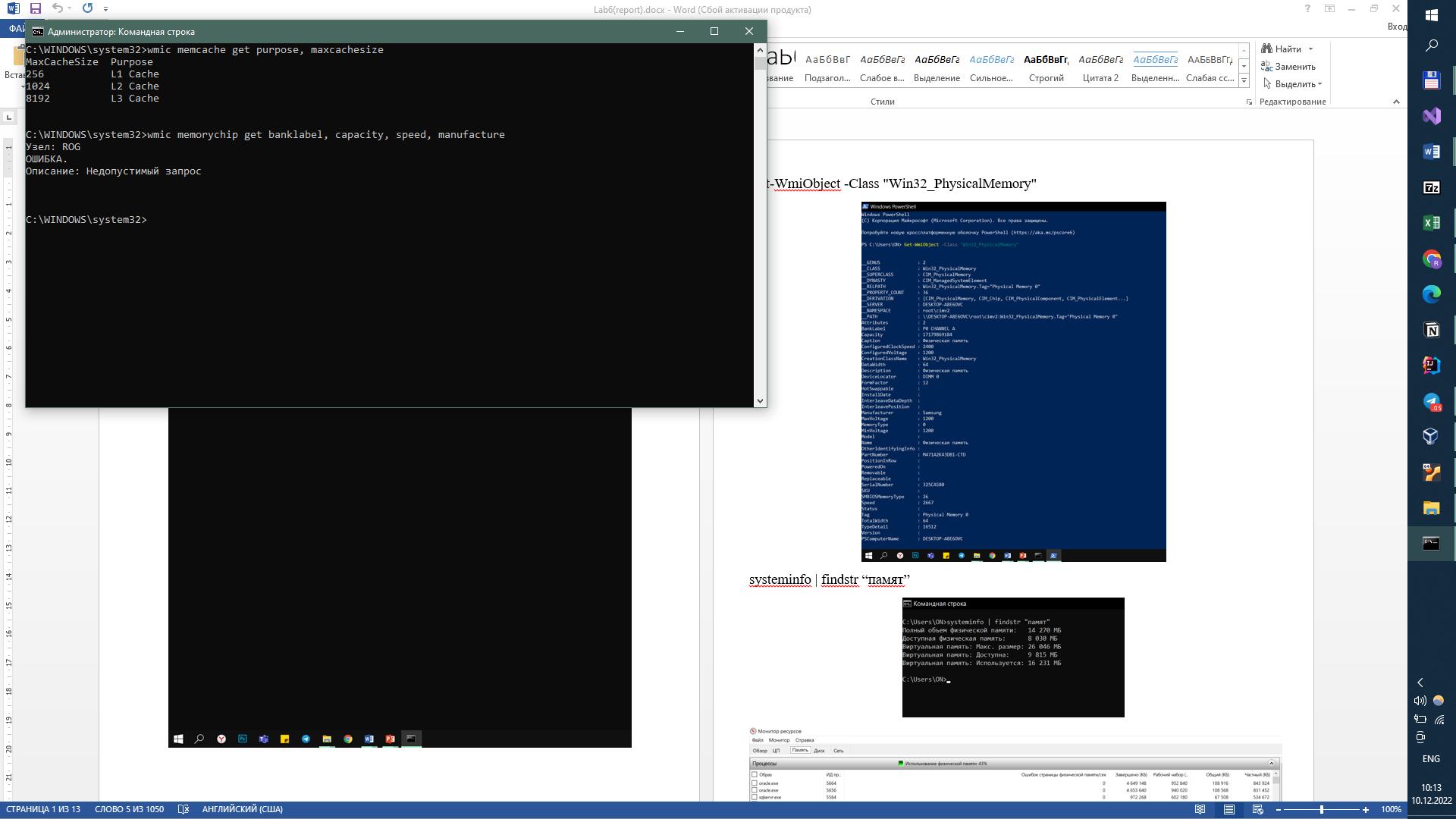
Пузиков Алексей, ФИТ 3/6

**Лабораторная