + y(x, n - 1); //Используем рекурсию для нахождения суммы косинусов

}

}

int main() {

setlocale(LC\_ALL, "rus");

double n, x;

cout << "Введите аргумент косинуса: ";

cin >> x;

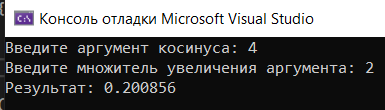
cout << "Введите множитель увеличения аргумента: ";

cin >> n;

cout << "Результат: " << y(x, n) << endl;

return 0;

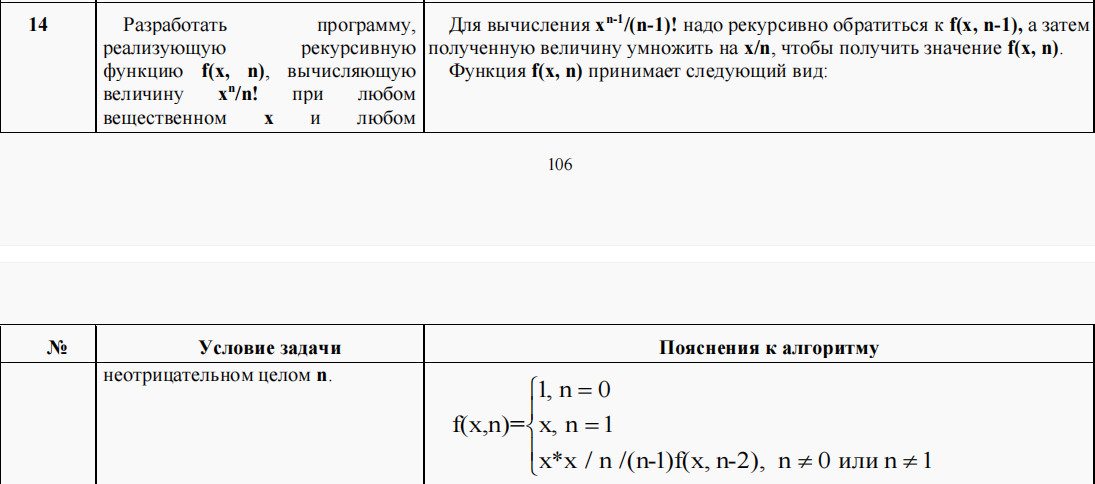
}

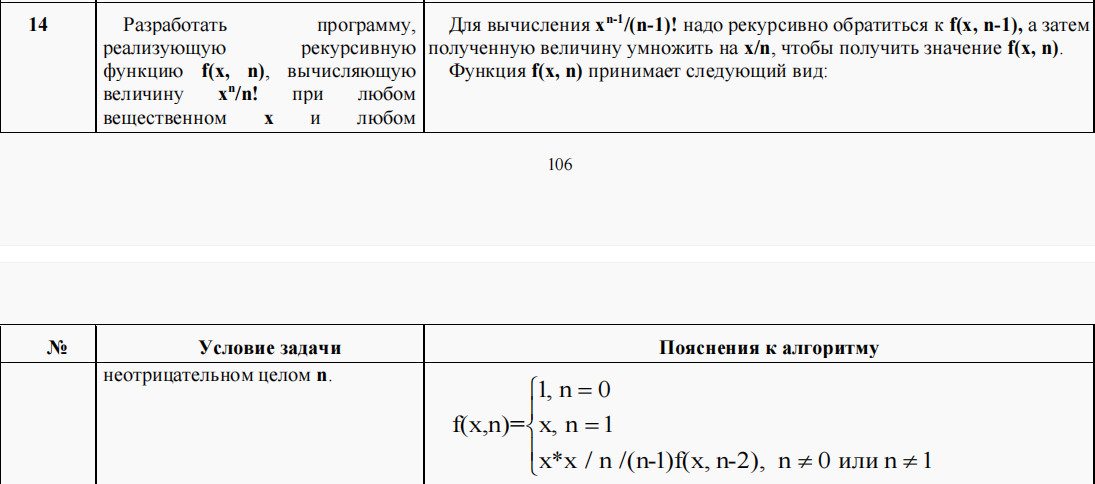






Вариант 14





#include <iostream>

using namespace std;

double factorial(double n) { //Вычисляем факториал значения n

if (!n) {

return 1;

}

else {

return n \* factorial(n - 1);

}

}

double y(double x, double n) {

return pow(x ,n) / factorial(n); //Используем рекурсию для вычисления функции

}

int main() {

setlocale(LC\_ALL, "rus");

double n, x;

cout << "Введите x: ";

cin >> x;

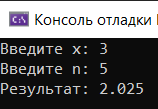
cout << "Введите n: ";

cin >> n;

