Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики

Кафедра Информатики и Прикладной Математики

Дисциплина: Системное программное обеспечение

Лабораторная работа №1

Выполнил Григорьев Г.Г, гр. P3217

Преподаватель: Зыков А.Г.

Санкт-Петербург, 2018 г.

Задание 1.

Создайте файл, содержащий массив структур, каждая из которых содержит информацию о Ваших результатах за три семестра (фамилия, семестр, дисциплина, оценка).

Реализуйте простое последовательное копирование содержимого файла тремя различными способами:

1. С использованием библиотеки С.
2. С использованием Windows.
3. С использованием вспомогательной функции Windows — CopyFile.

Результат вывести на печать рез-ты копирования и средний балл.

Для анализа способов копирования создайте произвольный файл большого размера (экспериментально увеличивая для получения наглядных результатов) и определите временные характеристики каждого из способов. Результаты выведите в виде таблицы.

Проведите сравнительный анализ способов. Достоинства и недостатки.

Текстовый файл с информацией о предметах:

Георгий Григорьев Мат.анализ 3
Георгий Григорьев Алгебра 4
Георгий Григорьев Программирование 5
Георгий Григорьев Мат.анализ 3
Георгий Григорьев Мат.анализ 3
Георгий Григорьев Программирование 5
Георгий Григорьев ЭВМ 5
Георгий Григорьев Дискр.математика 4
Георгий Григорьев ЯзыкиПрограммирования 5
Георгий Григорьев Мат.анализ 4
Георгий Григорьев Комп.графика 5
Георгий Григорьев Теор.автоматов 4

Занимаемое место: 727 байт

Программа 1. Декларация функций для копирования файлов

```
#include <Windows.h>
#include <stdio.h>
#define BUF_SIZE 256
int func1(string in, string out)
{
     char buffer[BUF_SIZE];
     FILE *from, *to;
     from = fopen(in.c_str(), "rb");
     to = fopen(out.c_str(), "wb");
     size_t bytes_in, bytes_out;
     while (bytes_in = fread(buffer, 1, buffer_size, from)
> 0)
     {
          bytes_out = fwrite(buffer, 1, bytes_in, to);

          if (bytes_out != bytes_in)
          {
               perror("Error writing to file.");
               return 1;
          }
     }

     fclose(from);
     fclose(to);
}

int func2(string in, string out)
```

```
{
     HANDLE hIn, hOut;
     DWORD nIn, nOut;
     CHAR buffer[BUF_SIZE];

     hIn = CreateFile(std::wstring(in.begin(),
in.end()).c_str(),
          GENERIC_READ, 0, NULL, OPEN_EXISTING, 0, NULL);
     if (hIn == INVALID_HANDLE_VALUE)
     {
          printf("Unable to open input file. Error: %õ\n",
GetLastError());
          return 2;
     }
     hOut = CreateFile(std::wstring(out.begin(),
out.end()).c_str(),
          GENERIC_WRITE, 0, NULL, CREATE_ALWAYS,
FILE_ATTRIBUTE_NORMAL, NULL);
     if (hOut == INVALID_HANDLE_VALUE)
     {
          printf("Unable to open output file. Error: %x\n",
GetLastError());
          return 3;
     }

     while (ReadFile(hIn, buffer, buffer_size, &nIn, NULL)
&& nIn > 0)
     {
          WriteFile(hOut, buffer, nIn, &nOut, NULL);
          if (nIn != nOut)
          {
               printf("Error writing to file: %x\n",
GetLastError());
               return 4;
          }
     }

     CloseHandle(hIn);
     CloseHandle(hOut);
}

int func3(string in, string out)
{
     if (!CopyFile(std::wstring(in.begin(),
in.end()).c_str(),
          std::wstring(out.begin(), out.end()).c_str(),
FALSE))
     {
          printf("Error executing function CopyFile: %x\n",
GetLastError());
          return 1;
     }
```

```
    return 0;
}
```

