Министерство образования и науки РФ

Севастопольский государственный университет

Отчёт

по лабораторной №2

**«WINAPI»**

Выполнил:

ст.гр.ИСб-22д

Воронин И.Ю.

Проверил:

Волкова А.В

Севастополь

2015

1.Цель работы

Изучение возможности использования программного интерфейса приложений(API) операционных систем Windows 95-2000, NT, XP, 7. Приобретение практических навыков создания и управления процессами и потоками, используя Win API в средах программирования Borland Delphi, C++ Builder или Visual Studio.

2.Вариант задания

* 1. Написать программу **Sort**, реализующую следующий алгоритм:
     1. Зафиксировать время начала **Tstrt** выполнения программы (пример получения текущего системного времени и вычисления разности между двумя значениями времени приведен в программе TimeDifference, представленной в Приложении).
     2. Вывести на экран время **Tstrt** в формате: минуты:секунды:миллисекунды;
     3. Зафиксировать текущий момент времени t1;
     4. Для **i** от 1 до 100(количество повторений может быть изменено в зависимости от быстродействия процессора) повторять:
        1. Заполнить массив целых чисел случайными значениями из диапазона 0-10000;
        2. Отсортировать массив;
     5. Зафиксировать текущий момент времени t2;
     6. Определить среднее время одной сортировки: (t2-t1)/100;
     7. Вывести на экран среднее время одной сортировки(в миллисекундах);
     8. Зафиксировать время окончания **Tend** выполнения программы;
     9. Вывести на экран время **Tend** в формате: минуты:секунды: миллисекунды;

Язык программирования, метод и направление сортировки, а также количество элементов массива N выбирается в соответствии с вариантом задания, приведенным в таблице 3.1.

* 1. Написать программу Master, выполняющую следующие действия:

3.2.1. Для **i** от 1 до 3 повторять:

3.2.2.1. Используя системные вызовы **CreateProcess,** создать два процесса **Sort** склассами приоритетов, в соответствии с вариантом задания (таблица 3.1). Необходимо, чтобы каждый процесс имел собственную консоль и окно консоли имело заголовок: "Process: *NP*; Prioritet: *PP*", где *NP* – номер процесса (1 или 2), а *PP* – приоритет соответствующего процесса. (Для изменения свойств окна консоли использовать структуру STARTUPINFO).

3.2.2.2. Ожидать окончания процессов, созданных в п. 3.2.2.1 (использовать функцию WaitForSingleObject).

* 1. Зафиксировать для отчета значения времени, получаемые при выполнении процессов в п 3.2.1.
  2. Написать программу **Threads,** содержащую процедуру сортировки массива (разработанную при выполнении пункта 3.1), содержащего N/50 элементов и процедуру вывода массива на экран – **mass\_print**. Программа должна выполнять следующие действия:
     1. Генерировать случайный массив, содержащий N/50 элементов.
     2. Используя системные вызовы **CreateThread,** создать программные потоки **sort** и **mass\_print** в приостановленном состоянии.
     3. Установить приоритеты потоков в THREAD\_PRIORITY\_NORMAL, используя системный вызов **SetThreadPriority.**
     4. Активизировать потоки, используя системные вызовы **ResumeThread**;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Visual Studio | Быстрая | Убыв. | 5500 | 3-1, 1-1, 3-3 |

3.Текст программы

#include <windows.h>

#include <iostream>

using namespace std;

//MASTER

int main(){

std::cout<<"HI";

STARTUPINFO info1,info2;

memset(&info1, 0, sizeof(info1));

info1.cb = sizeof(info1);

info1.lpTitle ="Process 1 ; Prioritet 3.";

memset(&info2, 0, sizeof(info2));

info2.cb = sizeof(info1);

info2.lpTitle = "Process 2 ; Prioritet 1.";

PROCESS\_INFORMATION pinfo1, pinfo2;

