Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Операционные системы

Студент: Коршун Н.И.

ФИТ 3 курс 5 группа

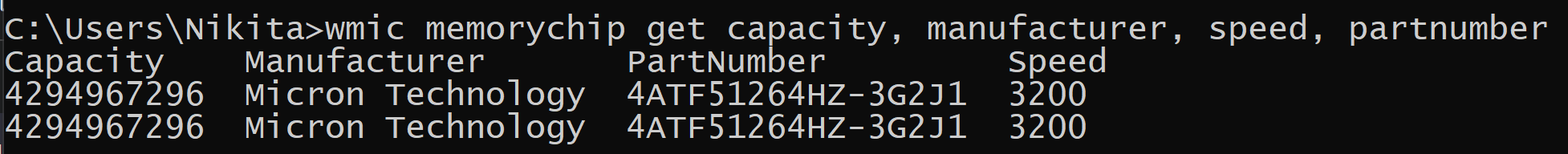
Преподаватель: Савельева М.Г.

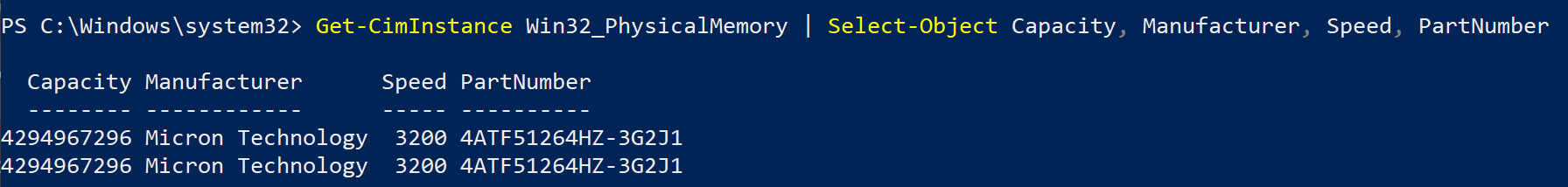
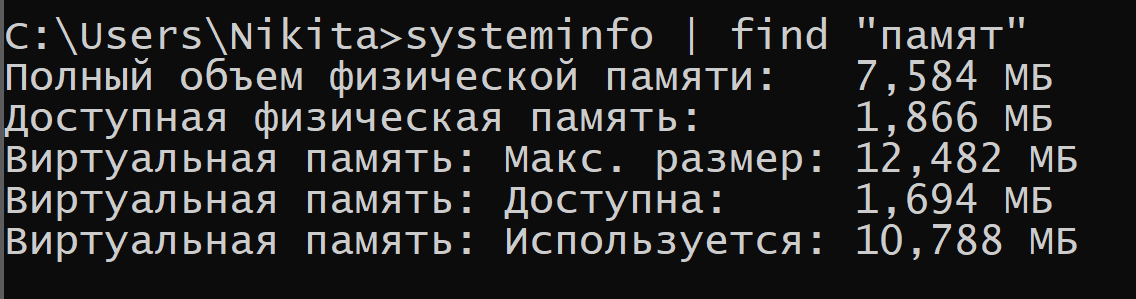
Минск 2023