работа №8**

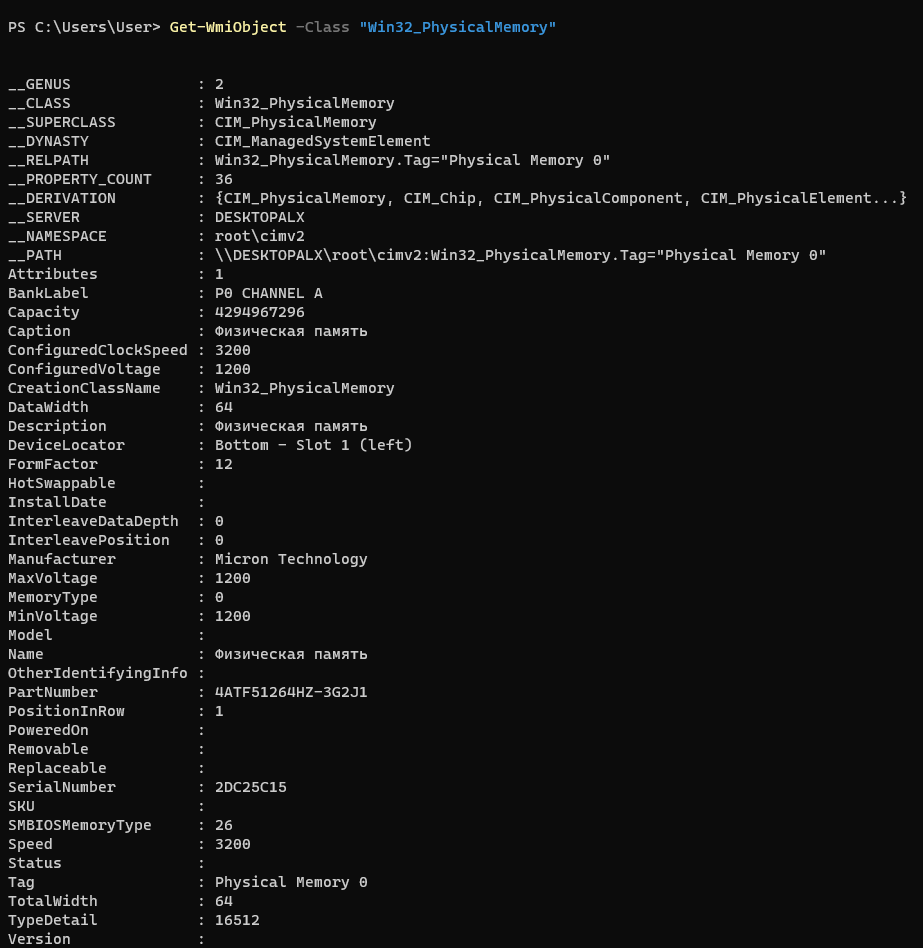
«Управление памятью»