CreateProcess(0,"OS2.exe",NULL,NULL,TRUE,

CREATE\_NEW\_CONSOLE | IDLE\_PRIORITY\_CLASS,NULL,NULL,&info1,&pinfo1);

CreateProcess(0,"OS2.exe",NULL,NULL,TRUE,

CREATE\_NEW\_CONSOLE | HIGH\_PRIORITY\_CLASS,NULL,NULL,&info2,&pinfo2);

SetProcessAffinityMask(pinfo1.hProcess,1);

SetProcessAffinityMask(pinfo2.hProcess,2);

WaitForSingleObject(pinfo1.hProcess,INFINITE);

WaitForSingleObject(pinfo2.hProcess,INFINITE);

std::cout<<"HI";

system("PAUSE");

#include <Windows.h>

#include <iostream>

#include <algorithm>

using namespace std;

int arr[110]={0};

void qs(int\* s\_arr, int first, int last);

void mass\_print(int \*arr,int n);

///////////////////////////////////////////

DWORD WINAPI TH2(const LPVOID param){

mass\_print(arr,109);

return 0;

}

DWORD WINAPI TH1(const LPVOID param){

qs(arr,0,109);

return 0;

}

///////////////////////////////////////////

int main(){

for(int i=0; i <110; i++){

arr[i] = abs((int)(rand()%10000));

}

HANDLE T1 = CreateThread(NULL,0,&TH1,0,CREATE\_SUSPENDED,0);

HANDLE T2 = CreateThread(NULL,0,&TH2,0,CREATE\_SUSPENDED,0);

SetThreadPriority(T1,THREAD\_PRIORITY\_BELOW\_NORMAL);

SetThreadPriority(T2,THREAD\_PRIORITY\_ABOVE\_NORMAL);

ResumeThread(T1);

ResumeThread(T2);

WaitForSingleObject(T1,INFINITE);

WaitForSingleObject(T2,INFINITE);

system("PAUSE");

return 0;

}

void qs(int\* s\_arr, int first, int last)

{

int i = first, j = last, x = s\_arr[(first + last) / 2];

do {

while (s\_arr[i] > x) i++;

while (s\_arr[j] < x) j--;

if (i <= j) {

if (s\_arr[i] < s\_arr[j]) swap(s\_arr[i], s\_arr[j]);

i++;

j--;

}

} while (i <= j);

if (i < last)

qs(s\_arr, i, last);

if (first < j)

qs(s\_arr, first, j);

}

void mass\_print(int \*arr,int n){

cout << endl;

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

cout << arr[i] << ' ';

}

cout << endl;

}

}

// OS2.cpp : Defines the entry point for the console application.

#include <windows.h>

#include <iostream>

#include <ctime>

#include <algorithm>

using namespace std;

void qs(int\* &s\_arr, int first, int last);

int main(int argc, char\* argv[])

{

int \*arr = new int[5500];

SYSTEMTIME t1,t2,t3;

GetSystemTime(&t1);

cout << "t1 "<<t1.wMinute << " : " << t1.wSecond << " : " << t1.wMilliseconds << endl;

for (int i = 0; i < 100; i++){

for (int i = 0; i < 5500; i++){

arr[i] = rand() % 10000;

}

qs(arr, 0, 5500 - 1);

}

GetSystemTime(&t2);

cout << "t2 " << t2.wMinute << " : " << t2.wSecond << " : " << t2.wMilliseconds << endl;

cout << "Srednee " << (((t2.wSecond \* 1000) + t2.wMilliseconds) - ((t1.wSecond \* 1000) + t1.wMilliseconds)) / 100 << "ms" << endl;

delete[] arr;

GetSystemTime(&t3);

cout << "t3 " << t3.wMinute << " : " << t3.wSecond << " : " << t3.wMilliseconds << endl;

system("PAUSE");

return 0;

}

void qs(int\* &s\_arr, int first, int last)