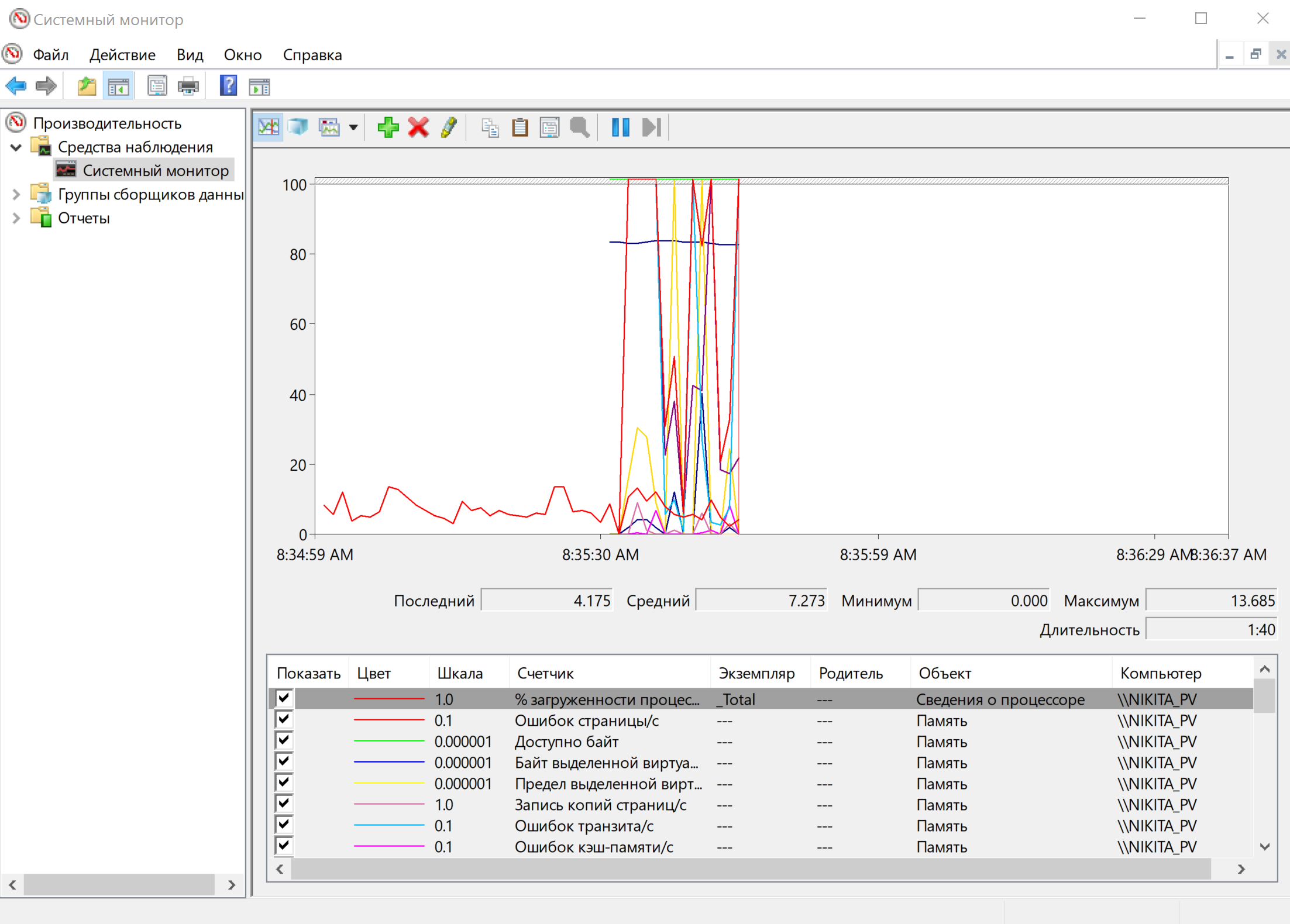
**Лабораторная работа №8**

**Задание 01. Windows**

1. Получите с помощью утилиты **wmic** информации об физической оперативной памяти компьютера, поясните эту информацию.
2. Получите с помощью утилиты **powershell** информации об физической оперативной памяти компьютера, поясните эту информацию.
3. Получите с помощью утилиты **systeminfo** информации об оперативной памяти компьютера, поясните эту информацию.
4. Получите с помощью утилиты **performance monitor** информации об оперативной памяти компьютера, поясните эту информацию.

