**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: Рекурсия**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 6383 |  | Лавренкова Е.Л. |
| Преподаватель |  | Шолохова О.М. |

Санкт-Петербург

2017

**Цель работы.**

Ознакомиться с основными понятиями и приёмами рекурсивного программирования, получить навыки программирования рекурсивных процедур и функций на языке программирования С++.

**Задание (вариант №4).**

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

**Основные теоретические положения.**

Функция факториал натурального аргумента n обозначается как n! и определяется соотношением

n!=1·2·3...(n – 1)·n . (2.1)

Удобно доопределить 0!=1 и считать, что n – целое неотрицательное число.

Некоторым недостатком определения (2.1) является наличие в нём многоточия «...», передающего речевой оборот «и так далее» и имеющего интуитивно понятный читателю смысл. Можно дать точное, так называемое рекурсивное определение функции n!, лишенное этого недостатка, т. е. не апеллирующее к нашей интуиции. Определим:

а) 0! = 1, (2.2)

б) n! = (n – 1)!·n при n > 0.

Соотношения (2.2) можно рассматривать как свойства ранее определенной функции, а можно (как в данном случае) использовать их для определения этой функции.

Далее для функции n! Используется «функциональное» (префиксное) обозначение fact(n), указывая имя функции и за ним в скобках – аргумент. Тогда (2.2) можно записать в виде

⎧ 1, если *n* = 0;

fact (*n*) = ⎨ (2.3)

⎩ fact (*n* − 1) ⋅ *n*, если *n* > 0;

или в другой форме записи

fact (*n*) ≡ **if** *n* = 0 **then** 1 **else** fact (*n* − 1) ⋅ *n*, (2.4)

где использовано условное выражение **if** *b* **then** *e*1**else** *e*2, означающее, что в том месте, где оно записано, следует читать e1, если выполняется условие b, и следует читать e2, если условие b не выполняется.

Функция, определяемая таким образом, единственна. Действительно, пусть есть две функции, например: fact1 (n) и fact2 (n), удовлетворяющие со-отношениям (2.2) или их эквивалентам (2.3), (2.4). Рассмотрим разность dfact (n) = fact1 (n) − fact2 (n). Очевидно, что, во-первых, в силу соотношения «а» из (2.2) имеем dfact (0) = 0, а, во-вторых, для функции dfact (n) также справедливо соотношение «б». Действительно,

dfact (n) = fact1 (n) − fact2 (n) = fact1 (n − 1) · n − fact2 (n − 1) · n =

= (fact1 (n − 1) − fact2 (n − 1)) · n = dfact (n − 1) · n.

По индукции легко доказывается, что из соотношений dfact (0) = 0 и dfact (n) = dfact (n − 1) · n следует, что dfact (n) = 0 для любого n > 0.

**Спецификация программы.**

*Назначение программы*.

Программа предназначена для нахождения всевозможных перестановок для n символов.

*Описание программы*.

Программа написана на языке С++. Входными данными для программы являются символы и их количество, вводимые пользователем с клавиатуры или из файла. Проверка значений при некорректном вводе количества символов предусмотрена только для целых чисел. Выходными данными являются всевозможные перестановки введённых пользователем символов.

*Пример диалога с пользователем*.

Если вы хотите ввести символы из консоли - нажмите 1.

Если вы хотите вывести символы из файла - нажмите 2.

Если вы хотите выйти - нажмите 3.

Введите номер операции:1

Введите количество вводимых символов: 2

Введите символы: 2

Неверное количество символов. Попробуйте ещё раз:

Введите символы: 28

28

82

Если вы хотите ввести символы из консоли - нажмите 1.

Если вы хотите вывести символы из файла - нажмите 2.

Если вы хотите выйти - нажмите 3.

Введите номер операции:3

(Выход из программы)

*Реализация*.

Описание алгоритма:

Сначала перебираем 2 элемента, пока не закончится количество перестановок. Далее увеличиваем количество перебираемых элементов на единицу и производим перестановки уже с новыми данными. Продолжаем данный алгоритм пока не закончатся все элементы строки.

