**ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1**

по дисциплине «Технические средства информатизации»

на тему:

«Получение информации об оборудовании»

Вариант №1

г. Санкт-Петербург

2018г.

Выполнил студент группы Y2338

ФИО Матвеенко Д.В.

Проверил:

Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_201\_г

**Министерство образования и науки Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ,**

**МЕХАНИКИ И ОПТИКИ»**

Факультет среднего профессионального образования

Цель работы – Получение информации об оборудованиии через API функции операционной системы.

Текст задания: разработать программу вывода информации на экран, используя функции Windows API (Linux API) или существующие команды оболочки. Вывести на экран количество занятой и свободной оперативной памяти. Программу разработать на C/C++ в виде приложения к ОС с графическим или текстовым интерфейсом.

Порядок выполнения работы.

1. Установлена IDE Visual Studio 2017 с предустановленными расширениями, для компиляции C++ кода в консольном приложении Windows.
2. Чтение документации Windows API, про возможности функции GlobalMemoryStatusEx для выполнения лабораторной работы.
3. Написание C++ кода для вывода текста о состоянии оперативной памяти в консольном приложении.
4. Компилирование и затем запуск программы, проверка корректности выводимых данных.

Текст исходной программы

#include "pch.h"

#include <iostream>

#include <windows.h>

#include <stdio.h>

#include <locale.h>

// Используется для перевода байтов в килобайты

#define DIV 1024

// Ширина поля для вывода чисел (Звездочка в спецификаторе формата "% \* I64d" принимает целое число)

#define WIDTH 8

int main() {

setlocale(LC\_ALL, "Russian");

MEMORYSTATUSEX statex;

statex.dwLength = sizeof(statex);

GlobalMemoryStatusEx(&statex);

printf("Используется %\*ld %% памяти.\n", WIDTH, statex.dwMemoryLoad);

printf("\n");

printf("Всего %\*I64d KB физической памяти.\n", WIDTH, statex.ullTotalPhys / DIV);

printf("Занято %\*I64d KB физической памяти.\n", WIDTH, statex.ullTotalPhys / DIV - statex.ullAvailPhys / DIV);

printf("Свободно %\*I64d KB физической памяти.\n", WIDTH, statex.ullAvailPhys / DIV);

printf("\n");

printf("Всего %\*I64d KB в файле подкачки.\n", WIDTH, statex.ullTotalPageFile / DIV);

printf("Занято %\*I64d KB в файле подкачки.\n", WIDTH, statex.ullTotalPageFile / DIV - statex.ullAvailPageFile / DIV);

printf("Свободно %\*I64d KB в файле подкачки.\n", WIDTH, statex.ullAvailPageFile / DIV);

printf("\n");

printf("Всего %\*I64d KB виртуальной памяти.\n", WIDTH, statex.ullTotalVirtual / DIV);

printf("Занято %\*I64d KB виртуальной памяти.\n", WIDTH, statex.ullTotalVirtual / DIV - statex.ullAvailVirtual / DIV);

printf("Свободно %\*I64d KB виртуальной памяти.\n", WIDTH, statex.ullAvailVirtual / DIV);

printf("\n");

// Показать количество доступной расширенной памяти.

printf("Свободно %\*I64d KB расширенной памяти.\n", WIDTH, statex.ullAvailExtendedVirtual / DIV);

}

Вывод по работе  
 Программа полностью выполняет требуемые условия задания, также выводит дополнительную информацию о состоянии оперативной памяти. Функция GlobalMemoryStatusEx из Windows API получает информацию о текущем использовании системой как физической, так и виртуальной оперативной памяти. Порядок загрузки программы: препроцессинг добавляет хэдеры в код (#include, #define), затем идет компиляция и ассемблирование. Процесс компоновки связывает все объектные файлы, статические библиотеки в единый исполняемый файл и при запуске программы, ЦП начинает выполнение кода с верхней части функции main(). Пример работы программы можно посмотреть ниже на рисунке 1.

  
*Рисунок 1 Пример работы программы*