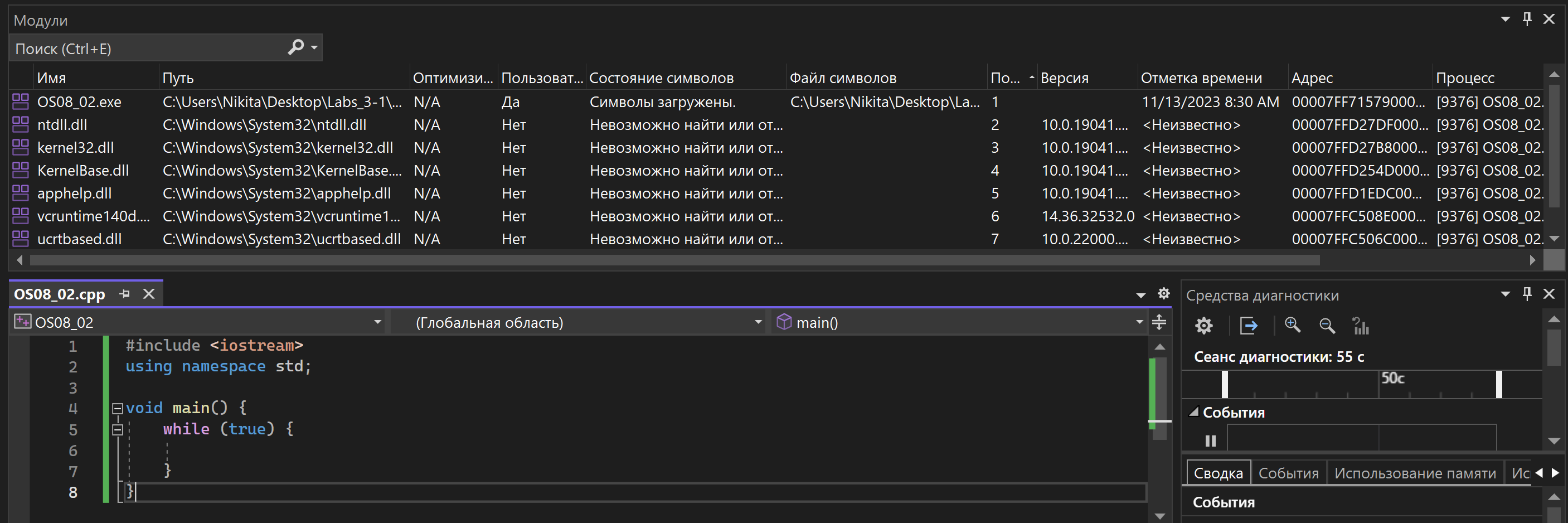


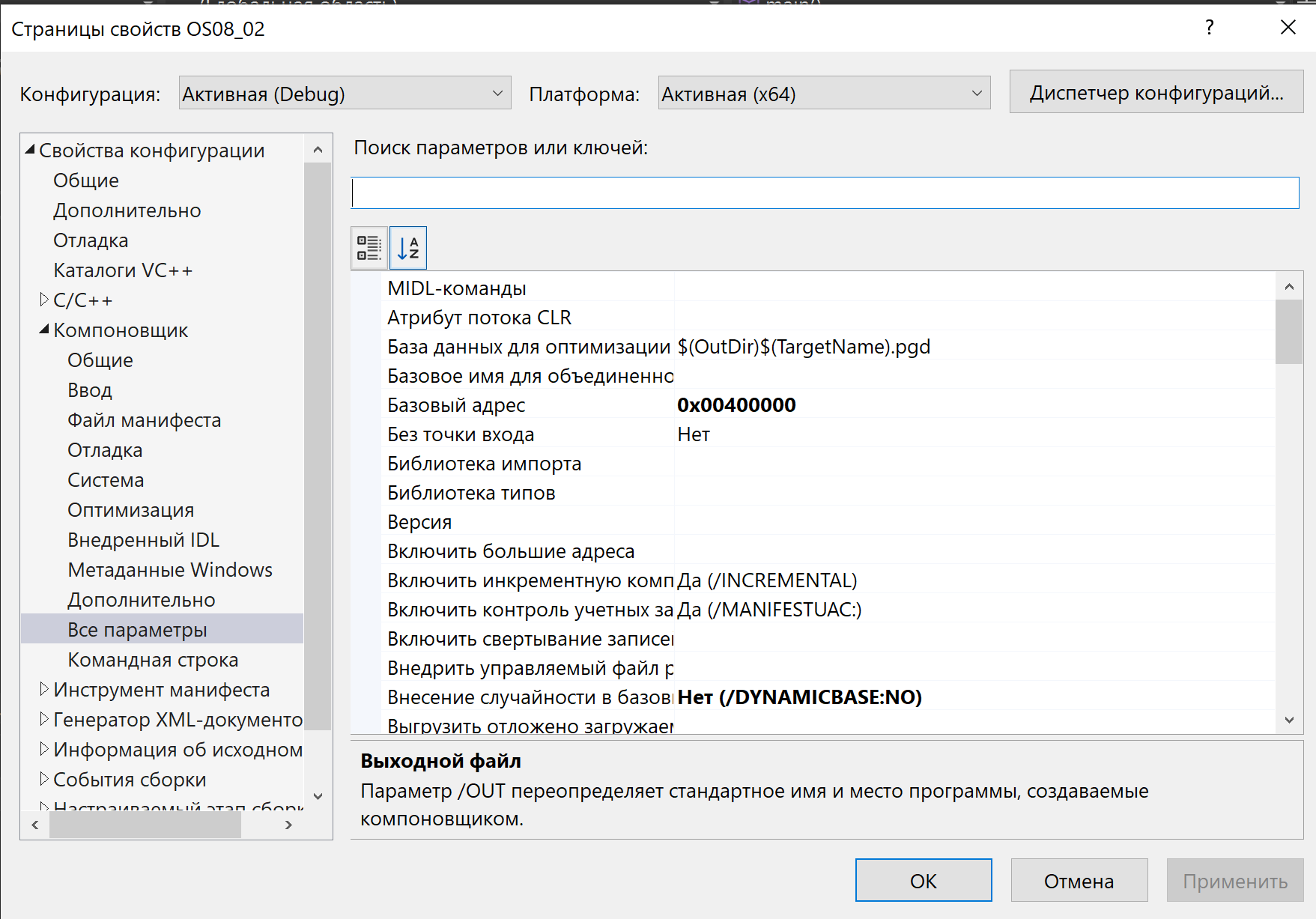
 



**Задание 02.Windows**

1. Разработайте консольное приложение **OS06\_02**, выполняющее длинный цикл.
2. Продемонстрируйте с помощью отладчика адреса расположения модулей приложения **OS06\_02**.
3. Установите для приложения **OS06\_02** стандартный адрес загрузки в память.
4. Продемонстрируйте с помощью отладчика стандартный адрес расположения модулей приложения **OS06\_02**







**Задание 03.Windows**

1. Разработайте консольное приложение **OS06\_03**, выполняющее получение 256 страниц оперативной памяти.
2. Разместите в этой памяти массив типа **int,** полностью занимающее все 256 страниц.
3. Заполните этот массив нарастающей последовательностью чисел с шагом 1.
4. Запишите 3 первых буквы своей фамилии в 16-ричными числами в кодировке Windows-1251.
5. 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 <cstring>  //  //int main() {  // const int pageSize = 4096; // Размер страницы в байтах  // const int numPages = 256; // Количество страниц  //  // // Выделение памяти для 256 страниц  // char\* memory = new char[pageSize \* numPages];  //  // // Размещение массива int, занимающего все 256 страниц  // int\* intArray = reinterpret\_cast<int\*>(memory);  //  // // Заполнение массива нарастающей последовательностью  // for (int i = 0; i < pageSize \* numPages / sizeof(int); ++i) {  // intArray[i] = i + 1;  // }  //  // // Запись 3 первых букв фамилии в 16-ричном представлении  // const char lastName[] = { 0x4B, 0x6F, 0x72 };  // // Копирование букв в память (выберите страницу и смещение внутри страницы)  // memcpy(memory, lastName, sizeof(lastName));  //  // // Отображение успешного завершения  // std::cout << "OS06\_03: Completed successfully." << std::endl;  //  // // Освобождение памяти  // delete[] memory;  //  // return 0;  //}  #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";  }  /\*  K - 75(10) - 0x4B(16)  o - 111(10) - 0x6F(16)  r - 114(10) - 0x72(16)  Страница 4B = 75  75 \* 4096 = 827392(10) = 0x4B000 - добавить для перехода на страницу  Смещение F0E = 1783(10) = 0x000006F7  Искомое значение: начало массива + 0x4B000 + 0x000006F7  \*/  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 + 75 \* 1024 + 1783;  cout << "------ Значение памяти в байте: " << \*byte << " ------\n";  VirtualFree(xmemaddr, NULL, MEM\_RELEASE) ? cout << "\tВиртуальная память освобождена\n" : cout << "\tВиртуальная память не освобождена\n";  saymem();  } |
| *Скриншоты запуска* ***OS08\_03*** |  |

