Федеральное агентство связи

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

Кафедра прикладной математики и кибернетики

Отчет по лабораторной работе

по дисциплине «Операционные системы»

Лабораторная работа №2

Выполнил: студент 3 курса группы ИП-811 Мироненко К. А.

Проверил: профессор кафедры ПМиК Малков Е. А.

**ОГЛАВЛЕНИЕ**

[1. Постановка задачи 3](#_Toc53520694)

[2. Описание работы программы 4](#_Toc53520695)

[3. Примеры работы программы 6](#_Toc53520696)

[*Приложение* Листинг 7](#_Toc53520697)

# Постановка задачи

**Цель:** знакомство с работой компилятора Visual C++ cl в командной строке и знакомство с документацией Windows API в MSDN.

**Упражнение 1.** Программно определить пути к системному каталогу Windows и каталогуmвременных файлов Windows, используя следующие функции Win32 API:

* UINT GetWindowsDirectory( LPTSTR lpBuffer, UINT uSize );
* DWORD GetTempPath( DWORD nBufferLength, LPSTR lpBuffer )

**Упражнение 2.** Используя функции Windows API получить системную информацию: количество ядер процессора, архитектуру процессора, нижнюю и верхнюю границы пользовательского адресного пространства (ключевое слово для поиска в MSDN – GetSystemInfo).

**Упражнение 3.** Используя функции Windows API получить информацию об использовании физической и виртуальной памяти (ключевое слово для поиска в MSDN – GlobalMemoryStatus ).

**Примечание:** использовать 32- и 64- разрядные версии компилятора.

# Описание работы программы

Используемые структуры:

* SYSTEM\_INFO

typedef struct \_SYSTEM\_INFO {

union {

DWORD dwOemId;

struct {

WORD wProcessorArchitecture;

WORD wReserved;

} DUMMYSTRUCTNAME;

} DUMMYUNIONNAME;

DWORD dwPageSize;

LPVOID lpMinimumApplicationAddress;

LPVOID lpMaximumApplicationAddress;

DWORD\_PTR dwActiveProcessorMask;

DWORD dwNumberOfProcessors;

DWORD dwProcessorType;

DWORD dwAllocationGranularity;

WORD wProcessorLevel;

WORD wProcessorRevision;

} SYSTEM\_INFO, \*LPSYSTEM\_INFO;

Структура **SYSTEM\_INFO** содержит информацию о текущей компьютерной системе, включая архитектуру и тип процессора, число процессоров в системе и т.д.

* MEMORYSTATUSEX

typedef struct \_MEMORYSTATUSEX {

DWORD dwLength;

DWORD dwMemoryLoad;

DWORDLONG ullTotalPhys;

DWORDLONG ullAvailPhys;

DWORDLONG ullTotalPageFile;

DWORDLONG ullAvailPageFile;

DWORDLONG ullTotalVirtual;

DWORDLONG ullAvailVirtual;

DWORDLONG ullAvailExtendedVirtual;

} MEMORYSTATUSEX, \*LPMEMORYSTATUSEX;

Структура **MEMORYSTATUSEX** содержит информацию о памяти компьютера (общей, доступной и виртуальной). (Все значения хранятся как 64-битные числа)

Используемые функции:

* GetWindowsDirectory

**UINT** **GetWindowsDirectory(**

**LPTSTR** lpBuffer, // адрес буфера

**UINT** uSize // размер буфера

**)**;

Функция GetWindowsDirectory получает путь к директории Windows. Директория Windows содержит такие файлы, как приложения, файлы инициализации и файлы помощи.