**Задание 01. Windows**

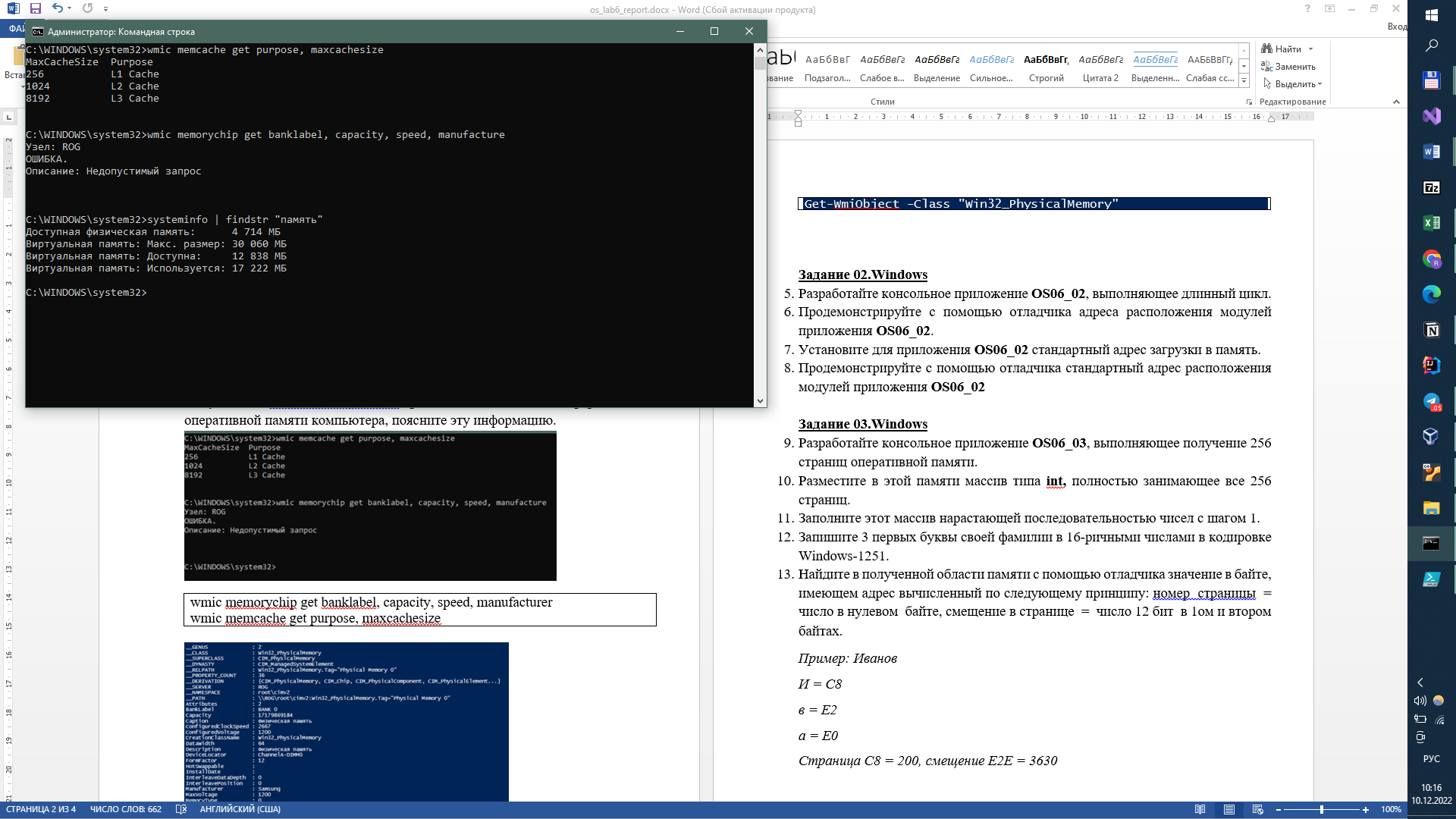
1. Получите с помощью утилиты **wmic** информации об физической оперативной памяти компьютера, поясните эту информацию.
2. Получите с помощью утилиты **powershell** информации об физической оперативной памяти компьютера, поясните эту информацию.
3. Получите с помощью утилиты **systeminfo** информации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