**Задание 04.Windows**

1. Разработайте консольное приложение **OS06\_04,** которое включает функцию **sh**, принимающую 1 параметр: дескриптор (HANDLE) heap.
2. Функция **sh** выводит на консоль, общий размер heap, размеры распределенной и нераспределенных областей памяти heap.
3. Приложение **OS06\_04** размещает в стандартной heap процесса int-массив размерности 300000.
4. Выведите с помощью функции **sh** информацию до размещения массива и после.
5. Объясните результат.

|  |  |
| --- | --- |
| Код программы **OS08\_04** | #include <iostream>  #include <windows.h>  void sh(HANDLE heap) {  PROCESS\_HEAP\_ENTRY entry = { 0 };  SIZE\_T committedSize = 0;  SIZE\_T uncommittedSize = 0;  while (HeapWalk(heap, &entry)) {  if (entry.wFlags & PROCESS\_HEAP\_ENTRY\_BUSY) {  committedSize += entry.cbData;  }  else {  uncommittedSize += entry.cbData;  }  }  std::cout << "Committed size: " << committedSize << std::endl;  std::cout << "Uncommitted size: " << uncommittedSize << std::endl;  }  int main() {  HANDLE heap = GetProcessHeap();  // Вывод информации до размещения массива  std::cout << "Before commit" << std::endl;  sh(heap);  // Размещение массива размерности 300000 в heap  int\* arr = (int\*)HeapAlloc(heap, 0, 300000 \* sizeof(int));  // Вывод информации после размещения массива  std::cout << "\nAfter commit" << std::endl;  sh(heap);  // Освобождение памяти  HeapFree(heap, 0, arr);  return 0;  } |
| Скриншот запуска **OS08\_04** |  |

**Задание 05.Windows**

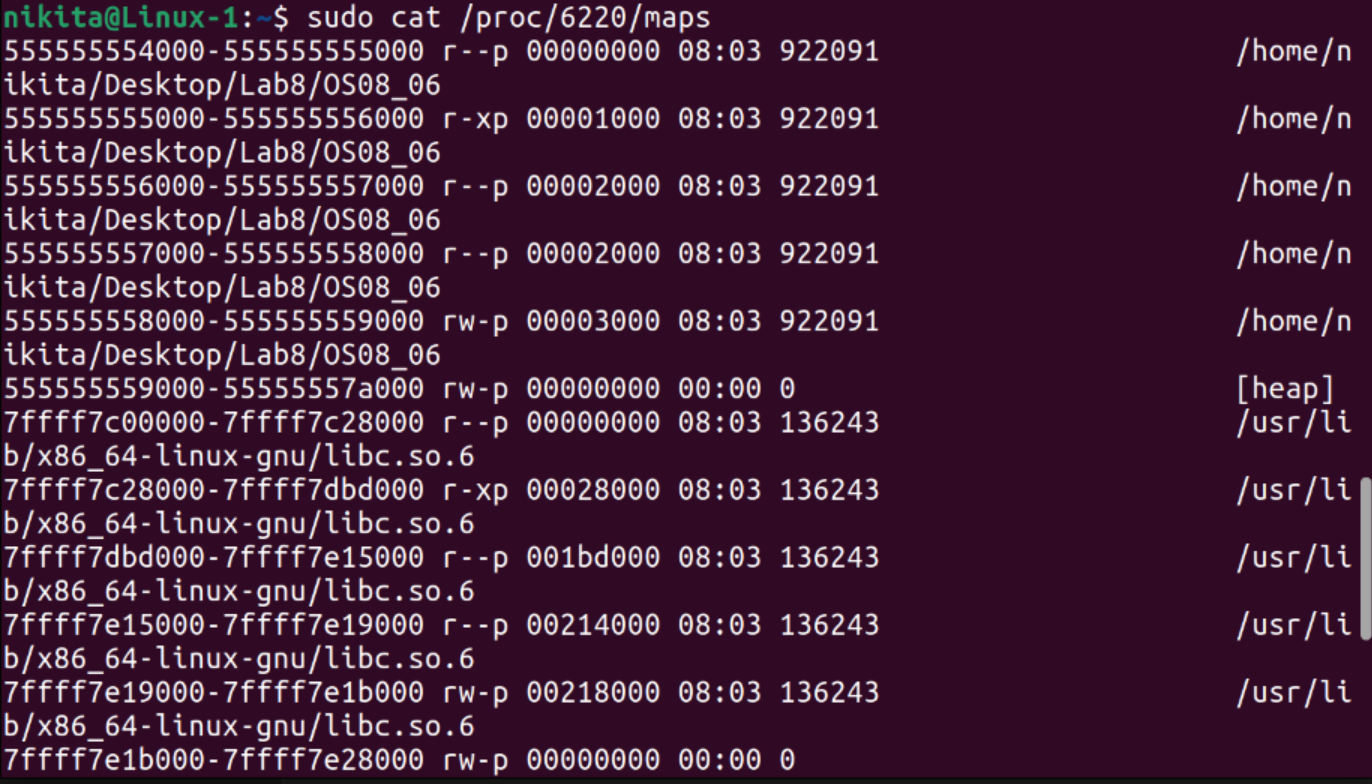
1. Разработайте консольное приложение **OS06\_05** аналогичное приложению **OS06\_05,** но использующее пользовательскую heap, которая имеет первоначальный размер 4MB.
2. Объясните результат.

