Министерство науки и образования РФ

Федеральное государственное бюджетное образовательное учреждение   
высшего профессионального образования   
«Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»   
им. В.И. Ульянова (Ленина)»

КАФЕДРА МО ЭВМ

Отчет по лабораторной работе №2

на тему:

“Поиск с возвратом”

по дисциплине “Алгоритмы и Структуры данных ”

Выполнил: студентка Камышова Ю.А.

Факультет КТИ

Группа 1381

Проверил: Казаков Б.Б.

Санкт-Петербург

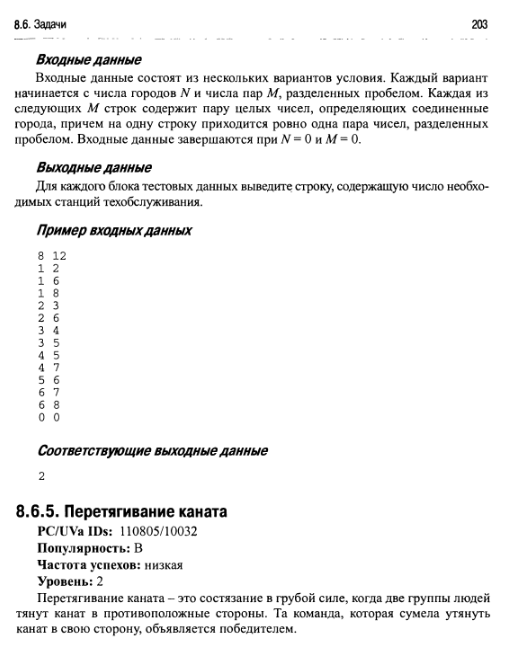
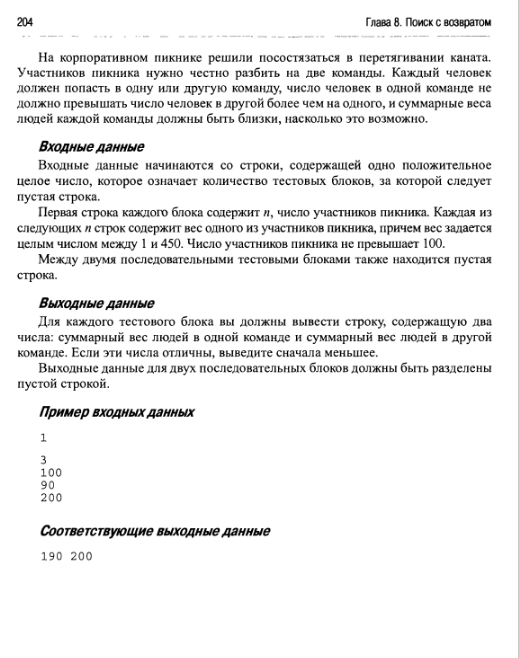
2013

**Цель**

Научиться применять на практике поиск с возвратом, используя поставленную задачу. Написать программу с использованием рекурсивной функции.

