Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Операционные системы

Студент: Трубач Д.С.

ФИТ 3 курс 5 группа

Преподаватель: Савельева М.Г.

Минск 2023

**Лабораторная работа №8**

Задание 1. Получите с помощью утилиты wmic информации об физической оперативной памяти компьютера, поясните эту информацию.

|  |  |
| --- | --- |
| wmic |  |

Получите с помощью утилиты powershell информации об физической оперативной памяти компьютера, поясните эту информацию.

|  |  |
| --- | --- |
| powershell |  |

Получите с помощью утилиты systeminfo информации об оперативной памяти компьютера, поясните эту информацию.

|  |  |
| --- | --- |
| systeminfo |  |

Получите с помощью утилиты performance monitor информации об оперативной памяти компьютера, поясните эту информацию.

|  |  |
| --- | --- |
| performance monitor |  |

Задание 2. Разработайте консольное приложение OS06\_02, выполняющее длинный цикл.

Продемонстрируйте с помощью отладчика адреса расположения модулей приложения OS08\_02.

Установите для приложения OS08\_02 стандартный адрес загрузки в память.

Продемонстрируйте с помощью отладчика стандартный адрес расположения модулей приложения OS08\_02

|  |  |
| --- | --- |
| Код программы OS08\_02 | #include <iostream>  #include <Windows.h>  using namespace std;  int main()  {  for (int i = 1; i <= 1000000; ++i)  {  cout << i << ". OS08\_02" << "\n";  Sleep(1000);  }  } |
| Скриншот запуска |  |
| Продемонстрируйте с помощью отладчика адреса расположения модулей приложения OS08\_02. |  |
| Установите для приложения OS08\_02 стандартный адрес загрузки в память. |  |
| Продемонстрируйте с помощью отладчика стандартный адрес расположения модулей приложения OS08\_02 |  |

Задание 3. Разработайте консольное приложение OS06\_03, выполняющее получение 256 страниц оперативной памяти.

Разместите в этой памяти массив типа int, полностью занимающее все 256 страниц.

Заполните этот массив нарастающей последовательностью чисел с шагом 1.

Запишите 3 первых буквы своей фамилии в 16-ричными числами в кодировке Windows-1251.

Найдите в полученной области памяти с помощью отладчика значение в байте, имеющем адрес вычисленный по следующему принципу: номер страницы = число в нулевом байте, смещение в странице = число 12 бит в 1ом и втором байтах.

Пример: Иванов

И = C8

в = E2

а = E0

Страница C8 = 200, смещение E2E = 3630

|  |  |
| --- | --- |
| Код программы OS08\_03 | #include <iostream>  #include <Windows.h>  using namespace std;  #define KB (1024)  #define MB (1024 \* KB)  #define PG (4 \* KB)  void saymem()  {  MEMORYSTATUS ms;  GlobalMemoryStatus(&ms);  cout << "Объём физической памяти: " << ms.dwTotalPhys / KB << " KB\n";  cout << "Доступно физической памяти: " << ms.dwAvailPhys / KB << " KB\n";  cout << "Объем виртуальной памяти: " << ms.dwTotalVirtual / KB << " KB\n";  cout << "Доступно виртуальной памяти: " << ms.dwAvailVirtual / KB << " KB\n\n";  }  /\*  Т - 210(10) - D2(16)  р - 240(10) - F0(16)  у - 243(10) - F3(16)  Страница D2 = 210  210 \* 4096 = 860160(10) = 0xC1F80 - добавить для перехода на страницу  F0F = (15 \* 16^2) + (0 \* 16^1) + (15 \* 16^0)  = (15 \* 256) + (0 \* 16) + (15 \* 1)  = 3840 + 0 + 15  = 3855  Смещение F0F = 3855(10) = 0xF0F  Искомое значение: начало массива + 0xC1F80 + 0xF0F  \*/  int main()  {  setlocale(LC\_ALL, "ru");  int pages = 256;  int countItems = pages \* PG / sizeof(int);  SYSTEM\_INFO system\_info;  GetSystemInfo(&system\_info);  cout << "\t Изначально в системе\n";  saymem();  LPVOID xmemaddr = VirtualAlloc(NULL, pages \* PG, MEM\_COMMIT, PAGE\_READWRITE); // выделено 1024 KB виртуальной памяти  cout << "\tВыделено " << pages \* PG / 1024 << " KB вирт. памяти\n";  saymem();  int\* arr = (int\*)xmemaddr;  for (int i = 0; i < countItems; i++)  arr[i] = i;  int\* byte = arr + 210 \* 1024 + 3855;  cout << "------ Значение памяти в байте: " << \*byte << " ------\n";  VirtualFree(xmemaddr, NULL, MEM\_RELEASE) ? cout << "\tВиртуальная память освобождена\n" : cout << "\tВиртуальная память не освобождена\n";  saymem();  } |
| Скриншот запуска |  |
| Запишите 3 первых буквы своей фамилии в 16-ричными числами в кодировке Windows-1251. |  |
| Найдите в полученной области памяти с помощью отладчика значение в байте, имеющем адрес вычисленный по следующему принципу: номер страницы = число в нулевом байте, смещение в странице = число 12 бит в 1ом и втором байтах. |  |

Задание 4. Разработайте консольное приложение OS08\_04, которое включает функцию sh, принимающую 1 параметр: дескриптор (HANDLE) heap.

Функция sh выводит на консоль, общий размер heap, размеры распределенной и нераспределенных областей памяти heap.

Приложение OS06\_04 размещает в стандартной heap процесса int-массив размерности 300000.

Выведите с помощью функции sh информацию до размещения массива и после.