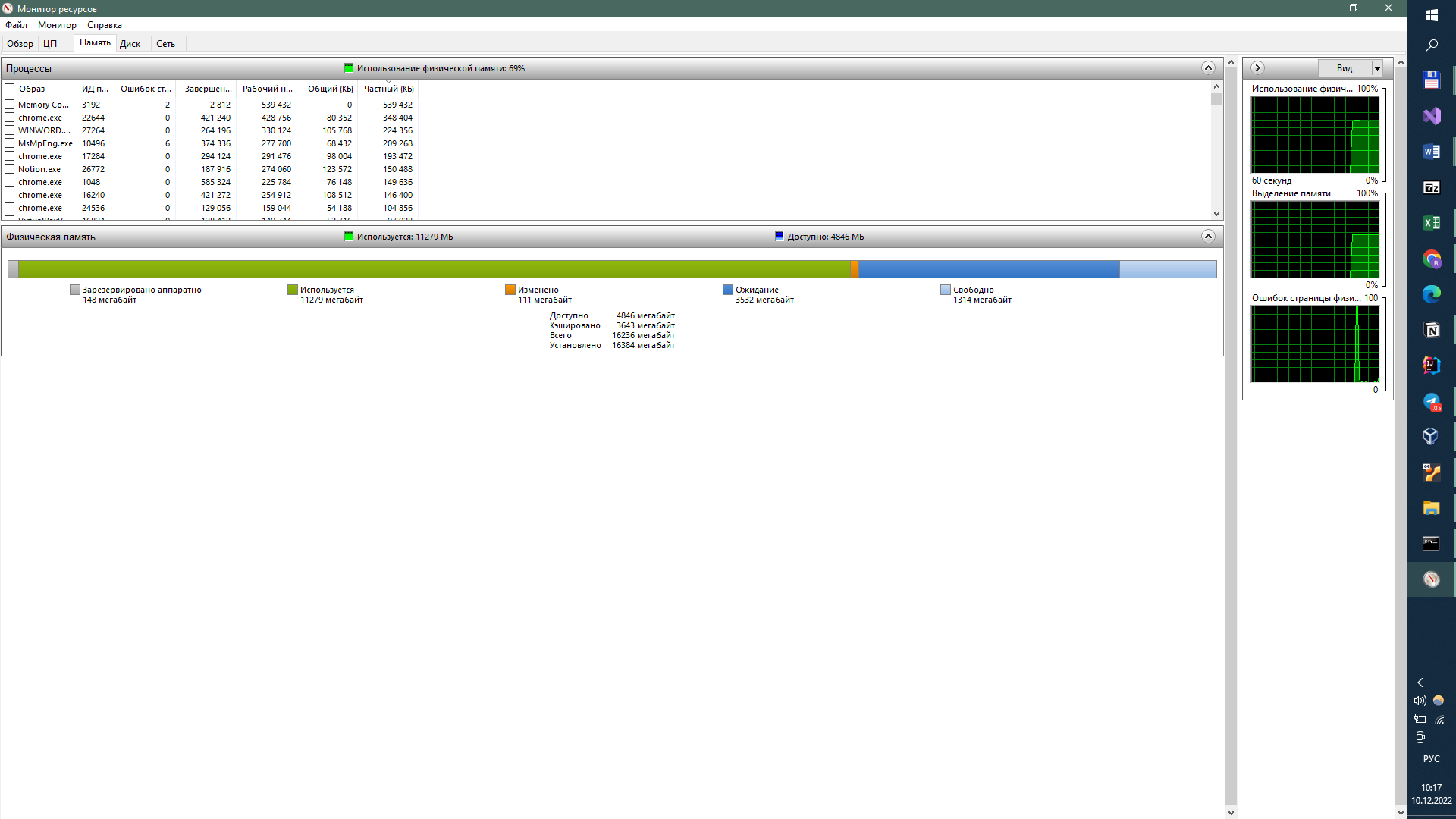
|  |
| --- |
| wmic memorychip get banklabel, capacity, speed, manufacturer  wmic memcache get purpose, maxcachesize |



