## Лабораторная работа №3

Тема: Сортировка одномерных массивов.

Цель: освоение составления и тестирования алгоритмов и программ сортировки массивов с использованием функций пользователя.

Теоретические положения

### Параметры-массивы в функциях

Если мы хотим передать в подпрограмму отдельный элемент
массива, то в качестве соответствующего ему формального параметра указывается простая переменная того же типа.

Массивы, так же как и простые переменные, можно передавать в функции в качестве параметров. Так как имя массива – это адрес, то передача массива происходит всегда по адресу.

Рассмотрим, например, функцию, вычисляющую среднее значение элементов массива. Желательно сделать ее так, чтобы в нее можно было передавать массивы любого размера и она всегда правильно вычисляла результат. В языке С++ функции не могут самостоятельно определять размер массива, поэтому он должен быть обязательно одним из параметров.

#include "stdafx.h"

int Sum(int A[], int N)

{

int i, sum;

sum = 0;

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

sum += A[i];

return sum/N;

}

void main()

{

int x[5]={1,2,3,4,5},y[3]={11,22,33};

cout<<Sum(x,5)<<"\t"<<Sum(y,3));

}

### Сортировка обменом (пузырьком)

Сортировка простыми обменами, или сортировка пузырьком (англ. bubble sort) – простой алгоритм сортировки. Для понимания и реализации этот алгоритм – простейший, но эффективен он лишь для небольших массивов. Сложность алгоритма: О(N2).

Алгоритм состоит из повторяющихся проходов по сортируемому массиву. За каждый проход элементы последовательно сравниваются попарно и, если порядок в паре неверный, выполняется обмен элементов. Проходы по массиву повторяются N-1 раз или до тех пор, пока на очередном проходе не окажется, что обмены больше не нужны, что означает – массив отсортирован. При каждом проходе алгоритма по внутреннему циклу, очередной наибольший элемент массива ставится на своё место в конце массива рядом с предыдущим «наибольшим элементом», а наименьший элемент перемещается на одну позицию к началу массива («всплывает» до нужной позиции, как пузырёк в воде, отсюда и название алгоритма).

Введение индикатора (флажка F) действительно произошедших во внутреннем цикле обменов уменьшает число лишних проходов в случаях с частично отсортированными массивами на входе. Перед каждым проходом по внутреннему циклу флажок сбрасывается в 0, а после действительно произошедшего обмена устанавливается в 1. Если после выхода из внутреннего цикла флажок равен 0, то обменов не было, то есть массив отсортирован и можно досрочно выйти из программы сортировки.

Псевдокод ещё более улучшенного алгоритма с проверкой действительно произошедших обменов во внутреннем цикле (на входе: массив A[N], состоящий из N элементов, с нумерацией от A[1] до A[N]):

ЦИКЛ ДЛЯ J=1 ДО N-1 ШАГ 1

 F=0

 ЦИКЛ ДЛЯ I=1 ДО N-J ШАГ 1

 ЕСЛИ A[I] > A[I+1] ТО

 ОБМЕН A[I],A[I+1]

 F=1

 СЛЕДУЮЩЕЕ I

 ЕСЛИ F=0 ТО ВЫХОД ИЗ ЦИКЛА

СЛЕДУЮЩЕЕ J

Задание

1. Самостоятельно изучить методы сортировки массивов и возможности языка программирования C++ для сортировки одномерных массивов.
2. Создать консольное приложение, в котором реализовать следующие вычисления согласно индивидуальному заданию из табл. 3.1:
* объявление заданного массива целых чисел фиксированной длины;
* инициализацию элементов массива посредством **генерации случайных чисел в заданном диапазоне**;
* обменную (пузырьковую) сортировку элементов массива заданного направления, реализовать в виде пользовательской функции int Sort(int Ar[]), которая принимает в качестве аргумента исходный массив, а возвращает количество перестановок;
* вывести на экран исходный и отсортированный массив, а также количество перестановок; вывод массивов реализовать в виде пользовательской функции void Output(int Ar[]).
1. Выполнить контрольную пошаговую сортировку исходного массива из контрольного примера вручную.

Таблица 3.1. Варианты индивидуальных заданий:

| № варианта | Массив и длина | Диапазон значений | Направление сортировки |
| --- | --- | --- | --- |
| 2 | Х(12) | [-50, 10] | По убыванию |

По результатам выполнения практической работы сформировать отчет, который должен содержать следующие элементы:

* титульный лист типового образца;
* тема, цель, задание, в т.ч. индивидуальная часть;
* листинг программы;
* результаты выполнения программы;
* контрольная пошаговая сортировка исходного массива;
* выводы.