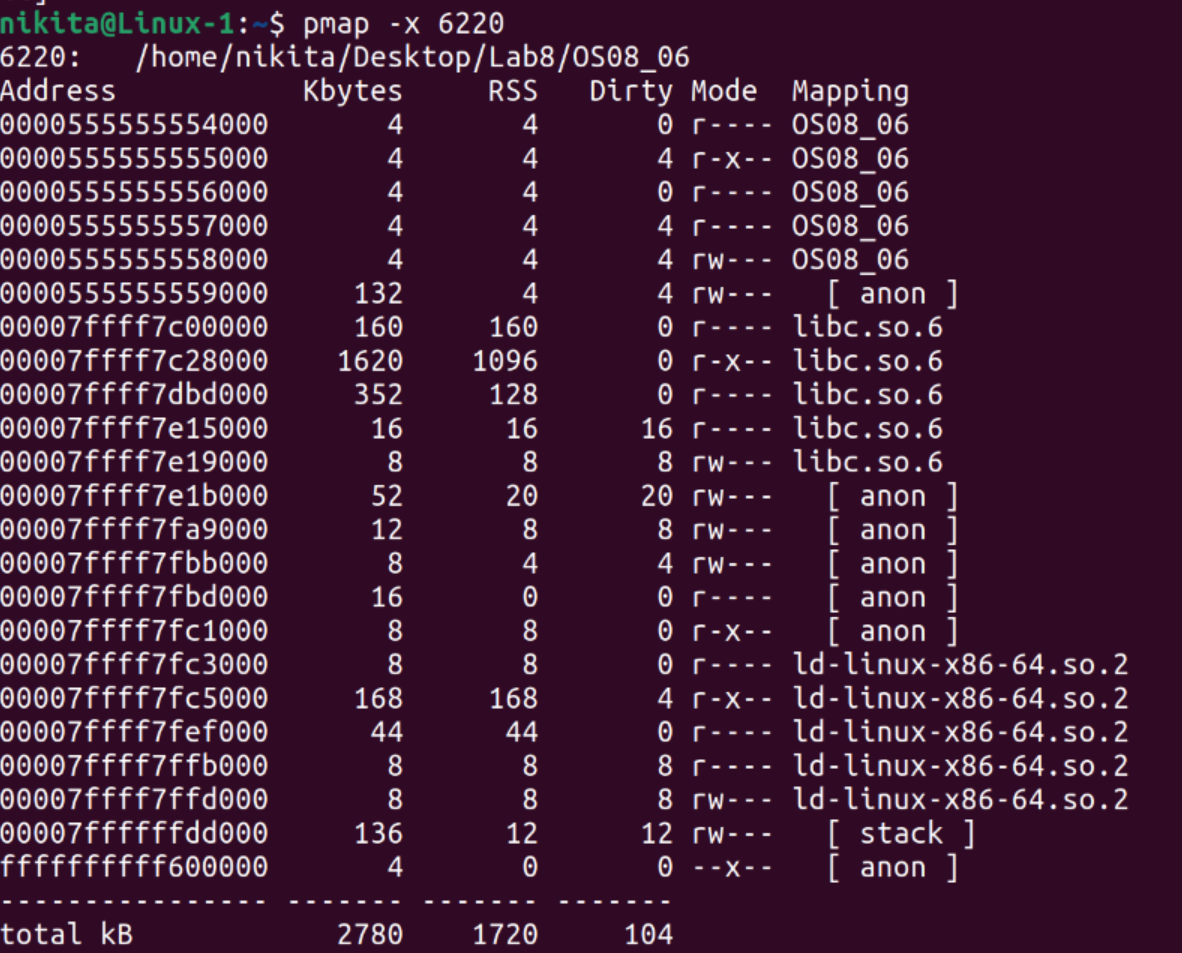
|  |  |
| --- | --- |
| Код программы OS08\_05 | #include <iostream>  #include <windows.h>  void sh(HANDLE heap) {  PROCESS\_HEAP\_ENTRY entry = { 0 };  SIZE\_T committedSize = 0;  SIZE\_T uncommittedSize = 0;  while (HeapWalk(heap, &entry)) {  if (entry.wFlags & PROCESS\_HEAP\_ENTRY\_BUSY) {  committedSize += entry.cbData;  }  else {  uncommittedSize += entry.cbData;  }  }  std::cout << "Committed size: " << committedSize << std::endl;  std::cout << "Uncommitted size: " << uncommittedSize << std::endl;  }  int main() {  HANDLE heap = HeapCreate(HEAP\_NO\_SERIALIZE | HEAP\_ZERO\_MEMORY, 4096 \* 1024, 0);  // Вывод информации до размещения массива  std::cout << "Before commit" << std::endl;  sh(heap);  // Размещение массива размерности 300000 в heap  int\* arr = (int\*)HeapAlloc(heap, 0, 300000 \* sizeof(int));  // Вывод информации после размещения массива  std::cout << "\nAfter commit" << std::endl;  sh(heap);  // Освобождение памяти  HeapFree(heap, 0, arr);  return 0;  } |
| Скриншоты запуска OS08\_05 |  |

