МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Отчет по лабораторной работе №1
по курсу «ЭВМ и периферийные устройства»

ОПРЕДЕЛЕНИЕ ВРЕМЕНИ РАБОТЫ ПРИКЛАДНЫХ ПРОГРАММ

Выполнил: студент 2-го курса гр. 17208
Гафиятуллин А.Р.

Новосибирск, 2018

1. ЦЕЛИ РАБОТЫ:

1. Изучение методики измерения времени работы подпрограммы;
2. Изучение приемов повышения точности измерения времени работы подпрограммы;
3. Изучение способов измерения времени работы подпрограммы;
4. Измерение времени работы подпрограммы в прикладной программе.

2. ХОД РАБОТЫ:

Для достижения поставленных целей был выбран 7 вариант задания:
Алгоритм сортировки методом пузырька. Дан массив случайных чисел длины N. На первой итерации попарно упорядочиваются все соседние элементы; на второй – все элементы, кроме последнего элемента; на третьей – все элементы, кроме последнего элемента и предпоследнего элемента и т.п.

Описание методики для определения времени работы программы:

1. Написана программа на языке C++, которая реализует алгоритм сортировки методом пузырька;

   Исходный код программы:

```
#include <iostream>
#include <vector>
#include <cstdlib>
#include <ctime>
#include <sys/times.h>
#include <unistd.h>
using namespace std;

template<typename T>
void bubble_sort(vector<T> &v)
{
    for(auto out_iter = v.end() - 1; out_iter != v.begin(); out_iter--)
```

```
        for(auto in_iter = v.begin(); in_iter != out_iter; in_iter++)
            if(*in_iter > *(in_iter + 1))
                swap(*in_iter, *(in_iter + 1));
}

int main()
{
    srand(time(NULL));
    long long int size;
    cin >> size;
    vector<int> v(size);
    for(auto &element : v)
        element = rand();

    struct tms start, finish;
    long clocks_per_sec = sysconf(_SC_CLK_TCK);
    times(&start);
    bubble_sort(v);
    times(&finish);
    double clocks = finish.tms_utime - start.tms_utime;
    cout << endl << "Total process time: " << (double)clocks / clocks_per_sec
        << "s" << endl;
    return 0;
}
```

Команда компиляции: g++ -std=c++11 main.cpp -o main

2. Проверена правильность работы программы на нескольких тестовых наборах входных данных:
    1. Вход: 81 78 76 35 29 81 52 93 22 67
       Выход: 22 29 35 52 67 76 78 81 81 93

2. Вход: 620 776 746 849 149 75 58 920 349 561 882 302 612 157 773
   Выход: 58 75 149 157 302 349 561 612 620 746 773 776 849 882 920
3. Вход: 9343 1879 3750 6152 4770 1050 3199 5646 2638 7844
   Выход: 1050 1879 2638 3199 3750 4770 5646 6152 7844 9343

3. На момент тестирования времени работы программы в Linux-машине с Elementary OS(Linux kernel 4.15.0-33-generic, Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz, оболочка Pantheon(X Windows System)) было запущено около 230-240 процессов, а так же была выполнена команда sync. Для более точного измерения времени работы алгоритма (без части кода, где вводятся данные) в многозадачной ОС, был использован таймер времени работы процесса times(), обрамляющий вызов функции сортировки методом пузырька. Тестовым путем было выясненно, что при текущей конфигурации компьютера и ОС, время работы алгоритма составляет порядка 15 секунд при сортировке около 29300 элементов, после чего был произведен 10-кратный запуск программы при заданном количестве элементов вектора, чтобы выяснить лучшее время работы алгоритма:
   1. Total process time: 15.11s
   2. Total process time: 15.14s
   3. Total process time: 15.05s
   4. Total process time: 15.11s
   5. Total process time: 15.07s
   6. Total process time: 15.1s
   7. Total process time: 15.1s
   8. Total process time: 15.12s
   9. Total process time: 15.11s
   10. Total process time: 15.12s
4. Наименьшее время работы функции сортировки – 15.05 секунд будем считать наиболее точным. Абсолютную погрешность times() оценим точностью этой функции, т.е не более 0.01 сек. в Linux. Относительную

погрешность выразим как отношение абсолютной погрешности к величине временного интервала работы функции сортировки:

$$\frac{0.01 сек.}{15.05 сек.} * 100\% = 0.066445183\% \approx 0.07\% < 1\%$$

Таким образом, время работы алгоритма сортировки методом пузырька при вводе 29300 элементов при указанных выше характеристиках компьютера составляет 15.05 секунд с относительной погрешностью 0.07%.

3. ВЫВОДЫ:

1. Были изучены методики измерения времени работы подпрограммы: относительная и абсолютная погрешность измерения;
2. Были изучены и использованы приемы повышения точности измерения времени работы подпрограммы: использование times() в многозадачных ОС, sync, многократные замеры показаний таймера, исключение из измерения стадий инициализации и завершения;
3. Были изучены способы измерения времени работы подпрограммы;
4. На примере сортировки методом пузырька научились измерять время работы подпрограммы в прикладной программе.