МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №1
по дисциплине «Алгоритмы и структуры данных»
Тема: Рекурсивный функции

Студент(ка) гр. 9382 _____ Голубева В.П.

Преподаватель _____ Фирсов М.А.

Санкт-Петербург
2020

Цель работы.

Научиться писать рекурсивные алгоритмы.

Задание.

4. Напечатать все перестановки заданных n различных натуральных чисел (или символов).

Основные теоретические положения.

Рукурсивная функция — функция, которая вызывает себя из себя же самой.

Реализованные функции.

void change_elements(int& a, int& b) — функция принимает на вход две ссылки типа на целые числа int&.

bool my_isdigit(string s) — принимает строку s типа string.

void my_permutation(string s, int* array, int numer, int count, ofstream & res) — строку для перестановки string s, указатель на массив int* array, количество элементов в массиве int count, счётчик int numer и ссылку на поток для записи результатов в файл ofstream & res.

Алгоритмы и структуры данных.

void change_elements(int& a, int& b) — меняет местами с помощью локальной переменной int с две целочисленные переменные.

bool my_isdigit(string s) — функция с помощью цикла for проходит по символам строки, проверяет, состоит ли она из чисел. Если хоть один символ из строки не цифра — функция возвращает false, иначе цикл не прерывается и возвращается true.

void my_permutation(string s, int* array, int numer, int count, ofstream & res) — рукурсивная функция принимает искомую строку для перестановки

string s, указатель на массив с номерами элементов в перестановке int* array, количество элементов в массиве int numer и ссылку на поток для записи результатов в файл ofstream & res. Функция рекурсивно переставляет номера элементов для строки, а затем выводит перестановку в соответствии с номерами этих элементов.

В главной функции проверяется корректность введённого значения n. Формируются переменные ifstream temp - для чтения файла с входными данными и ofstream res — для записи в выхоной файл. С помощью цикла for по заданному числу n программа проходит по строчкам файла и вызывает функцию my_permutation. Затем происходит закрытие файлов ввода-вывода.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	qwerty	Incorrect value of n. Try again	Некорректное значение для n(n — количество, должно быть целым неотрицательным числом)
2.	-123	Incorrect value of n. Try again	Некорректное значение для n(n — количество, должно быть целым неотрицательным числом)

Пример работы программы:

n=9

Current sequence is: 1357

Numer of digits: 0 1 2 3 Permutation: 1357

Numer of digits: 0 1 3 2 Permutation: 1375

Numer of digits: 0 2 1 3 Permutation: 1537

Numer of digits: 0 2 3 1 Permutation: 1573

Numer of digits: 0 3 2 1 Permutation: 1753

Numer of digits: 0 3 1 2 Permutation: 1735

Numer of digits: 1 0 2 3 Permutation: 3157

Numer of digits: 1 0 3 2 Permutation: 3175

Numer of digits: 1 2 0 3 Permutation: 3517

Numer of digits: 1 2 3 0 Permutation: 3571

Numer of digits: 1 3 2 0 Permutation: 3751

Numer of digits: 1 3 0 2 Permutation: 3715

Numer of digits: 2 1 0 3 Permutation: 5317

Numer of digits: 2 1 3 0 Permutation: 5371

Numer of digits: 2 0 1 3 Permutation: 5137

Numer of digits: 2 0 3 1 Permutation: 5173

Numer of digits: 2 3 0 1 Permutation: 5713

Numer of digits: 2 3 1 0 Permutation: 5731

Numer of digits: 3 1 2 0 Permutation: 7351

Numer of digits: 3 1 0 2 Permutation: 7315

Numer of digits: 3 2 1 0 Permutation: 7531

Numer of digits: 3 2 0 1 Permutation: 7513

Numer of digits: 3 0 2 1 Permutation: 7153

Numer of digits: 3 0 1 2 Permutation: 7135

Current sequence is: ygj

Numer of digits: 0 1 2 Permutation: ygj

Numer of digits: 0 2 1 Permutation: yjg

Numer of digits: 1 0 2 Permutation: gyj

Numer of digits: 1 2 0 Permutation: gjy

Numer of digits: 2 1 0 Permutation: jgy

Numer of digits: 2 0 1 Permutation: jyg

Current sequence is: 9860

Numer of digits: 0 1 2 3 Permutation: 9860

Numer of digits: 0 1 3 2 Permutation: 9806

Numer of digits: 0 2 1 3 Permutation: 9680

Numer of digits: 0 2 3 1 Permutation: 9608

Numer of digits: 0 3 2 1 Permutation: 9068

Numer of digits: 0 3 1 2 Permutation: 9086

Numer of digits: 1 0 2 3 Permutation: 8960

Numer of digits: 1 0 3 2 Permutation: 8906

Numer of digits: 1 2 0 3 Permutation: 8690

Numer of digits: 1 2 3 0 Permutation: 8609

Numer of digits: 1 3 2 0 Permutation: 8069

Numer of digits: 1 3 0 2 Permutation: 8096

Numer of digits: 2 1 0 3 Permutation: 6890

Numer of digits: 2 1 3 0 Permutation: 6809

Numer of digits: 2 0 1 3 Permutation: 6980

Numer of digits: 2 0 3 1 Permutation: 6908

Numer of digits: 2 3 0 1 Permutation: 6098

Numer of digits: 2 3 1 0 Permutation: 6089

Numer of digits: 3 1 2 0 Permutation: 0869

Numer of digits: 3 1 0 2 Permutation: 0896

Numer of digits: 3 2 1 0 Permutation: 0689

Numer of digits: 3 2 0 1 Permutation: 0698

Numer of digits: 3 0 2 1 Permutation: 0968

Numer of digits: 3 0 1 2 Permutation: 0986

Current sequence is: 4

Numer of digits: 0 Permutation: 4

Current sequence is: 3768

Numer of digits: 0 1 2 3 Permutation: 3768

Numer of digits: 0 1 3 2 Permutation: 3786

Numer of digits: 0 2 1 3 Permutation: 3678

Numer of digits: 0 2 3 1 Permutation: 3687

Numer of digits: 0 3 2 1 Permutation: 3867

Numer of digits: 0 3 1 2 Permutation: 3876

Numer of digits: 1 0 2 3 Permutation: 7368

Numer of digits: 1 0 3 2 Permutation: 7386

Numer of digits: 1 2 0 3 Permutation: 7638

Numer of digits: 1 2 3 0 Permutation: 7683

Numer of digits: 1 3 2 0 Permutation: 7863

Numer of digits: 1 3 0 2 Permutation: 7836

Numer of digits: 2 1 0 3 Permutation: 6738

Numer of digits: 2 1 3 0 Permutation: 6783

Numer of digits: 2 0 1 3 Permutation: 6378

Numer of digits: 2 0 3 1 Permutation: 6387

Numer of digits: 2 3 0 1 Permutation: 6837

Numer of digits: 2 3 1 0 Permutation: 6873

Numer of digits: 3 1 2 0 Permutation: 8763

Numer of digits: 3 1 0 2 Permutation: 8736

Numer of digits: 3 2 1 0 Permutation: 8673

Numer of digits: 3 2 0 1 Permutation: 8637

Numer of digits: 3 0 2 1 Permutation: 8367

Numer of digits: 3 0 1 2 Permutation: 8376

Current sequence is: jy

Numer of digits: 0 1 Permutation: jy

Numer of digits: 1 0 Permutation: yj

Current sequence is: 2318

Numer of digits: 0 1 2 3 Permutation: 2318

Numer of digits: 0 1 3 2 Permutation: 2381

Numer of digits: 0 2 1 3 Permutation: 2138

Numer of digits: 0 2 3 1 Permutation: 2183

Numer of digits: 0 3 2 1 Permutation: 2813

Numer of digits: 0 3 1 2 Permutation: 2831

Numer of digits: 1 0 2 3 Permutation: 3218

Numer of digits: 1 0 3 2 Permutation: 3281

Numer of digits: 1 2 0 3 Permutation: 3128

Numer of digits: 1 2 3 0 Permutation: 3182

Numer of digits: 1 3 2 0 Permutation: 3812

Numer of digits: 1 3 0 2 Permutation: 3821

Numer of digits: 2 1 0 3 Permutation: 1328

Numer of digits: 2 1 3 0 Permutation: 1382

Numer of digits: 2 0 1 3 Permutation: 1238

Numer of digits: 2 0 3 1 Permutation: 1283

Numer of digits: 2 3 0 1 Permutation: 1823

Numer of digits: 2 3 1 0 Permutation: 1832

Numer of digits: 3 1 2 0 Permutation: 8312

Numer of digits: 3 1 0 2 Permutation: 8321