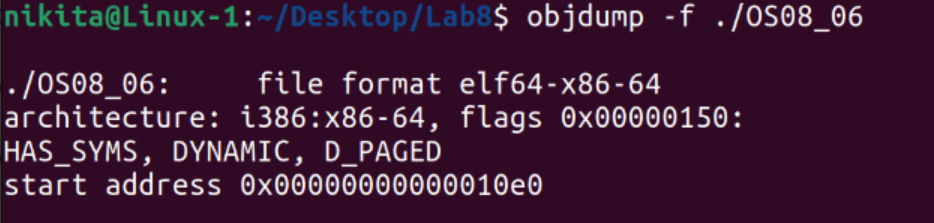
**Задание 06.Linux**

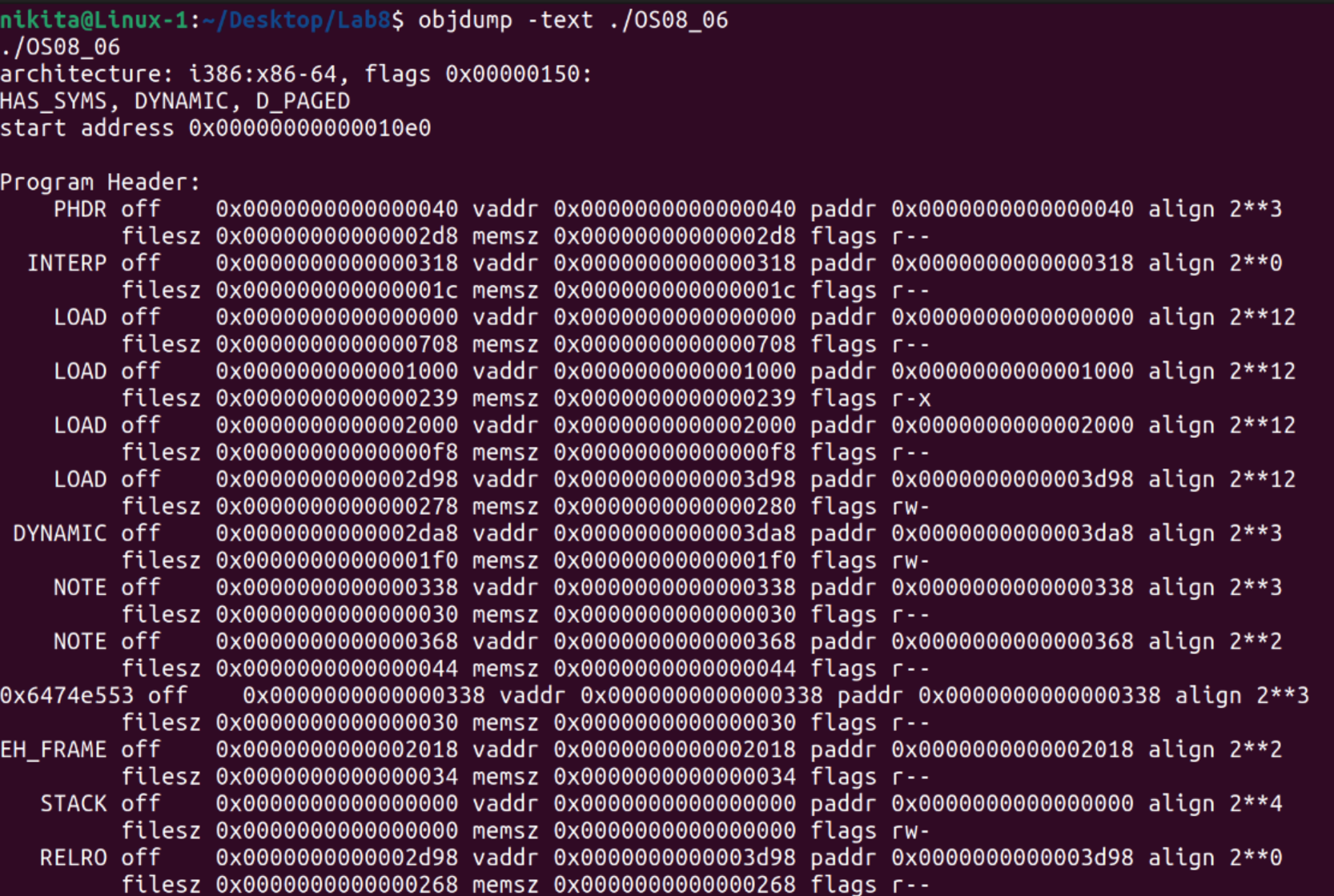
1. Разработайте консольное приложение **OS06\_06**, выполняющее длинный цикл.
2. Продемонстрируйте с помощью файловой системы **/proc** структуру адресного пространства.
3. Продемонстрируйте с помощью **pmap**  структуру адресного пространства.

