Университет ИТМО

Кафедра прикладной математики и информатики

Системное программное обеспечение

Лабораторная работа № 4

Выполнил:

Сорокин Юрий

Группа Р3217

Преподаватель:

Зыков А.Г.

СПб

2016 г.

# Лабораторная работа № 4

#include "windows.h"

#include "stdio.h"

#include "utils.h"

#define FILE1\_NAME "1.txt"

#define FILE2\_NAME "2.txt"

#define FILE3\_NAME "3.txt"

#define FILE\_SIZE 128\*1024

#define MIN\_KEY 1

#define MAX\_KEY 5\*1024\*1024

#define MIN\_VALUE\_LEN 5

#define MAX\_VALUE\_LEN 25

void printProcessInfo(int i, PROCESS\_INFORMATION info) {

FILETIME creationTime, exitTime, kernelTime, userTime;

SYSTEMTIME sysCreationTime, sysExitTime, sysKernelTime, sysUserTime;

GetProcessTimes(info.hProcess, &creationTime, &exitTime, &kernelTime, &userTime);

FileTimeToSystemTime(&creationTime, &sysCreationTime);

ULARGE\_INTEGER bufLargeInteger;

printf("\n\n\n============\nProcess %d info\n", i);

printf("Process ID: %ld\n", info.dwProcessId);

long handleCount;

printf("Creation date: %d/%d/%d %d:%d:%d\n", sysCreationTime.wDay, sysCreationTime.wMonth,

sysCreationTime.wYear, sysCreationTime.wHour, sysCreationTime.wMinute, sysCreationTime.wMilliseconds);

bufLargeInteger.LowPart = creationTime.dwLowDateTime;

bufLargeInteger.HighPart = creationTime.dwHighDateTime;

printf("\t%lld mks\n", bufLargeInteger.QuadPart / 10);

FileTimeToSystemTime(&exitTime, &sysExitTime);

printf("Exit date: %d/%d/%d %d:%d:%d\n", sysExitTime.wDay, sysExitTime.wMonth,

sysExitTime.wYear, sysExitTime.wHour, sysExitTime.wMinute, sysExitTime.wMilliseconds);

bufLargeInteger.LowPart = exitTime.dwLowDateTime;

bufLargeInteger.HighPart = exitTime.dwHighDateTime;

printf("\t%lld mks\n", bufLargeInteger.QuadPart / 10);

bufLargeInteger.LowPart = kernelTime.dwLowDateTime;

bufLargeInteger.HighPart = kernelTime.dwHighDateTime;

printf("Kernel time: %lld mks\n", bufLargeInteger.QuadPart / 10);

bufLargeInteger.LowPart = userTime.dwLowDateTime;

bufLargeInteger.HighPart = userTime.dwHighDateTime;

printf("User time: %lld mks\n", bufLargeInteger.QuadPart / 10);

printf("============\n");

}

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

generateFile(FILE1\_NAME, FILE\_SIZE, MIN\_KEY, MAX\_KEY, MIN\_VALUE\_LEN, MAX\_VALUE\_LEN);

generateFile(FILE2\_NAME, FILE\_SIZE, MIN\_KEY, MAX\_KEY, MIN\_VALUE\_LEN, MAX\_VALUE\_LEN);

generateFile(FILE3\_NAME, FILE\_SIZE, MIN\_KEY, MAX\_KEY, MIN\_VALUE\_LEN, MAX\_VALUE\_LEN);

int key;

sscanf(argv[1], "%d", &key);

char cmd[1024];

sprintf(cmd, "sort\_find %s %d", FILE1\_NAME, key);

STARTUPINFO stup;

GetStartupInfo(&stup);

PROCESS\_INFORMATION firstProcessInformation, secondProcessInformation, thirdProcessInformation;

printf("==============\nPROCESS 1\n==============\n");

CreateProcess(NULL, cmd, NULL, NULL, FALSE, NORMAL\_PRIORITY\_CLASS, NULL, NULL, &stup, &firstProcessInformation);

WaitForSingleObject(firstProcessInformation.hProcess, INFINITE);

sprintf(cmd, "sort\_find %s %d", FILE2\_NAME, key);

printf("\n==============\nPROCESS 2\n==============\n");

CreateProcess(NULL, cmd, NULL, NULL, FALSE, NORMAL\_PRIORITY\_CLASS, NULL, NULL, &stup, &secondProcessInformation);

WaitForSingleObject(secondProcessInformation.hProcess, INFINITE);

sprintf(cmd, "sort\_find %s %d", FILE3\_NAME, key);

printf("\n==============\nPROCESS 3\n==============\n");

CreateProcess(NULL, cmd, NULL, NULL, FALSE, NORMAL\_PRIORITY\_CLASS, NULL, NULL, &stup, &thirdProcessInformation);

WaitForSingleObject(thirdProcessInformation.hProcess, INFINITE);

printProcessInfo(1, firstProcessInformation);

printProcessInfo(2, secondProcessInformation);

printProcessInfo(3, thirdProcessInformation);

CloseHandle(firstProcessInformation.hThread);

CloseHandle(firstProcessInformation.hProcess);

CloseHandle(secondProcessInformation.hThread);

CloseHandle(secondProcessInformation.hProcess);

CloseHandle(thirdProcessInformation.hThread);

CloseHandle(thirdProcessInformation.hProcess);

return 0;

}

**Вывод**

При выполнении данной работы я ознакомился с созданием и работой с процессами.