{

int i = first, j = last, x = s\_arr[(first + last) / 2];

do {

while (s\_arr[i] > x) i++;

while (s\_arr[j] < x) j--;

if (i <= j) {

if (s\_arr[i] < s\_arr[j]) swap(s\_arr[i], s\_arr[j]);

i++;

j--;

}

} while (i <= j);

if (i < last)

qs(s\_arr, i, last);

if (first < j)

qs(s\_arr, first, j);

}

4.Тест программы

Запуск двух функций CreateProces(….)(рис.4.1)

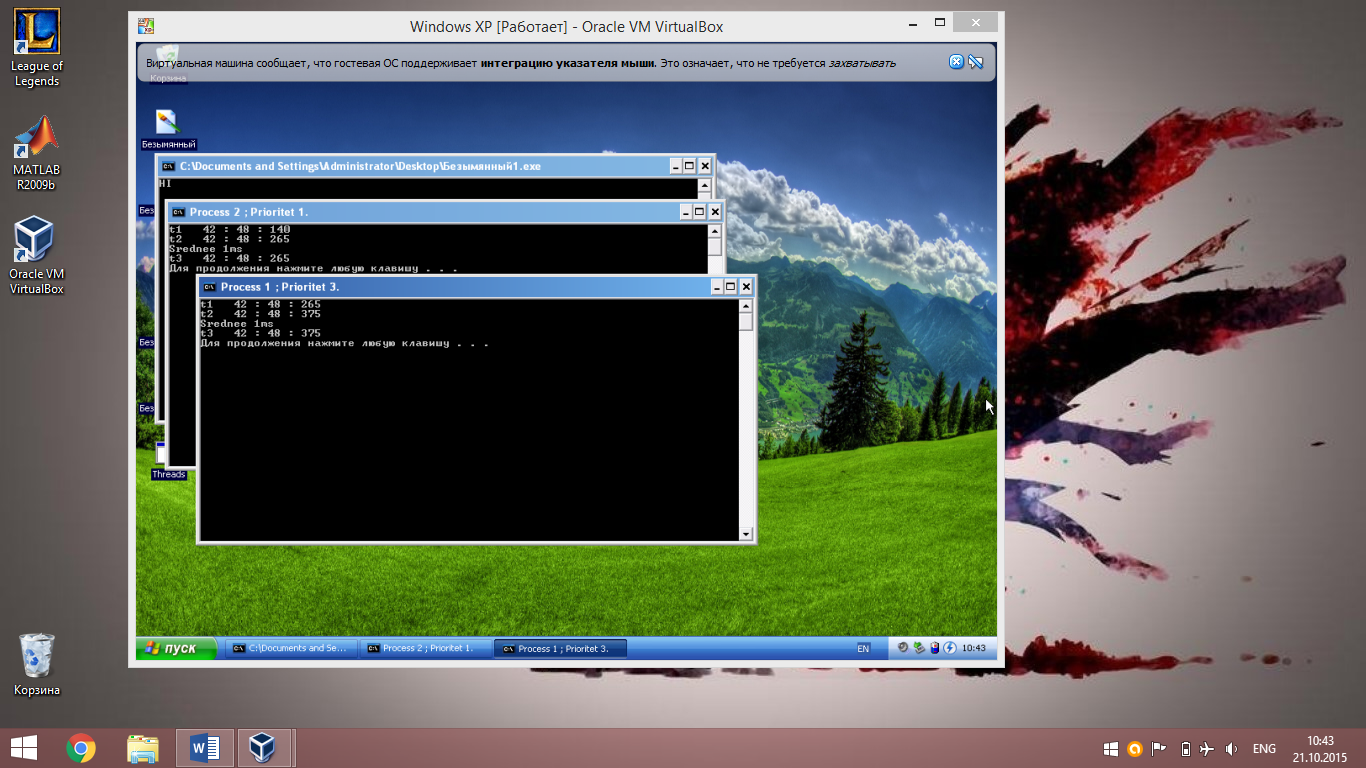


Рисунок 4.1.-CreateProcess.