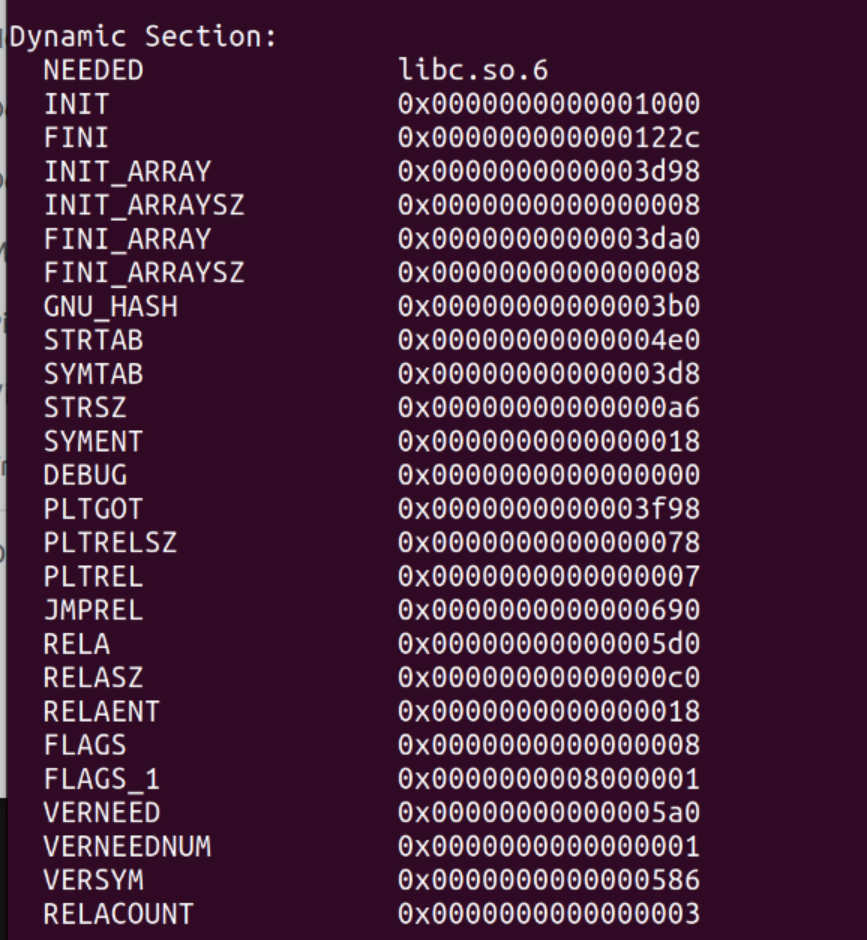
Определите с помощью утилиты objdump адрес загрузки main-модуля, секций с кодом, данными, неинициализированными глобальными переменными.