|  |
| --- |
| Get-WmiObject -Class "Win32\_PhysicalMemory" |

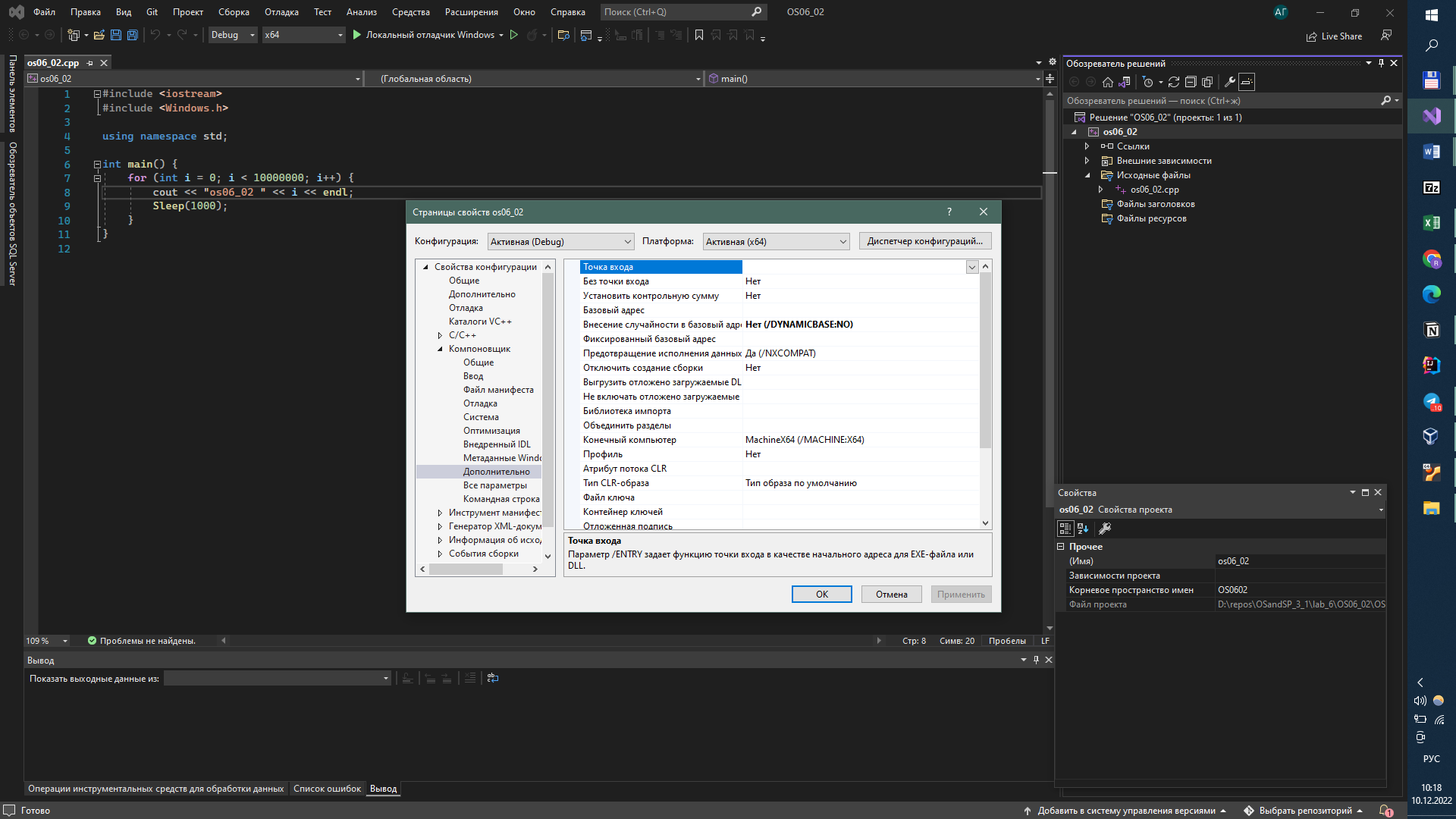