Numer of digits: 3 2 1 0 Permutation: 8132

Numer of digits: 3 2 0 1 Permutation: 8123

Numer of digits: 3 0 2 1 Permutation: 8213

Numer of digits: 3 0 1 2 Permutation: 8231

Current sequence is: a

Numer of digits: 0 Permutation: a

Current sequence is: 8273

Numer of digits: 0 1 2 3 Permutation: 8273

Numer of digits: 0 1 3 2 Permutation: 8237

Numer of digits: 0 2 1 3 Permutation: 8723

Numer of digits: 0 2 3 1 Permutation: 8732

Numer of digits: 0 3 2 1 Permutation: 8372

Numer of digits: 0 3 1 2 Permutation: 8327

Numer of digits: 1 0 2 3 Permutation: 2873

Numer of digits: 1 0 3 2 Permutation: 2837

Numer of digits: 1 2 0 3 Permutation: 2783

Numer of digits: 1 2 3 0 Permutation: 2738

Numer of digits: 1 3 2 0 Permutation: 2378

Numer of digits: 1 3 0 2 Permutation: 2387

Numer of digits: 2 1 0 3 Permutation: 7283

Numer of digits: 2 1 3 0 Permutation: 7238

Numer of digits: 2 0 1 3 Permutation: 7823

Numer of digits: 2 0 3 1 Permutation: 7832

Numer of digits: 2 3 0 1 Permutation: 7382

Numer of digits: 2 3 1 0 Permutation: 7328

Numer of digits: 3 1 2 0 Permutation: 3278

Numer of digits: 3 1 0 2 Permutation: 3287

Numer of digits: 3 2 1 0 Permutation: 3728

Numer of digits: 3 2 0 1 Permutation: 3782

Numer of digits: 3 0 2 1 Permutation: 3872

Numer of digits: 3 0 1 2 Permutation: 3827

Выводы.

Были изучены алгоритмы рекурсии. Была разработана рекрсивная программа, выводящая все перестановки в числе(или последовательности символов).

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: Makefile

```
Permutation: 1alg.o
	g++ -o Permutation 1alg.o
1alg.o: 1alg.cpp name.hpp
	g++ -c 1alg.cpp -o 1alg.o
clean:
	rm -rf *.o Permutation
```

Название файла: 1alg.cpp

```
#include "name.hpp"

void change_elements(int& a, int& b){//меняем местами элементы
	int c;
	c=a;
	a=b;
	b=c;
}

bool my_isdigit(string s){//проверка строки, состоит ли она из цифр
	for (int i=0;i<s.length();i++){
		if (!isdigit(s[i]))
		return false;
	}
	return true;
}

	void my_permutation(string s, int* array, int numer, int count, ofstream
& res){//функция генерации номеров перестановок и вывод по значению
	for (int i=0;i<numer;i++){
			cout<<' ';
			res<<' ';

		}

	if (numer == count){

		cout<<" Numer of digits: ";
		res<<" Numer of digits: ";

		for (int i=0;i<count;i++){//
			cout<<*(array+i)<<' ';
			res<<*(array+i)<<' ';
		}
		cout<<"Permutation: ";
		res<<"Permutation: ";
		for (int i=0;i<count;i++){
			cout<<s[*(array+i)];
			res<<s[*(array+i)];
		}
		cout<<'\n';
		res<<'\n';
```

```
            }
            else{
                  for (int j=numer; j<count; j++){
                        change_elements(*(array+numer), *(array+j));
                        my_permutation(s, array, numer+1, count, res);//вызов
рекурсивной функции
                        change_elements(*(array+numer), *(array+j));
                  }
            }
      }

      int main(){

            string n;
            cin>>n;
            cout<<'\n';

            while(!my_isdigit(n)){
                  cout<<"Incorrect value of n. Try again\n";

                  cin>>n;
                  cout<<'\n';


                  }

            int p=stoi(n);
            ifstream temp("test.txt");
            ofstream res("res.txt", ios::app);
            string s;
            int j=0;
            if (temp.is_open()&&res.is_open()){
                  while(temp>>s && j<p){//извлечение данных из файла
                        cout<<" Current sequence is: "<<s<<'\n';
                        res<<" Current sequence is: "<<s<<'\n';;
                        int count=s.length();
                        int *array= new int(count);

                        for (int i=0; i<count;i++)
                              array[i]=i;
                        my_permutation(s, array, 0, count, res);//генерируем
перестановки для искомой строки из файла
                        j+=1;
                        delete[] array;
                        cout<<'\n';
                        res<<'\n';
                  }
            }
            else{
                  cout<<"Problem with opening test.txt or res.txt";
            }
            res<<"\n----------------------------------------------------------\
n";
            temp.close();
            res.close();
            return 0;
      }
```

Название файла: name.hpp

```
#pragma once
```

```
#include <iostream>
#include <cstring>
#include <fstream>
#include <limits>
#include <cctype>

using namespace std;

void change_elements(int& , int& );
bool my_isdigit(string );
void my_permutation(string , int* , int , int , ofstream & );
```