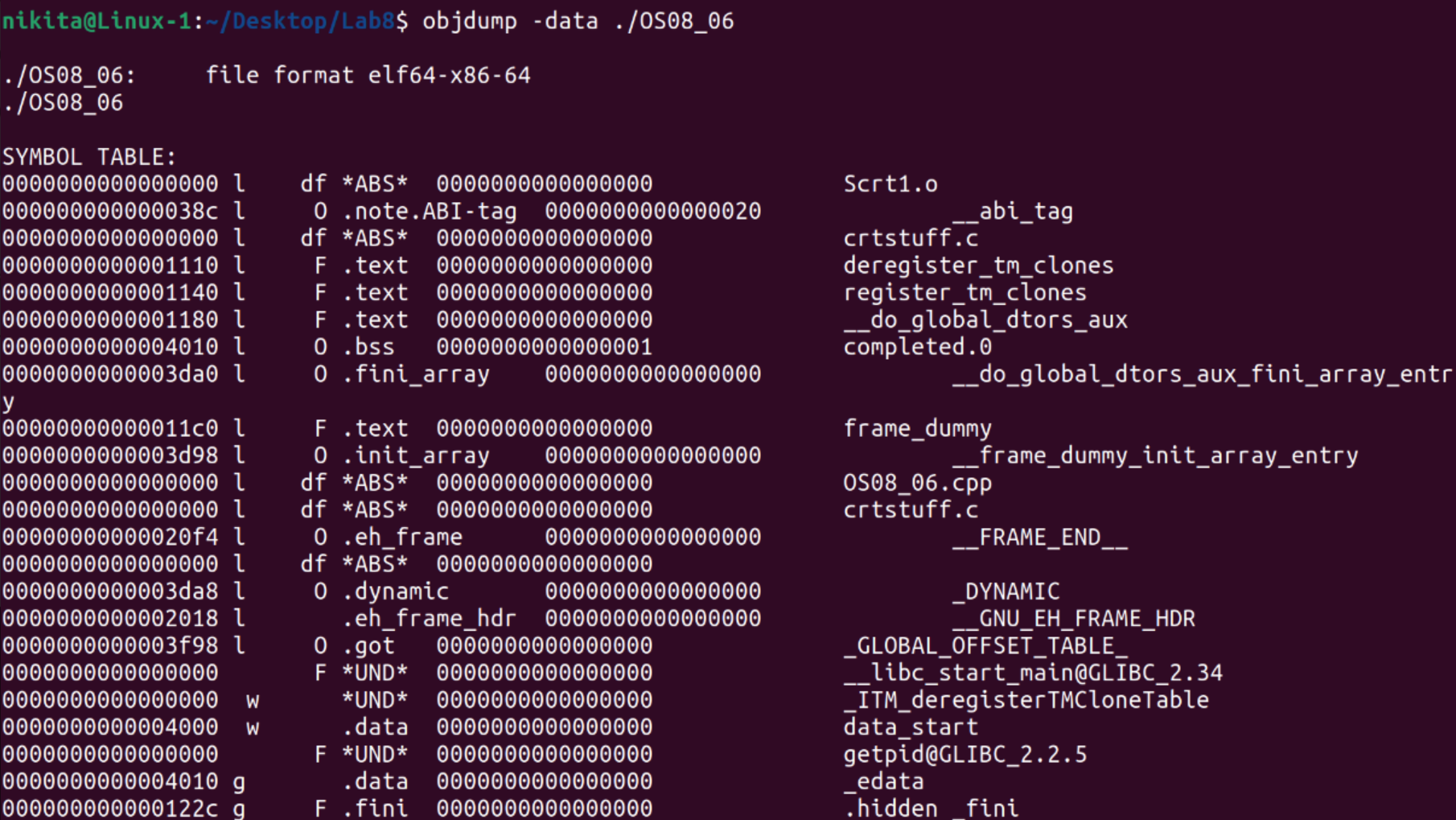


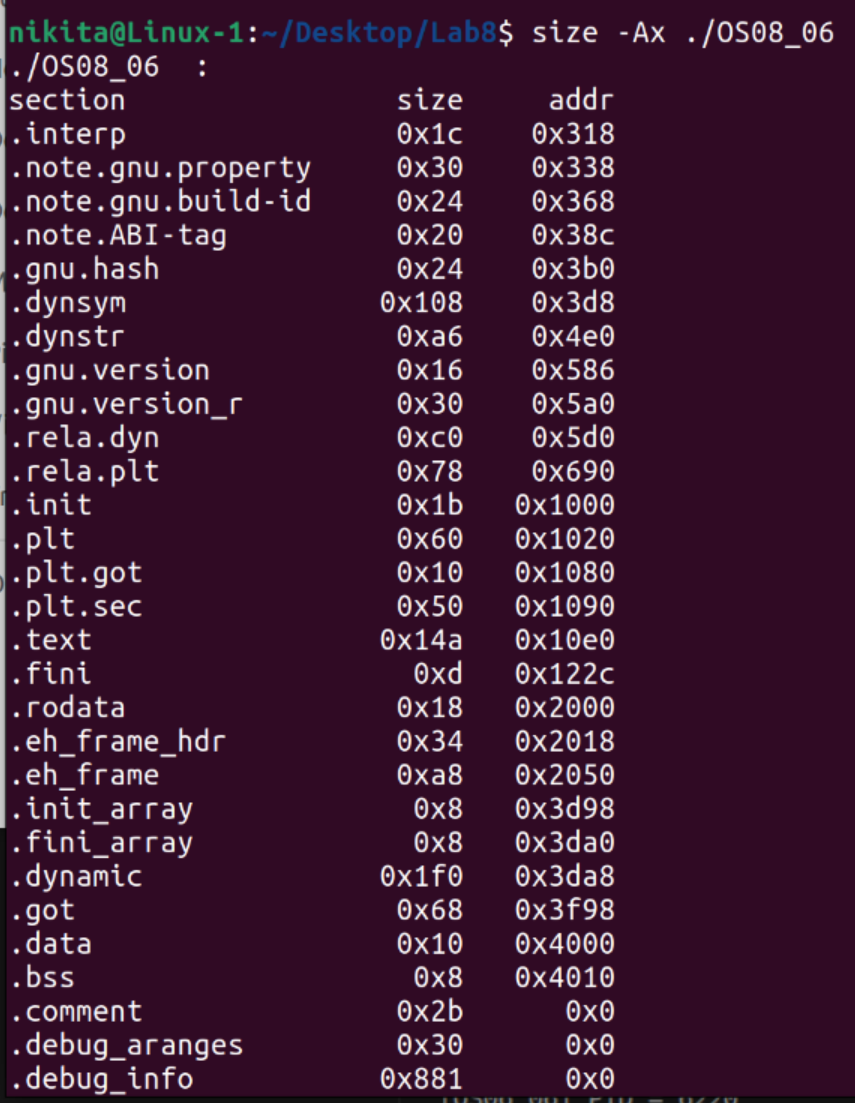












**Задание 07.Linux**

1. Разработайте консольное приложение **OS06\_07**, которое динамически выделяет 256 МБ памяти.
2. В выделенной памяти разместите int-массив максимальной размерности. Проинициализируйте массив последовательными значениями с шагом 1.
3. Выведите на консоль адрес выделенной памяти.
4. После инициализации приложение должно приостановить свое выполнение на длительный интервал времени.
5. 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("[OS06\_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 **OS08\_07** |  |