**Задание**

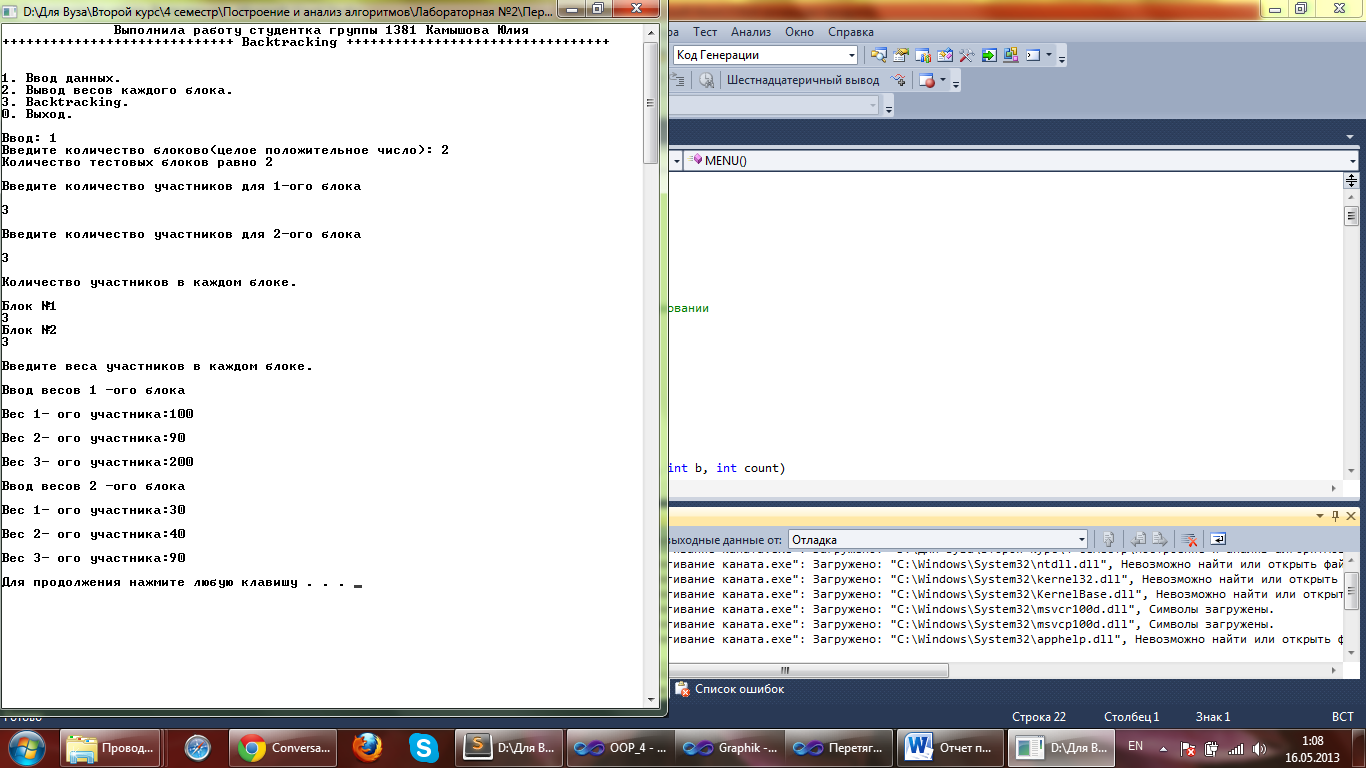
8.6.5 . Перетягивание каната



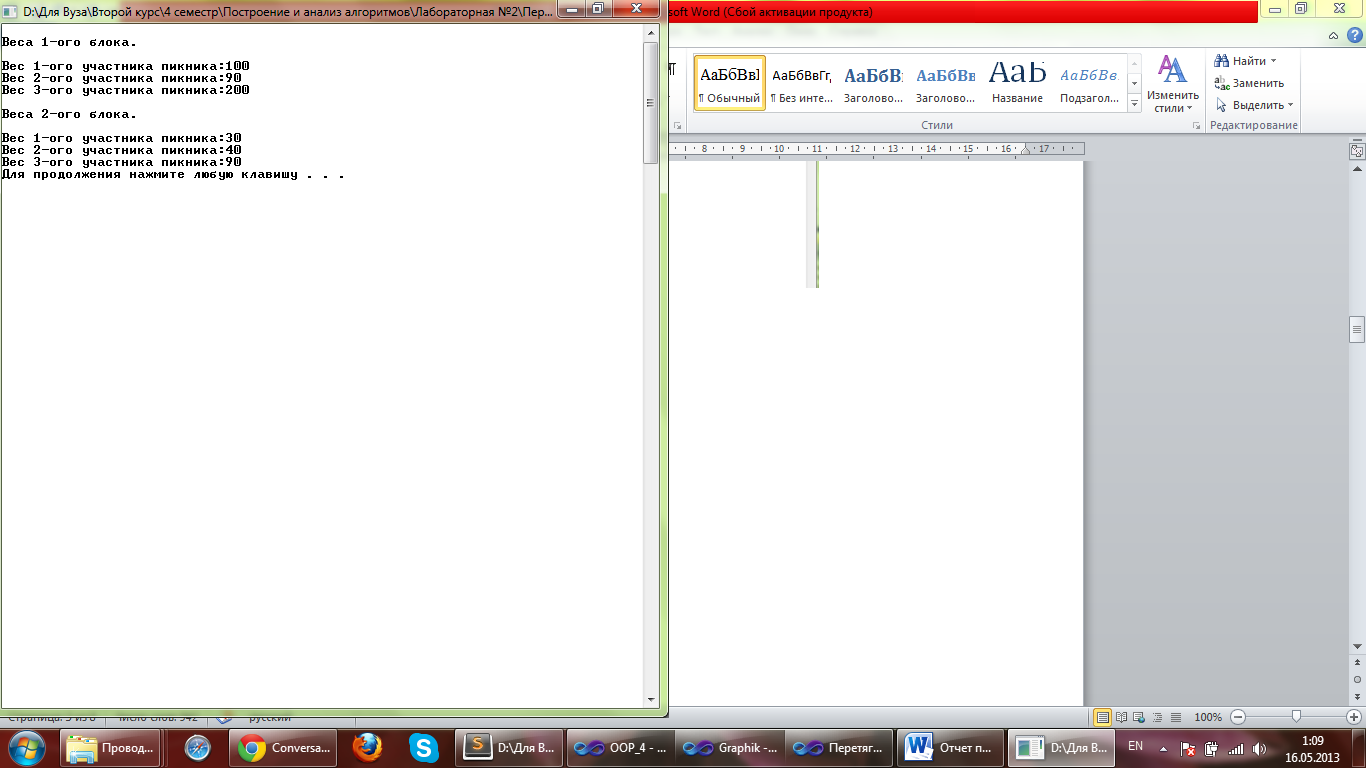
**Описание работы программы**

( принтскрины )

