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

ОТЧЕТ

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

Студент гр. 8381 _____ Панченко М. С.

Преподаватель _____ Жангиров Т. Р.

Санкт-Петербург
2019

Цель работы.

Ознакомиться с основными методами и приемами рекурсивного программирования в C++ и написать программу с использованием рекурсии.

Задание.

Вариант 18.

Пусть определена функция Ф преобразования целочисленного век-тора a:

$$\Phi(\alpha) = \begin{cases} \alpha, \text{ если } \|\alpha\| = 1, \\ ab, \text{ если } \|\alpha\| = 2, \alpha = ab \text{ и } a \le b, \\ ba, \text{ если } \|\alpha\| = 2, \alpha = ab \text{ и } b < a, \\ \Phi(\beta)\Phi(\gamma) \quad \text{, если } \|\alpha\| > 2, \ \alpha = \beta\gamma, \text{ где } \|\beta\| = \|\gamma\| \text{ или } \|\beta\| = \|\gamma\| + 1. \end{cases}$$

Например: Ф(1,2,3,4,5) = 1,2,3,4,5; Ф(4,3,2,1) = 3,4,1,2; Ф(4,3,2) = 3,4,2. Отметим, что функция Ф преобразует вектор, не меняя его длину. Реализовать функцию Ф рекурсивно.

Основные теоретические положения.

Рекурсия — определение, описание, изображение какого-либо объекта или процесса внутри самого этого объекта или процесса, то есть ситуация, когда объект является частью самого себя.

Ход работы:

Написание работы производилось на базе операционной системы Windows 10 в среде разработки QtCreator с использованием фреймворка Qt. Сборка, отладка и тестирование также производились в QtCreator.
Для реализации программы был разработан графический интерфейс с помощью встроенного в QtCreator UI-редактора.

В программе были реализованы специальные слоты для обработки сигналов clicked() для кнопки, считывающей, введенную информацию пользователем, с консоли, либо же считывающей из определенно выбранного файла. Вся обработка этих сигналов расположена в файле с названием mainwindow.cpp.

Программа посимвольно считывает входные данные, а именно содержимое исходного вектора. Далее этот вектор попадает в тело рекурсии. Вектор обрабатывается преобразовываясь рекурсией снова и снова, пока его длина на определенной глубине не станет меньше 3. При этом в ходе выполнения программы на консоль отображается глубина рекурсии.

По завершении обработки вектора рекурсией на экран выводится измененный вектор.

Функция: *void rec(int indent, int length, int *arr, int *newarr, string& result)*

Рекурсивная функция.

Аргументы:

indent – индекс смещения для отображения глубины рекурсии.

length – длина исходного вектора.

arr – указатель на исходный массив типа int.

newarr – указатель на результирующий в процессе вычисления массив int.

Описание:

Как только мы попадаем в тело рекурсии, первым делом выводится на консоль нужное количество отступов(в зависимости от глубины рекурсии), определенное с помощью переменной *indent*, и сообщение о том что мы попали в рекурсию с *массивом arr*.

Далее есть 3 варианта исхода событий в текущем теле нашей рекурсии.

Первый случай при длине вектора *length* > 2. С таким вектором происходит следующее. Если длина его четна, то он разбивается ровно на пополам, и каждая его часть попадает в отдельную рекурсию с соответственно новыми аргументами (количеством отступов, длиной, указателем на исходный вектор и указателем на конечный верктор). Если же длина вектора нечетна, то он тоже разбивается на два и каждая из его частей так же попадает в рекурсию на уровень ниже, но части вектора теперь имеют разную длину, первый на 1 ед длины больше второго.

Во втором случае обрабатывается такой вектор, для которого длина *length* = 2. Сравнивая составные части (*length* = 1) данного вектора, записываем их в порядке возрастания в итоговый вектор newarr.

И в третьем случае мы имеем дело с вектором длиной *length = 1*. Его мы просто записываем на соответствующее место в векторе *newarr.*

Функция ***int main()***

Описание:

В главной функции программы происходит выделение блоков динамической памяти под интовые массивы исходного и конечного векторов, которые будут задействованы в ходе программы. В последующем за этим цикле идет считывание массива *arr*. И только теперь вектора попадают на обработку в функцию *rec*. На выходе из нее мы имеем конечный вектор newarr Он выводится на консоль и программа заканчивает свою работу.