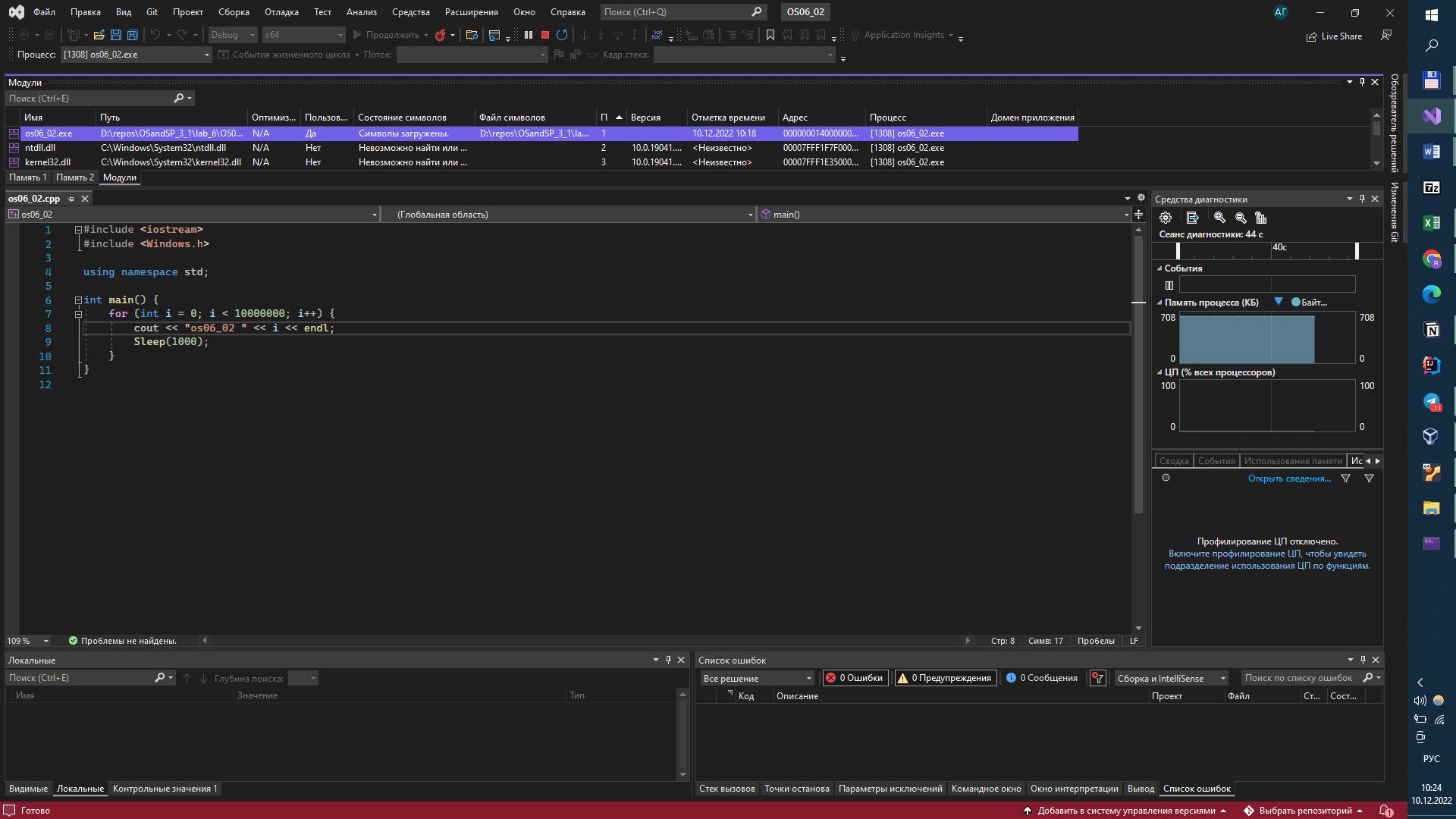
|  |
| --- |
| systeminfo | findstr “память” |



