Министерство образования и науки Российской Федерации

Севастопольский государственный университет

Кафедра ИС

Отчет

по лабораторной работе № 6

«Программирование алгоритмов

обработки одномерных массивов»

Выполнил

ст. гр. И12д

Серегин А.В.

Проверил:

асс. Забаштанский А.К.

Севастополь

2015

1. Цель работы

Изучение особенностей представления и обработки одномерных массивов в языках С/С++ с учетом связи указателей и массивов. Получение практических навыков реализации алгоритмов обработки одномерных массивов средствами С/С++. Исследование особенностей обработки одномерных динамических массивов.

2. Варианты задания

Требуется ввести с клавиатуры исходные данные, выполнить их обработку в соответствии с вариантом задания и вывести результаты обработки на экран. Варианты (по указанию преподавателя) выбирать либо из приведенных ниже, либо в соответствии с вариантами заданий к лабораторной работе №7 методических указаний [2].

Вариант 20:

Даны натуральные числа n, p, q и действительные числа a1, a2, …, an, причем n>=q>p>=1. Определить, имеются ли в последовательности a1, a2, …, aq три отрицательных члена, идущих подряд. Седи чисел ap, ap+1, …, an найти ближайшее к какому-нибудь целому.

Упорядочить члены последовательности ap, ap+1, …, aq по возрастанию, используя алгоритм сортировки методом прямого обмена.

3. Структурная схема

На рисунках ниже представлены структурные схемы алгоритма программы.



Рисунок 1 – Функция main.



a) б)

Рисунок 2 :

a) Функция ScanArray;

б) Функция PrintArray.

a) б)

Рисунок 3 :

a) Функция TreeNegative;

б) Функция Sort\_Bubble.



Рисунок 4 – Функция ReturnNearest.

4. Код программы

#include <iostream>

#include <iomanip>

using namespace std;

//чтение массива

void ScanArray(double \*a, const int n)

{

cout << "Введите массив:" << endl;

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

cin >> a[i];

}

//вывод массива

void PrintArray(double \*a, const int n)

{

cout << "Массив:" << endl;

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

cout << a[i] << ' ';

cout << endl;

}

//поиск 3-х подряд идущих отрицательных элементов

bool ThreeNegative(double \*a, const int n)

{

for (int i(0); i < n - 2; i++)

if (a[i] < 0 && a[i + 1 < 0 && a[i + 2] < 0]) return true;

return false;

}

//нахождение ближайшего к целому числа в массиве

double Int\_near(double \*a, const int p, const int n)

{

double nearest = a[p-1];

for (int i(p); i < n; i++)

if (abs(round(a[i]) - a[i]) < abs(round(nearest) - nearest)) nearest = a[i];

return nearest;

}

//сортирока пузырьком

void Sort\_Bubble(double \*a, const int p, const int n)

{

double temp;

for (int j(n-1); j > p - 1; j--)

for (int i(p - 1); i < j; i++)

if (a[i] > a[i + 1]) //обмен

{

temp = a[i];

a[i] = a[i + 1];

a[i + 1] = temp;

}

}

int main()

{

setlocale(LC\_ALL, "Russian");

int p, q, n;

cout << "Введите p, q, n:" << endl;

cin >> p >> q >> n;

double \*a = new double[n];

ScanArray(a, n);

cout.setf(ios::boolalpha);

cout << "Наличие 3х отрицательных элементов на промежутке 1-q: " <<

ThreeNegative(a, q) << endl;

cout << "Ближайший к целому на промежутке p-q: " << Int\_near(a, p, q) << endl;

Sort\_Bubble(a, p, n);

PrintArray(a, n);

system("pause");

return 0;

}

5. Тестовые примеры

На рисунке ниже представлен тестовый пример работы программы.

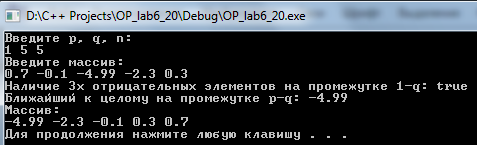


Рисунок 5 – тестовый пример программы.

Выводы

Массив – удобный способ для хранения однотипных типов данных. Плюсом массивов будет легкость нахождения элемента массива по его индексу, при том, время доступа ко всем элементам массива одинаковое. С помощью массивов можно с легкостью отсортировать ряд чисел по возрастанию(убыванию) или расположить символы или строки в алфавитном порядке.