Рекурсия ()

Если текущий\_элемент равен количество\_символов

Вывести строку

Пока элемент\_цикла меньше количество\_символов

Поменять местами (символ\_от\_текущего\_элемента, символ\_от\_элемента\_цикла)

Рекурсия()

Поменять местами (символ\_от\_текущего\_элемента, символ\_от\_элемента\_цикла)

Описание функций:

|  |  |  |  |
| --- | --- | --- | --- |
| *Имя функции* | *Тип функции* | *Входные данные* | *Комментарий* |
| main | int | - | Выбор пользователя способы ввода данных |
| swap | void | int &x, int &y | Меняет местами значения x и y |
| next | int k, char \*A, const int n | Сравнивает введенные символы между собой |

**Тестирование.**

|  |  |  |
| --- | --- | --- |
| **Количество символов** | **Введённые символы** | **Результат** |
| -7 | ― | Количество символов должно быть положительным. Попробуйте ещё раз. |
| 3 | пл | Неверное количество символов. Попробуйте ещё раз: |
| 4 | ыq3& | ыq3& ыq&3 ы3q& ы3&q ы&3q ы&q3 qы3& qы&3 q3ы& q3&ы q&3ы q&ы3 3qы& 3q&ы 3ыq& 3ы&q 3&ыq 3&qы &q3ы &qы3 &3qы &3ыq &ы3q &ыq3 |

**Выводы.**

Была разработана программа, которая находит все перестановки введенных натуральных чисел (или символов). В результате выполнения работы были получены знания о языке программирования С++, а также навыки программирования рекурсивных функций.

**Приложение А. Исходный код.**

#include <iostream>

#include <fstream>

#include <Windows.h>

using namespace std;

// фукнция меняет x и y местами

void swap(int &x, int &y) {

int temp = x;

x = y;

y = temp;

}

// рекурсивная функция переборов всех возможный перестановок строки

void next(int k, char \*A, const int n, bool toFile) {

if (k == n)

{

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

cout << A[i];

if (!toFile)

cout << endl;

else

cout << endl;

return;

}

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

{

swap(A[k], A[i]);

next(k + 1, A, n, toFile);

swap(A[k], A[i]);

}

}

// функция меню

void Menu() {

cout << "Если вы хотите ввести символы из консоли - нажмите 1." << endl;

cout << "Если вы хотите вывести символы из файла - нажмите 2." << endl;

cout << "Если вы хотите выйти - нажмите 3." << endl;

}

int main() {

setlocale(0, "");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int m;

Menu();

cout << "\nВведите номер операции: ";

cin >> m;

while (true) {

switch (m) {

case 1: {

system("cls");

char A[1000] = {};

int n;

cout << "Введите количество вводимых символов: ";

cin >> n;

while (n <= 0) {

cout << "Количество символов должно быть положительным. Попробуйте ещё раз." << endl;

cin >> n;

}

cout << "\nВведите символы: ";

cin >> A;

while (n != strlen(A)) {

cout << "Неверное количество символов. Попробуйте ещё раз: " << endl;

cout << "\nВведите символы: ";

cin >> A;

}

next(0, A, n, false);

Menu();

cout << "\nВведите номер операции: ";

cin >> m;

break;

}

case 2: {

system("cls");

ifstream in;

in.open("in.txt");

if (!in) {

cout << "Файла с данными не существует. Введите символы с консоли." << endl << "\n";

cout << "\nНажмите 1: ";

cin >> m;

break;

}

char A[1000];

int n;

in >> n;

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

in >> A[i];

next(0, A, n, true);

Menu();

cout << "\nВведите номер операции: ";

cin >> m;

break;

}

case 3: {

exit(0);

break;

}

default:

cout << "\nНеверный ввод. Попробуйте ещё раз." << endl;

Menu();

cout << "\nВведите номер операции: ";

cin >> m;

break;

}

}

}