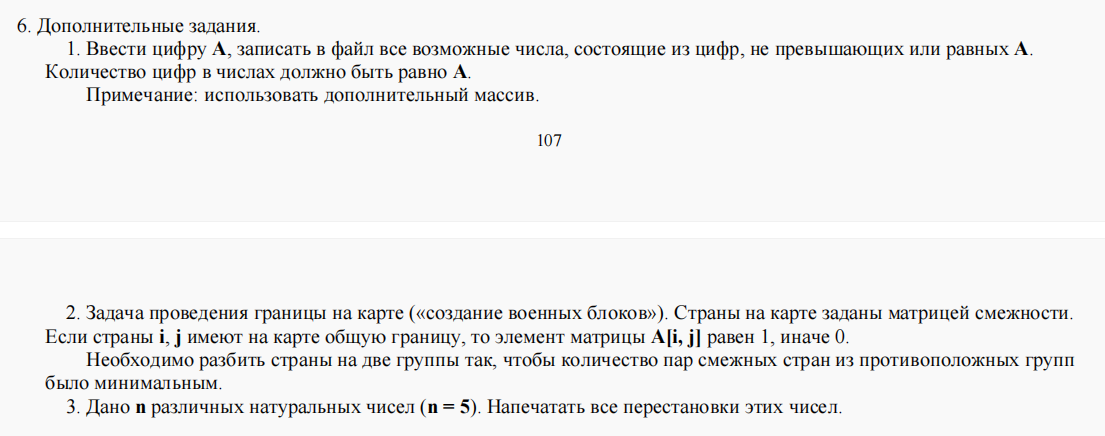
cout << "Результат: " << y(x, n) << endl;

return 0;

}



Доп. задания

6.1

#include <iostream>

#include <fstream>

using namespace std;

void toFile(int, ofstream&, int, int\*);

void toFile(int digit, ofstream& ofile, int current, int\* row) {

if (current == digit) { //Если текущая цифра совпала с изначальной, то закачиваем подбор цифр

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

ofile << row[i]; //Записываем их в файл

}

ofile << endl;

return;

}

for (int i = 0; i <= digit; i++) { //Разряд числа не должен превышать исходную цифру

row[current] = i; //Записываем в каждый разряд новую цифру

toFile(digit, ofile, current + 1, row); //После записи числа увеличиваем левый разряд числа на 1

}

}

int main()

{

setlocale(LC\_ALL, "rus");

int A;

cout << "Введите цифру A: ";

cin >> A;

while (A < 0 || A > 9) {

cout << "Введено не число, попробуйте еще раз: ";

cin >> A;

}

ofstream ofile("row.txt"); //Создаем файл ofstream для записи

if (ofile.fail()) { //В случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

int\* row = new int[A]; //Выделение динамической памяти для массива чисел

toFile(A, ofile, 0, row);

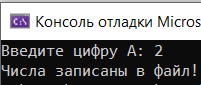
ofile.close();

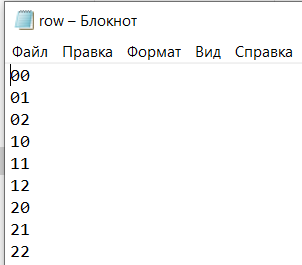
cout << "Числа записаны в файл!";

delete[] row;

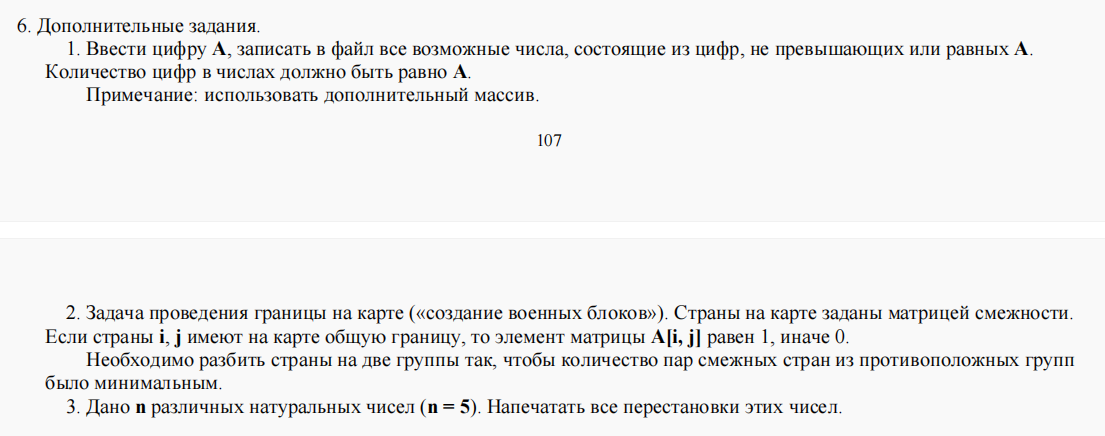
return 0;