* GetTempPath

**DWORD GetTempPath(**

**DWORD** *nBufferLength***,** // размер буфера

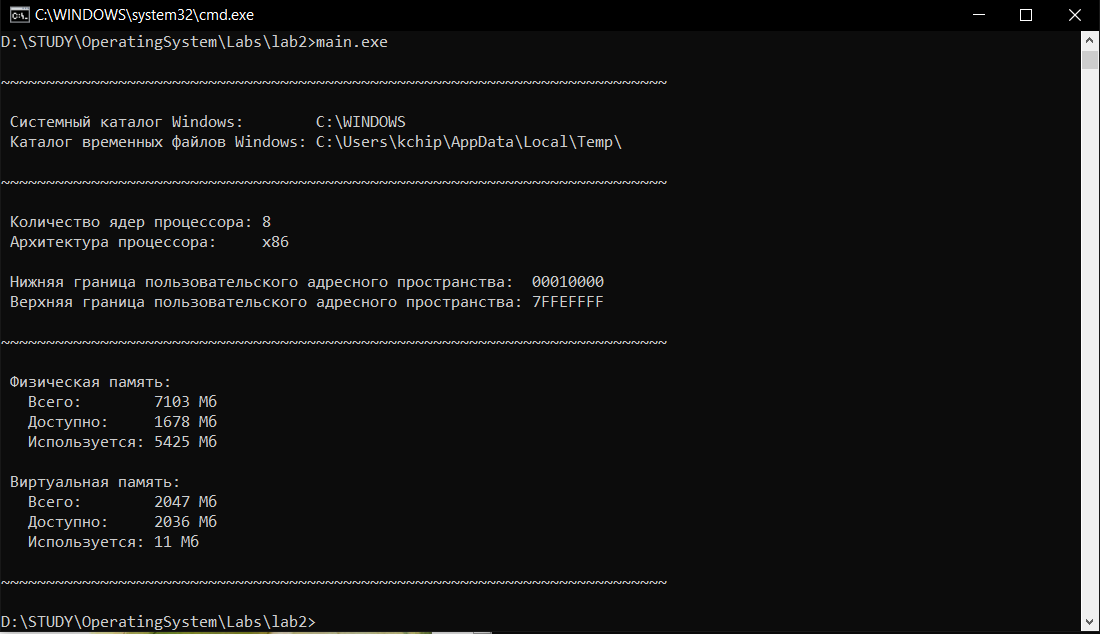
**LPTSTR** *lpBuffer* // буфер пути

**);**

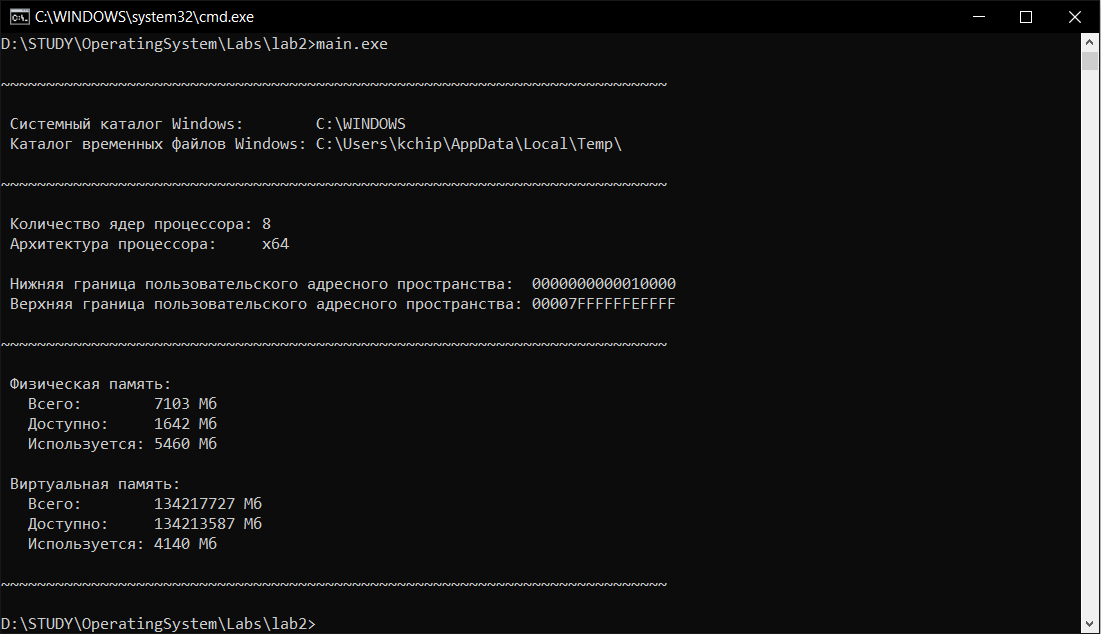
Функция GetTempPath извлекает путь к каталогу, назначенному для временных файлов.

* GetSystemInfoФункция GetSystemInfo получает информацию о текущей системе.
* GlobalMemoryStatusExФункция GlobalMemoryStatusEx получает информацию о текущей памяти компьютера (физической и виртуальной).

# Примеры работы программы



(Результат работы 64-битной версии программы)



(Результат работы 64-битной версии программы)

# Приложение. Листинг

**main.c**

#include <windows.h>

#include <stdio.h>

#include <locale.h>

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

{

setlocale(LC\_ALL, "");

char systemPath[MAX\_PATH], tempPath[MAX\_PATH];

SYSTEM\_INFO systemInformation;

MEMORYSTATUSEX memory;

memory.dwLength = sizeof(memory);

GetWindowsDirectory(systemPath, MAX\_PATH);

GetTempPath(MAX\_PATH, tempPath);

GetSystemInfo(&systemInformation);

GlobalMemoryStatusEx(&memory);

printf("\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");

printf("\n Системный каталог Windows: %s", systemPath);

printf("\n Каталог временных файлов Windows: %s", tempPath);

printf("\n\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");

printf("\n Количество ядер процессора: %d", systemInformation.dwNumberOfProcessors);

printf("\n Архитектура процессора: ");

switch (systemInformation.wProcessorArchitecture)

{

case 0:

printf("x86");

break;

case 9:

printf("x64");

break;

default:

printf("неизвестно");

break;

}

printf("\n\n Нижняя граница пользовательского адресного пространства: %p", systemInformation.lpMinimumApplicationAddress);

printf("\n Верхняя граница пользовательского адресного пространства: %p", systemInformation.lpMaximumApplicationAddress);

printf("\n\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");

printf("\n Физическая память:");

printf("\n Всего: %llu Мб", memory.ullTotalPhys / 1048576);

printf("\n Доступно: %llu Мб", memory.ullAvailPhys / 1048576);

printf("\n Используется: %llu Мб", (memory.ullTotalPhys - memory.ullAvailPhys) / 1048576);

printf("\n\n Виртуальная память:");

printf("\n Всего: %llu Мб", memory.ullTotalVirtual / 1048576);

printf("\n Доступно: %llu Мб", memory.ullAvailVirtual / 1048576);

printf("\n Используется: %llu Мб", (memory.ullTotalVirtual - memory.ullAvailVirtual) / 1048576);

printf("\n\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");

return 0;

}