Программа 2. Создание массива данных и вычисление среднего балла

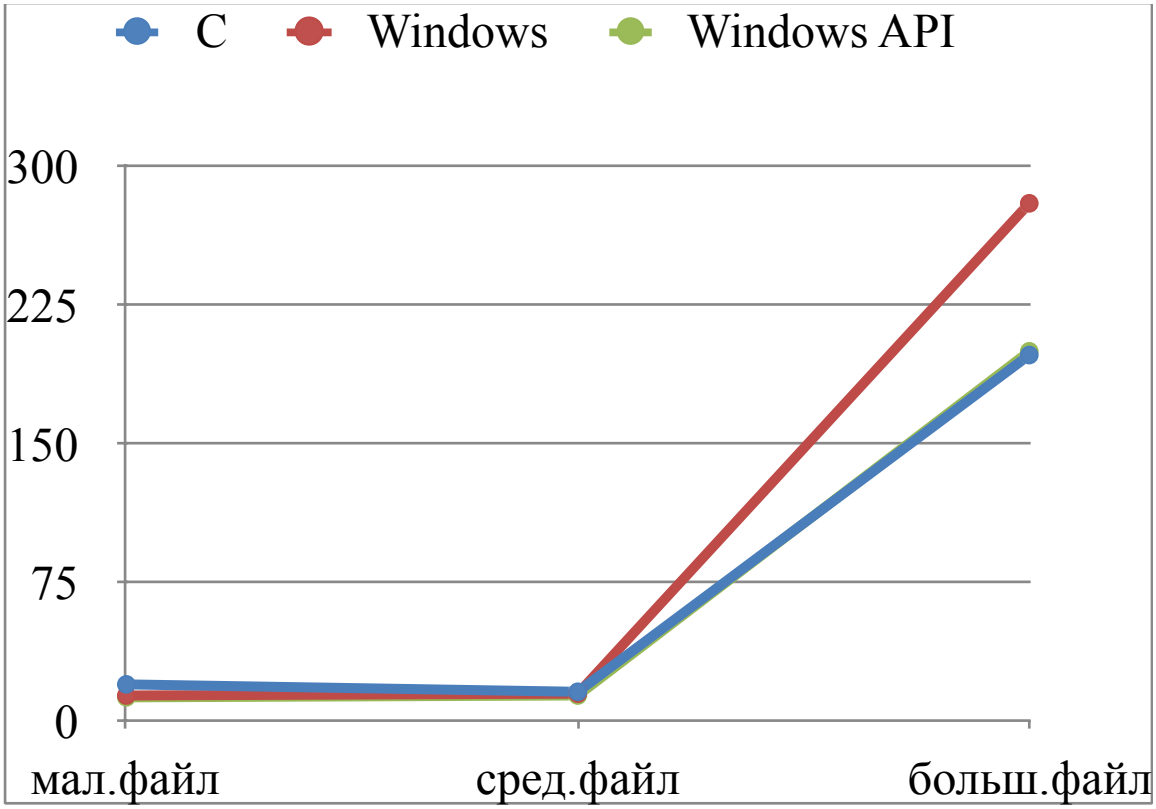
```
struct Result
{
    string name;
    string surname;
    string discipline;
    double score = 0;
};

double calculateAVR(string in)
{
    vector<Result> results;
    ifstream input(in, ios_base::in);

    int results_counter = 0;
    double scores_sum = 0;

    while (!input.eof())
    {
        Result temp;
        input >> temp.name >> temp.surname >>
temp.discipline >> temp.score;

        scores_sum += temp.score;
        ++results_counter;
    }
    input.close();

    return (scores_sum) / results_counter;
}
```

Программа 3. Замеры времени на исполнение функций копирования

```
#include <iostream>
#include <fstream>
#include <string>
#include <chrono>
#include <vector>
#include "cpC.h"
#include "cpW.h"
#include "cpCF.h"
using namespace std;
using namespace chrono;
```

```
using chrono::high_resolution_clock;
int main(int argc, char* argv[])
{
    string first_path = argv[0];
    string second_path = argv[1];
    high_resolution_clock::time_point start, end;

    start = high_resolution_clock::now();
    func1(first_path, second_path);
    end = high_resolution_clock::now();
    duration<double> time_custom =
duration_cast<duration<double>>(end - start);

    start = high_resolution_clock::now();
    func2(first_path, second_path);
    end = high_resolution_clock::now();
    duration<double> time_windows_custom =
duration_cast<duration<double>>(end - start);

    start = high_resolution_clock::now();
    func3(first_path, second_path);
    end = high_resolution_clock::now();
    duration<double> time_windows_builtin =
duration_cast<duration<double>>(end - start);

    cout << endl;
    cout << "File in: " << first_path << endl;
    cout << "File out: " << second_path << endl;
    cout << "Custom copy resulted in:         \t" <<
time_custom.count() << endl;
    cout << "Windows custom copy resulted in: \t" <<
time_windows_custom.count() << endl;
    cout << "Windows builtin copy resulted in:\t" <<
time_windows_builtin.count() << endl;
    cout << endl;
    cout << "Calculated average score: " <<
calculateAVR(first_path);

    return 0;
}
```

Были созданы 3 файла путем копирования строк в изначальном файле. Размеры файлов для тестирования 727 байт, 727 Кбайт, 72,7 Мбайт (в 1000 и 100000 раз больше оригинального файла)



Данные 3 запусков были усреднены и занесены в таблицу, построены графики сравнения.

Вывод:

Скорость копирования маленьких файлов быстрее средствами Windows, средние файлы всеми средствами копируются одинаково. Большие файлы средствами C или готовыми функциями копируются сравнительно быстрее.