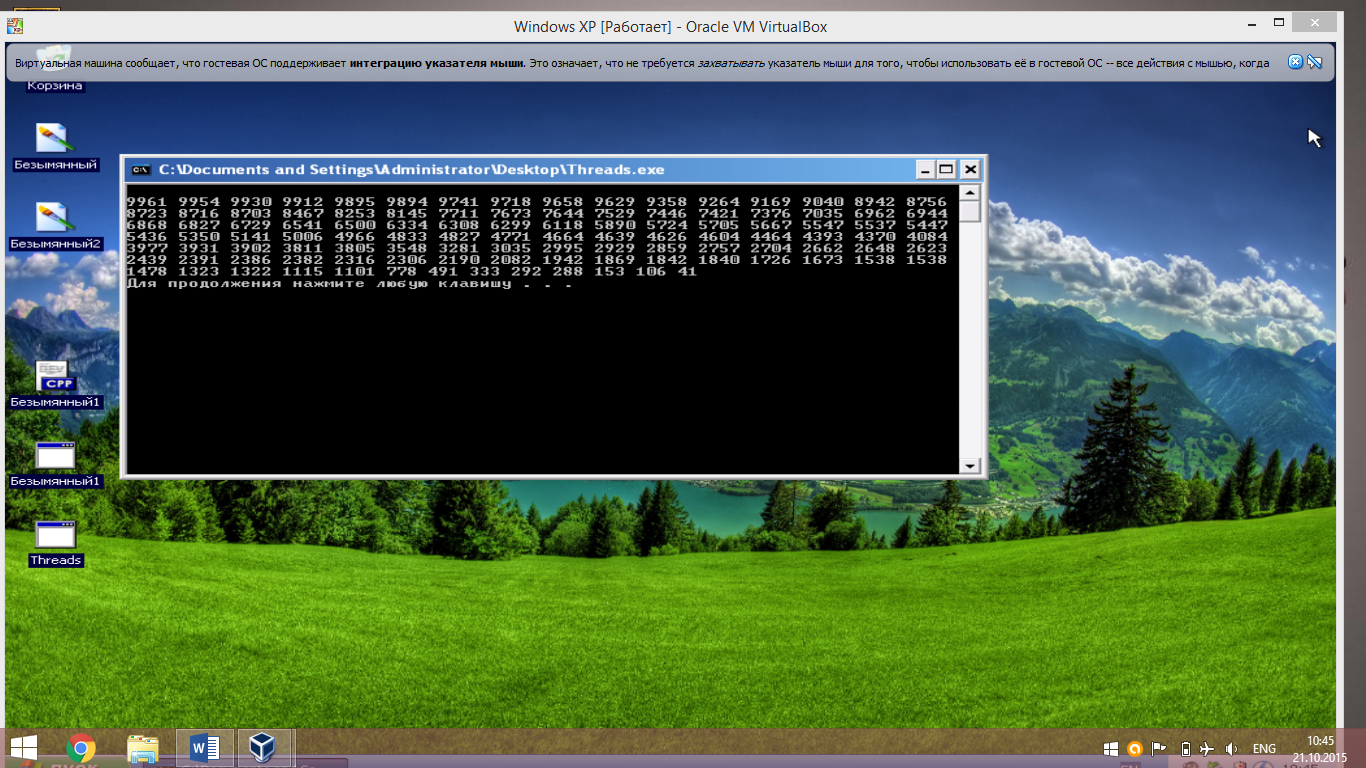


Рисунок 4.2.-CreateProcess.

ВЫВОДЫ

В данной лабораторной работе были изучены основные функции работы с процессами и потоками в Windows. Была использована WINAPI. Были проведены тесты в операционной системе на одноядерной виртуальной машине.