Объясните результат.

|  |  |
| --- | --- |
| Код программы OS08\_04 | #include <iostream>  #include <Windows.h>  using namespace std;  void sh(HANDLE pheap)  {  PROCESS\_HEAP\_ENTRY phe;  phe.lpData = NULL;  while (HeapWalk(pheap, &phe))  {  // R - начало непрерывной области  // U - нераспределенная область  // B - распределенная область  cout << "-- address = " << showbase << hex << phe.lpData <<  ", size = " << dec << phe.cbData <<  ((phe.wFlags & PROCESS\_HEAP\_REGION) ? " R" : "") <<  ((phe.wFlags & PROCESS\_HEAP\_UNCOMMITTED\_RANGE) ? " U" : "") <<  ((phe.wFlags & PROCESS\_HEAP\_ENTRY\_BUSY) ? " B" : "") << "\n";  }  }  int main()  {  HANDLE pheap = GetProcessHeap();  cout << "\n================== BEFORE ==================\n\n";  sh(pheap);  int size = 300000;  int\* m = new int[size];  cout << "-- m = " << showbase << hex << m << ", size = " << dec << size << '\n';  cout << "\n\n================== AFTER ==================\n\n";  sh(pheap);  } |
| Скриншот запуска |  |

Задание 5. Разработайте консольное приложение OS08\_05 аналогичное приложению OS08\_04, но использующее пользовательскую heap, которая имеет первоначальный размер 4MB.

Объясните результат.

|  |  |
| --- | --- |
| Код программы OS08\_05 | #include <iostream>  #include <Windows.h>  using namespace std;  #define KB (1024)  void sh(HANDLE pheap)  {  PROCESS\_HEAP\_ENTRY phe;  phe.lpData = NULL;  while (HeapWalk(pheap, &phe))  {  // R - начало непрерывной области  // U - нераспределенная область  // B - распределенная область  cout << "-- address = " << showbase << hex << phe.lpData <<  ", size = " << dec << phe.cbData <<  ((phe.wFlags & PROCESS\_HEAP\_REGION) ? " R" : "") <<  ((phe.wFlags & PROCESS\_HEAP\_UNCOMMITTED\_RANGE) ? " U" : "") <<  ((phe.wFlags & PROCESS\_HEAP\_ENTRY\_BUSY) ? " B" : "") << "\n";  }  }  int main()  {  // Параметры пользовательской кучи:  // 1. доступ не синхронизирован  // 2. куча заполняется нулями  // 3. начальный размер 4 мб  // 4. конечный размер ограничен размером виртуальной памяти  HANDLE heap = HeapCreate(HEAP\_NO\_SERIALIZE | HEAP\_ZERO\_MEMORY, 4096 \* 1024, 0);  sh(heap);  int\* x1 = (int\*)HeapAlloc(heap, HEAP\_NO\_SERIALIZE | HEAP\_ZERO\_MEMORY, 300000 \* sizeof(int));  cout << "\nAddress: " << x1 << "\n\n";  sh(heap);  HeapFree(heap, HEAP\_NO\_SERIALIZE, x1);  sh(heap);  HeapDestroy(heap);  } |
| Скриншот запуска |  |

Задание 6. Разработайте консольное приложение OS08\_06, выполняющее длинный цикл.

Продемонстрируйте с помощью файловой системы /proc структуру адресного пространства.

Продемонстрируйте с помощью pmap структуру адресного пространства.

Определите с помощью утилиты objdump адрес загрузки main-модуля, секций с кодом, данными, неинициализированными глобальными переменными.

|  |  |
| --- | --- |
| Код программы OS08\_06 | #define \_GNU\_SOURCE  #include <stdio.h>  #include <stdlib.h>  #include <unistd.h>  #include <sys/types.h>  #include <sched.h>  /\*  sudo cat /proc/pid/maps  pmap -X pid  objdump -f OS08\_06  size -Ax OS08\_06  .text - секция с кодом (+константы)  .data - инициализированные данные  .bss - неинициализированные данные  \*/  int main()  {  pid\_t pid = getpid();  nice(0);  for (int i = 0; i < 10000000; i++)  {  printf("[OS08\_06] PID = %d\n", pid);  sleep(1);  }  exit(0);  } |
| Скриншот запуска | E:\3course\5sem\OS\Lab8\Images\objdump full.png  E:\3course\5sem\OS\Lab8\Images\objdump text.png  E:\3course\5sem\OS\Lab8\Images\objdupm data.pngE:\3course\5sem\OS\Lab8\Images\size.png |

Задание 7. Разработайте консольное приложение OS08\_07, которое динамически выделяет 256 МБ памяти.

В выделенной памяти разместите int-массив максимальной размерности. Проинициализируйте массив последовательными значениями с шагом 1.

Выведите на консоль адрес выделенной памяти.

После инициализации приложение должно приостановить свое выполнение на длительный интервал времени.

С помощью утилиты pmap определите область памяти в которой выделена память.

|  |  |
| --- | --- |
| Код программы OS08\_07 | #define \_GNU\_SOURCE  #include <stdio.h>  #include <stdlib.h>  #include <unistd.h>  #include <sys/types.h>  #include <sched.h>  int main()  {  pid\_t pid = getpid();  printf("[OS08\_07] PID = %d\n", pid);  sleep(10);  void\* memr = malloc(1024 \* 1024 \* 256);  printf("memr(1024 \* 1024) = %p\n", memr);  int\* arr = (int\*)memr;  for (int i = 0; i < 256; i++)  arr[i] = i;  sleep(20);  exit(0);  } |
| Скриншот запуска |  |