**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*

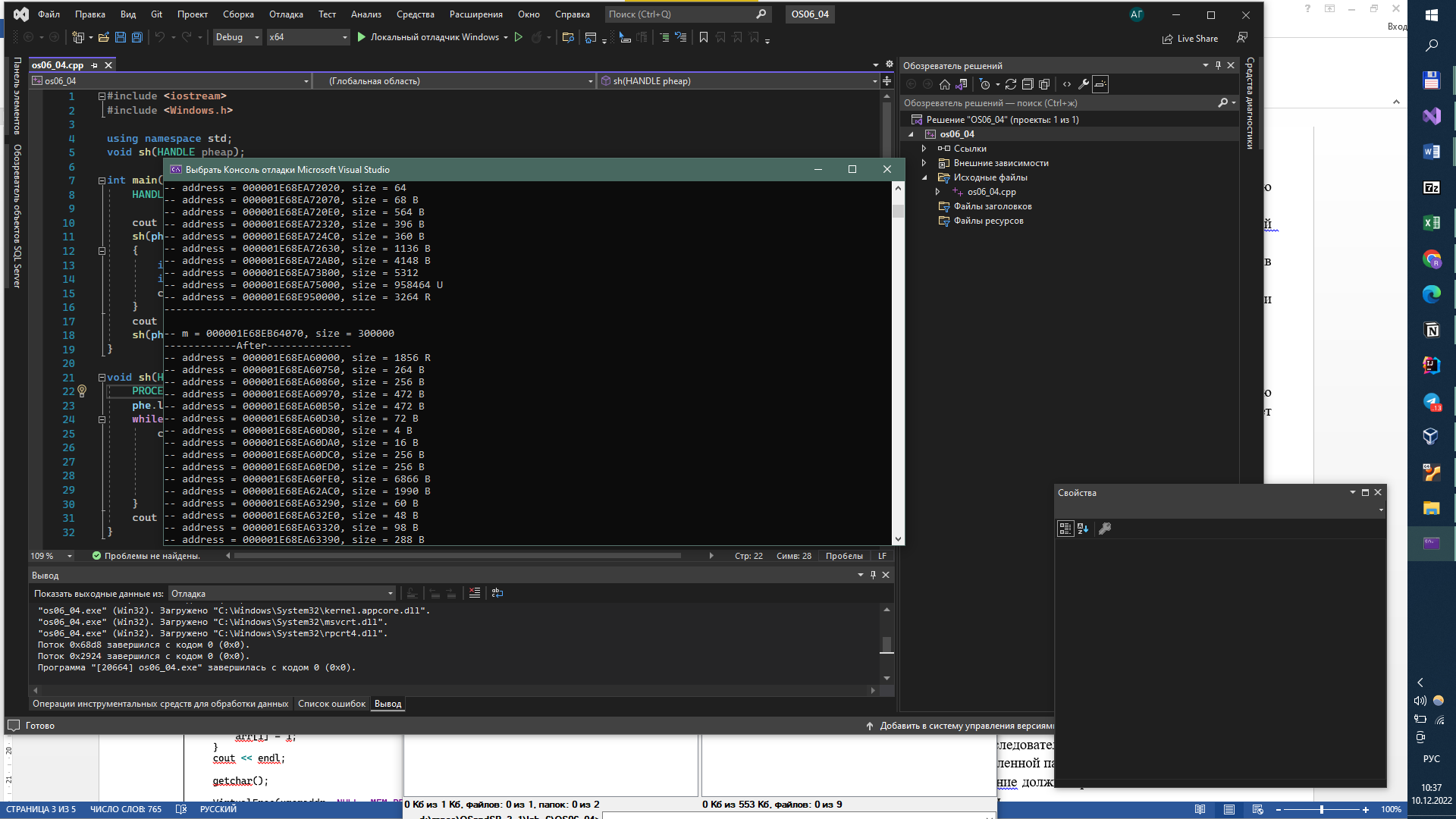
*а = E0*

*Страница C8 = 200, смещение E2E = 3630*

|  |
| --- |
| #include <iostream>  #include <Windows.h>  #define KB (1024)  #define PG (4 \* KB)  using namespace std;  //К CA  //о EE  //в E2  //страница CA = 202  //смещение EEE = 3822  //адрес = 202 \* PG + 3822 = 0x000CAEEE  int main() {  int pages = 256;  int countItems = pages \* PG / sizeof(int);  LPVOID xmemaddr = VirtualAlloc(NULL, pages \* PG, MEM\_COMMIT, PAGE\_READWRITE);  int\* arr = (int\*)xmemaddr;  for (int i = 0; i < countItems; i++) {  arr[i] = i;  }  cout << endl;  getchar();  VirtualFree(xmemaddr, NULL, MEM\_RELEASE) ?  cout << "VirtualFree true\n" : cout << "VirtualFree false\n";  system("pause");  } |