}





6.2



#include <iostream>

using namespace std;

const int MAX = 100; // Максимальное количество стран

int matrix[MAX][MAX]; // Матрица смежности

int colors[MAX]; // Цвета стран: 0 (не окрашена), 1 и 2 (две группы)

int n; // Количество стран

bool canColor(int country, int color) {

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

if (matrix[country][i] && colors[i] == color) {

return false; // Если есть смежная страна с тем же цветом, вернуть false

}

}

return true;

}

bool colorCountries(int country) {

if (country == n) { // Если все страны были окрашены

return true;

}

for (int c = 1; c <= 2; ++c) {

if (canColor(country, c)) {

colors[country] = c; // Окрасить страну в цвет c

if (colorCountries(country + 1)) { // Рекурсивно окрасить следующую страну

return true;

}

colors[country] = 0; // Откатить окраску текущей страны

}

}

return false; // Не удалось окрасить текущую страну

}

int main() {

setlocale(LC\_ALL, "rus");

cout << "Введите количество стран: ";

cin >> n;

cout << "Введите матрицу смежности:\n";

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

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

cin >> matrix[i][j];

}

}

if (colorCountries(0)) {

cout << "Страны можно разделить на две группы с минимальным количеством смежных пар.\n";

cout << "Группа 1:\n";

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

if (colors[i] == 1) {

cout << i << " ";

}

}

cout << "\nГруппа 2:\n";

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

if (colors[i] == 2) {

cout << i << " ";

}

}

}

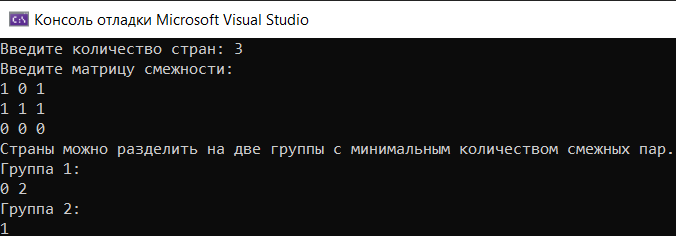
else {

cout << "Невозможно разделить страны на две группы с минимальным количеством смежных пар.\n";

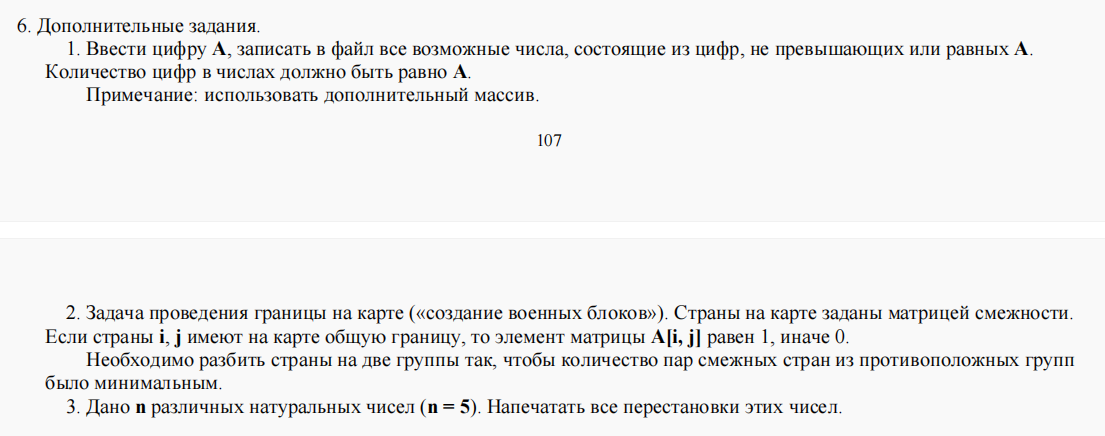
}

return 0;

}



6.3



#include <iostream>

using namespace std;

const int n= 5;

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

void replaceDigits(int [], int, int );

void swapDigits(int&, int&);

//Меняем цифры местами

void swapDigits(int& a, int& b) {

int temp = a;

a = b;

b = temp;

}

//Функция для вывода генерации и вывода перестановок

void replaceDigits(int[], int size, int index) {

if (index == size) { //Если индекс равен размеру массива, значит, мы завершили перестановку

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

cout << numbers[i] << " "; //Выводим получившееся число

}

cout << endl;

}

else { //Иначе продолжаем переставлять числа

for (int i = index; i < size; i++) { //Начинаем пербирать с начального индекс index

swapDigits(numbers[i], numbers[index]); //Меняем текущий элемент с элементом на позиции index

replaceDigits(numbers, size, index + 1); //Вызываем рекурсию для следующего индекса

swapDigits(numbers[i], numbers[index]); // Возвращаем элементы на исходные места для следующей итерации

}

}

}

int main() {

setlocale(LC\_ALL, "rus");

cout << "Перестановки чисел: " << endl;

replaceDigits(numbers, n, 0);

return 0;

}

