ФГБОУ ВО “Чувашский государственный университет им.

И.Н. Ульянова”

Кафедра вычислительной техники

Лабораторная работа №3

**ИСЧЕРПЫВАЮЩИЙ ПОИСК**

Выполнил:

Студент гр. ИВТ-41-22

Иванов В.С.

Проверил:

кандидат технических наук

Павлов Леонид Александрович

Чебоксары 2023

***Цель работы*** – ознакомление с методами решения комбинаторных задач, получение навыков программирования оптимизационных задач.

**Подготовка к работе**

1. Разработать алгоритм и программу решения задачи коммивояжера с использованием общей схемы решения методом ветвей и границ [ ].

определить

// последнее сохраненное решение есть

// решение с наименьшей стоимостью

3. Разработать алгоритм и программу приближенного решения задачи коммивояжера.

}

}

}

}

}

}

Индивидуальные задания

1. Получить все расстановки восьми ладей на шахматной доске, при которых ни одна ладья не угрожает другой.

#include <iostream>

#include <vector>

using namespace std;

int n, c = 0;

vector <int> perm;

vector <bool> used;

void print\_table() {

c++;

cout << c << ")\n";

for (int i = 1; i <= n; i++) {

for (int j = 1; j <= n; j++) {

if (perm[i] == j) {

printf("L");

}

else {

printf("\*");

}

}

puts("");

}

}

void generate(int pos) {

if (pos == n + 1) {

print\_table();

puts("");

}

else {

for (int i = 1; i <= n; i++) {

if (!used[i]) {

used[i] = true;

perm[pos] = i;

generate(pos + 1);

used[i] = false;

}

}

}

}

int main() {

setlocale(LC\_ALL, "Russian");

n = 6;

perm = vector <int>(n + 1);

used = vector <bool>(n + 1);

int fact = 1;

for (int i = 1; i <= n; i++) {

fact \*= i;

used[i] = false;

}

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

generate(i);

}

cout << "Всего расстановок: " << c << endl;

return 0;

}

3. Дано натуральное число *N*. Найти все меньшие *N* числа Мерсена. Простое число называется числом Мерсена, если оно может быть представлено в виде 2*p* – 1, где *p* – тоже простое число

#include <iostream>

#include <vector>

#include <cmath>

using namespace std;

bool isPrime(int n) {

if (n <= 1) {

return false;

}

for (int i = 2; i <= sqrt(n); ++i) {

if (n % i == 0) {

return false;

}

}

return true;

}

int main() {

int N; // Заданный параметр N

cout << "Enter N: " << endl;

cin >> N;

vector<int> mersenneNumbers;

for (int p = 2; p < N; ++p) { // Перебираем все числа до N

if (isPrime(p)) { // Проверяем, является ли p простым числом

int mersenne = pow(2, p) - 1;

if (mersenne < 0) continue;

if (mersenne < N) {

mersenneNumbers.push\_back(mersenne);

}

}

}

cout << "Mersenn's numbers less than " << N << ":\n";

for (int num : mersenneNumbers) {

cout << num << " ";

}

cout << endl;

return 0;

}

3. Имеется *n* человек, которых нужно назначить на *n* работ. Стоимость назначения *i*-го человека на *i*-ю работу равна *Cij*. Найти назначение, при котором каждая работа выполняется некоторым человеком и которое минимизирует общую стоимость назначения.

#include <iostream>

#include <vector>

#include <limits>

const int MAXN = 100;

int cost[MAXN][MAXN]; // Максимальное количество работников и работ

int n; // количество работников

int minCost(int worker, std::vector<bool>& assigned, std::vector<int>& assignment) {

if (worker == n) {

int totalCost = 0;

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

totalCost += cost[i][assignment[i]];

}

return totalCost;

}

int min\_cost = std::numeric\_limits<int>::max();

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

if (!assigned[job]) {

assigned[job] = true;

assignment[worker] = job;

int current\_cost = minCost(worker + 1, assigned, assignment);

if (current\_cost < min\_cost) {

min\_cost = current\_cost;

}

assigned[job] = false;

}

}

return min\_cost;

}

int main() {

// Инициализация стоимости назначения работников на работы

// cost[i][j] - стоимость назначения i-го работника на j-ю работу

cost[0][0] = 3; cost[0][1] = 4; cost[0][2] = 1;

cost[1][0] = 2; cost[1][1] = 5; cost[1][2] = 6;

cost[2][0] = 7; cost[2][1] = 0; cost[2][2] = 2;

n = 3; // количество работников

std::vector<bool> assigned(n, false); // флаги, указывающие, назначен ли работник на работу

std::vector<int> assignment(n); // назначения работников на работы

int totalCost = minCost(0, assigned, assignment);

std::cout << "Assignment: ";

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

std::cout << "(" << i << ", " << assignment[i] << ") ";

}

std::cout << "with total cost " << totalCost << std::endl;

return 0;

}

Выводы: ознакомился с методами решения комбинаторных задач, получил навыки программирования оптимизационных задач.