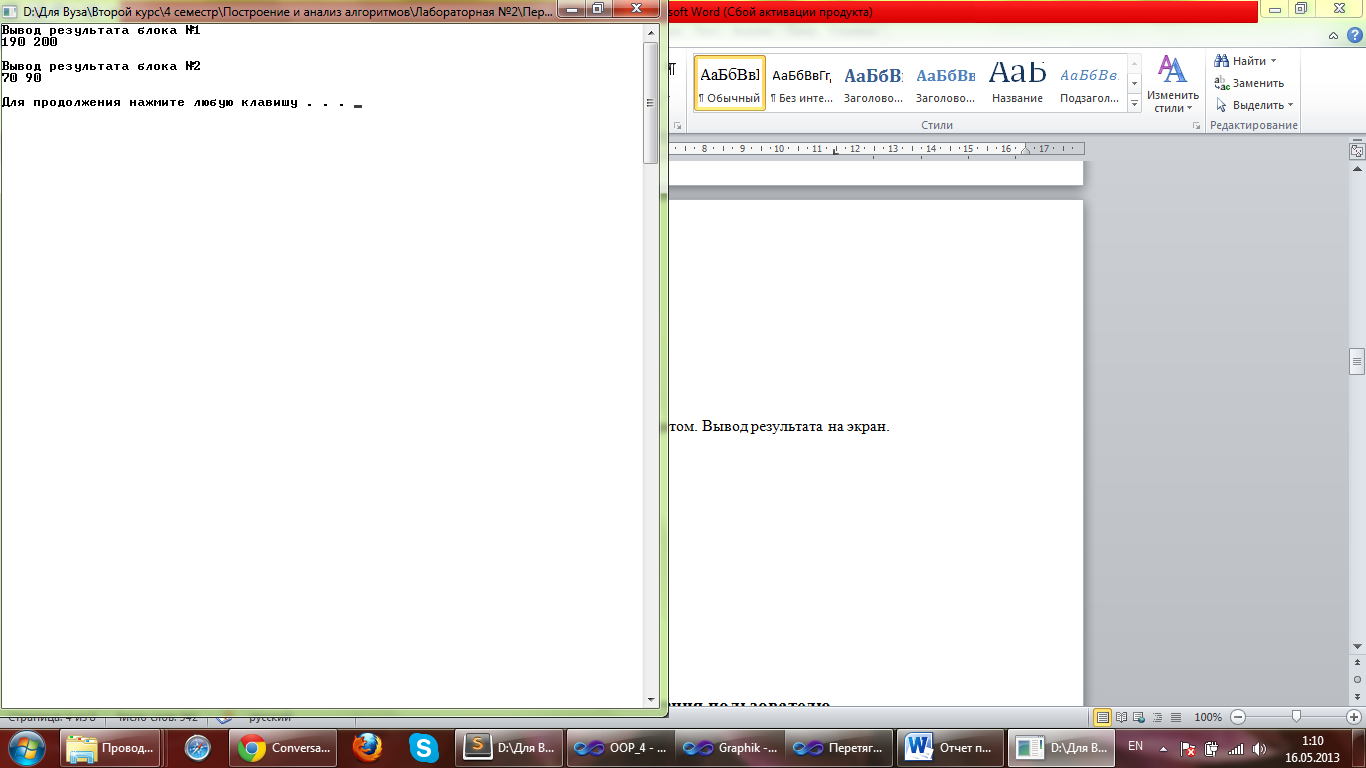
Шаг №1. Запуск программы. Задание количества тестовых блоков, количества участников, задание весов.