Тестирование программы:

Входные данные	Результат работы программы
1 2 3 4 5	1 2 3 4 5
5 4 3 2 1	4 5 3 1 2
4 3 2	3 4 2
1	1

Выводы.

В ходе выполнения лабораторной работы получены, а также закреплены знания по теме «рекурсия».

ИСХОДНЫЙ КОД:

файл mainwindow.cpp:

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include "components.h"
#include "vec.h"

#include <thread>
#include <QApplication>
#include <QTextStream>

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
}

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::on_fileInput_clicked()
{
    QString input;
    QString fileName = QFileDialog::getOpenFileName(this,
                tr("open the *.txt"),  QDir::homePath(),
                tr("(*.txt);;All Files (*)"));
    QFile file(fileName);
    if(!file.open(QIODevice::ReadOnly))
        QMessageBox::information(0, "info", file.errorString());
    QTextStream in(&file);
    ui->inputLine->setText(in.readAll());
    checkInput();
}

void MainWindow::on_consoleInput_clicked()
{
    checkInput();
}

void MainWindow::checkInput(){
    QString input = ui->inputLine->text();
    QStringList list = input.split(" ");
    int* arr = new int[list.length()];
    int* newarr = new int[list.length()];
    string result;

    //check if string is unsupported
    for (int i = 0; i < input.size(); ++i){
        if((input[i].isLetter()) || (input[i].isPunct()) && (input[i] != '-')){
            QLabel *label = new QLabel("there isn't the number array!");
            label->show();
            label->resize(200, 200);
            ui->windowResult->clear();
            return;
        }
    }
```

```
        for (int i = 0; i < list.length(); ++i) {
            arr[i] = list[i].toInt();
        }

        rec(0, list.length(), arr, newarr, result);
        result.append("Changed vector:\n");
        for (int i = 0; i < list.length(); ++i) {
            result.append(to_string(newarr[i]) + " ");
        }

        ui->windowResult->setText(QString::fromStdString(result));
        ofstream resultFile("C:/a/result.txt", ios_base::out);
        resultFile << result;
        resultFile.close();

        delete[] arr;
        delete[] newarr;
    }
```

файл main.cpp:

```
#include "mainwindow.h"
#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    MainWindow w;
    w.show();
    return a.exec();
}
```

файл vec.cpp:

```
#include "basics.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>
#include <stdarg.h>
#include <QLabel>

#include "mainwindow.h"
#include "ui_mainwindow.h"
#include "components.h"
#include "vec.h"

#include <thread>
#include <QApplication>

void rec(int indent, int length, int *arr, int *newarr, string& result){
    for(int i = 0; i < indent; i++)
        result.append("\t");

    result.append("--> rec with vector: ");

    for(int i = 0; i < length; i++)
        result.append(to_string(arr[i]) + " ");
```

```
        result.append("\n");

        int halfl = length/2;
        if(length > 2){
            rec(indent + 1, length - halfl, arr, newarr, result);
            rec(indent + 1, halfl, arr + length - halfl, newarr + length - halfl,
    result);
        }
        if(length == 2){
            (newarr[0] = (arr[0] <= arr[1]) ? arr[0] : arr[1]);
            (newarr[1] = (arr[0] <= arr[1]) ? arr[1] : arr[0]);
        }
        if(length == 1)
            newarr[0] = arr[0];

        for(int i = 0; i < indent; i++)
            result.append("\t");

        result.append("<-- rec\n");
    }
```

файл mainwindow.h:

```
#ifndef MAINWINDOW_H
 #define MAINWINDOW_H

 #include <QMainWindow>

 QT_BEGIN_NAMESPACE
 namespace Ui { class MainWindow; }
 QT_END_NAMESPACE

 class MainWindow : public QMainWindow
 {
     Q_OBJECT

 public:
     void checkInput();
     MainWindow(QWidget *parent = nullptr);
     ~MainWindow();

 private slots:
     void on_fileInput_clicked();

     void on_consoleInput_clicked();

 private:
     Ui::MainWindow *ui;
 };
 #endif // MAINWINDOW_H
```

файл vec.h:

```
#ifndef TEXTANALYZE_H
#define TEXTANALYZE_H
#include "basics.h"
void rec(int ident, int length, int* arr, int* newarr, string& result);
#endif // TEXTANALYZE_H
```