Шаг № 2. Вывод данных на экран.



Шаг №3. Применение поиска с возвратом. Вывод результата на экран.



**Инструкция пользователю**

1. Запустите программу Перетягивание каната.exe.

2. Введите такие данные как количество тестовых блоков, количество участников в каждом блоке и веса каждого участника. Есть ограничения на ввод количества участников (от 2 до 100) и на вес (от 1 до 450).

3. Запустите поиск с возвратом предварительно выбрав пункт меню “Backtracking”.

**Текст программы**

#include "stdafx.h"

#include <iostream>

#include "stdlib.h"

#include <fstream>

using namespace std;

// Поиск с возвратом

bool see\_all(bool see[], int n) //Функция проверяющая все ли люди использовались в соревновании

{ //Если использовались все участникивернуть true, иначе false

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

{

if (see[i] != true)

return false;

}

return true;

}

//backtrack

int backtrack(int &comA, int &comB, int chislo, int massiv[], bool k, bool see[], int a, int b, int count)

{

int p = 0; //параметр определяющий какой человек используеться на данном шаге перебора

for (int i = 0; i < chislo; i++) //Пока не найдем неиспользованного участника

{

if (see[i] != true) //Если участник не использован, то

{ //запишем его к одной из команд(определяеться параметром k)

if (k == true)

{

count++;

a += massiv[i];

}

else

{

count--;

b += massiv[i];

}

see[i] = true; //отметим то, что i участник использовался

p = i;

break;

}

}

if (see\_all(see,chislo)) //Если использованы все участники

{

if ((abs(a-b) < abs(comA - comB)) && ((count >=-1) && (count <=1)))

{ //Проверяем на то, что разность весов в командах меньше существующей и то, что разница людей в команде не больше одного

comA = a; //Если да то фиксируем решение

comB = b;

if (comA - comB == 0) //Если разница весов 0, то решение оптимально и выходим

return 1;

}

}

else //Если не использованы все участники, то запускаем перебор заново

{

int ret = backtrack(comA, comB, chislo, massiv, true, see, a, b, count); //запускаем перебор рекурсивно (добавляем в первую команду)

see[++p] = false; //участник использовавшийся в последнем переборе используеться заново, а его вес не учитываться

if (ret == 1) //Если разница между весами команд 0, то выход

return 1;

ret = backtrack(comA, comB, chislo, massiv, false, see, a, b, count); //запускаем перебор рекурсивно (добавляем во вторую команду)

see[p] = false; //участник использовавшийся в последнем переборе используеться заново, а его вес не учитываеться

if (ret == 1) //Если разница между весами команд 0, то выход

return 1;

}

return 0;

}

// Построение блока

void postroenie\_bloka(int \*\*massiv,int blok,int \*A)

{

int i,j;

cout<<"Введите веса участников в каждом блоке.\n"<<endl;

for(i = 0; i < blok; i++)

{

cout<<"Ввод весов "<<i+1<<" -ого блока\n"<<endl;

for(j = 0; j < A[i]; j ++)

{

cout<<"Вес "<<j+1<<"- ого участника:";

cin>>massiv[i][j];

do

{

if(massiv[i][j] > 450 || massiv[i][j] <= 0)

{

cout<<"Введите вес участника от 1 до 450!"<<endl;

cout<<"Ввод:";

cin>>massiv[i][j];

}

}while(massiv[i][j] > 450 || massiv[i][j] <= 0);

cout<<"\n";

}

}

}

// Вывод на экран кол-ва введенных блоков и весов участников

void bloks\_and\_weight(int \*\*massiv,int blok,int \*A)

{

int i,j;

for(i = 0; i < blok; i++)

{

cout<<"\nВеса "<<i+1<<"-ого блока.\n"<<endl;

for(j = 0; j < A[i]; j ++)

{

cout<<"Вес "<<j+1<<"-ого участника пикника:";

cout<<massiv[i][j];

cout<<"\n";

}

}

}

void MENU()

{

cout<<" Выполнила работу студентка группы 1381 Камышова Юлия\n"

"+++++++++++++++++++++++++++++ Backtracking +++++++++++++++++++++++++++++++++\n\n\n"

"1. Ввод данных.\n"

"2. Вывод весов каждого блока.\n"

"3. Backtracking.\n"

"0. Выход.\n"<<endl;

}

int main()

{

setlocale(0,"Russian");

char str[256];

int blok = 0,i;

int \*\*massiv,\*A=NULL;

int menu = 0;

int \*comA = NULL;

int \*comB = NULL;

do

{

system("cls");

MENU();

cout<<"Ввод: ";

cin >> menu;

switch(menu)

{

case 1:

cout<<"Введите количество блоково(целое положительное число): ";

cin >> str;

blok = atoi(str);

do

{

if(blok < 0)

{

cout<<"Введите корректное число блоков!\n"<<endl;

cout<<"Введите количество тестовых блоково(целое положительное число): ";

cin >> str;

blok = atoi(str);

}

}while(blok < 0);

cout<<"Количество тестовых блоков равно "<<blok<<endl;

A = new int[blok];

massiv = new int\*[blok];

for (i = 0; i < blok; i++)

{

cout<<"\nВведите количество участников для "<<i+1<<"-ого блока\n"<<endl;

cin>>A[i];

massiv[i] = new int[A[i]];

do

{

if(A[i] > 100 || A[i] <= 0 || A[i] == 1)

{

cout<<"\nВведите количество от 2 до 100 участников для "<<i+1<<"-ого блока!"<<endl;

cout<<"Ввод: ";

cin>>A[i];

}

}while(A[i] > 100 || A[i] <= 0 || A[i]== 1);

}

cout<<"\n";

cout<<"Количество участников в каждом блоке.\n"<<endl;

for(i = 0; i < blok; i++)

{

cout<<"Блок №"<<i+1<<endl;

cout<<A[i]<<endl;

}

cout<<"\n";

postroenie\_bloka(massiv,blok,A);

system("pause");

break;

case 2:

system("cls");

bloks\_and\_weight(massiv,blok,A);

system("pause");

break;

case 3:

system("cls");

//Решение задачи

comA = new int[blok];

comB = new int[blok];

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

{

comA[i] = INT\_MAX; //Сделаем так что бы разность между командами

comB[i] = 0; //в начале была максимальной

int count = 0;

bool \*see = new bool[A[i]];

for (int k = 0; k < A[i]; k++)

{

see[k] = false;

}

backtrack(comA[i], comB[i], A[i], massiv[i], true, see, 0, 0, 0);

cout<<"Вывод результата блока №"<<i+1<<endl;

cout<<comA[i]<<" "<<comB[i]<<endl;

cout<<"\n";

}

system("pause");

break;

case 0:

break;

default:

cout<<"Введите корренктный пункт меню!\n"<<endl;

system("pause");

};

}while(menu != 0);

system("pause");

return 0;

}

**Вывод**

В ходе проделанной работы был написан и отработан алгоритм поиска с возвратом. Получены практические навыки работы с ним.