**Задание 04.Windows**

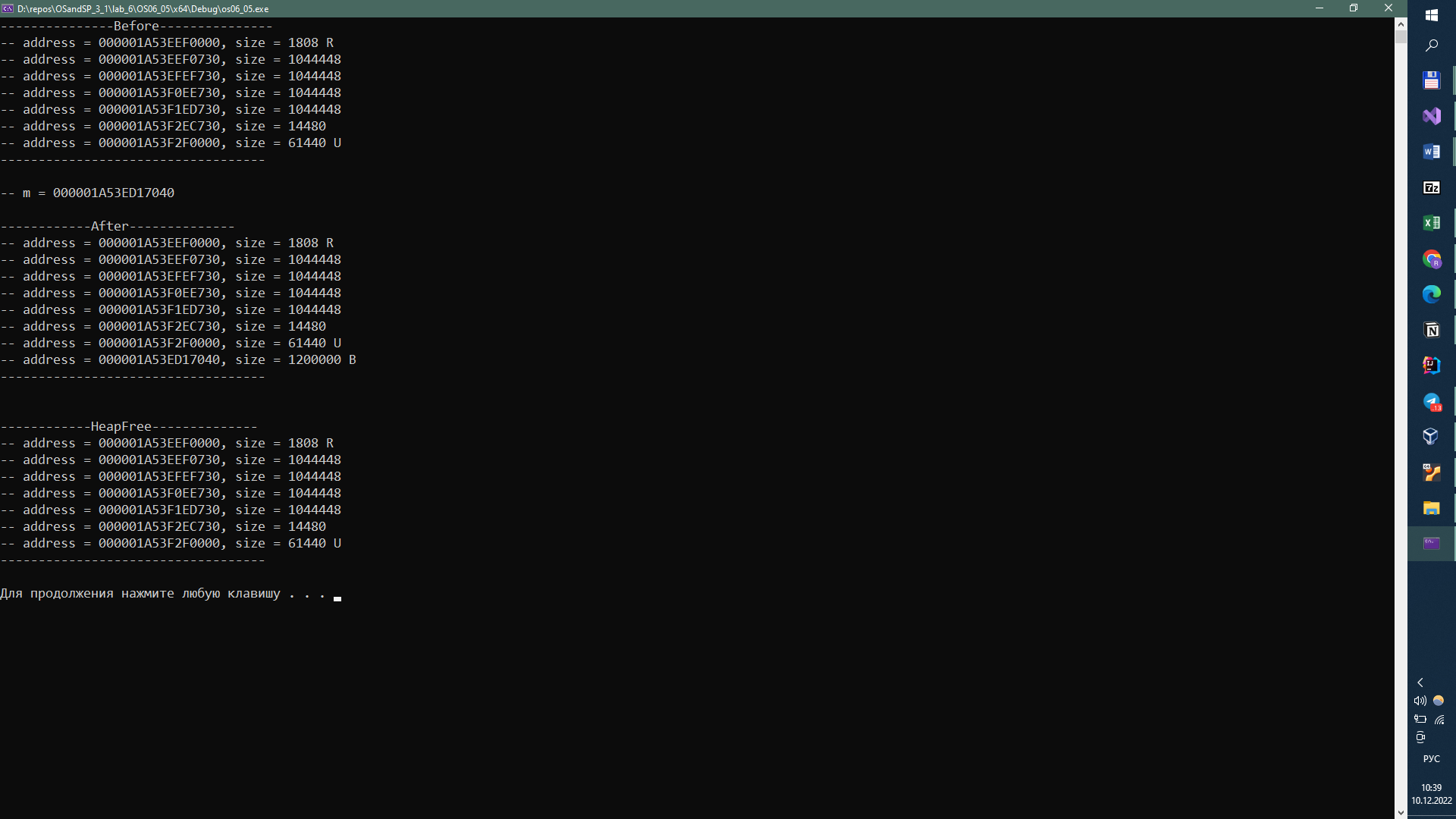
1. Разработайте консольное приложение **OS06\_04,** которое включает функцию **sh**, принимающую 1 параметр: дескриптор (HANDLE) heap.
2. Функция **sh** выводит на консоль, общий размер heap, размеры распределенной и нераспределенных областей памяти heap.
3. Приложение **OS06\_04** размещает в стандартной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



**Задание 05.Windows**

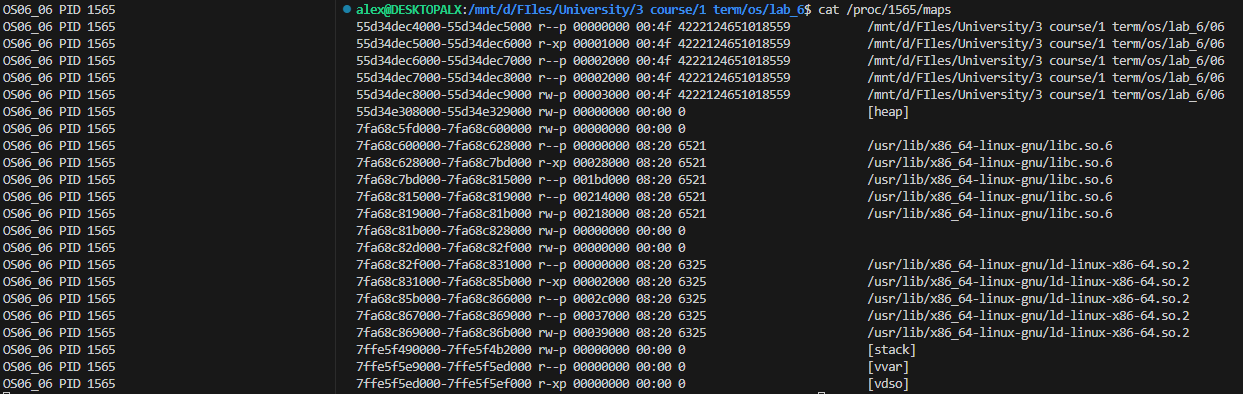
1. Разработайте консольное приложение **OS06\_05** аналогичное приложению **OS06\_05,** но использующее пользовательскую heap, которая имеет первоначальный размер 4MB.
2. Объясните результат.

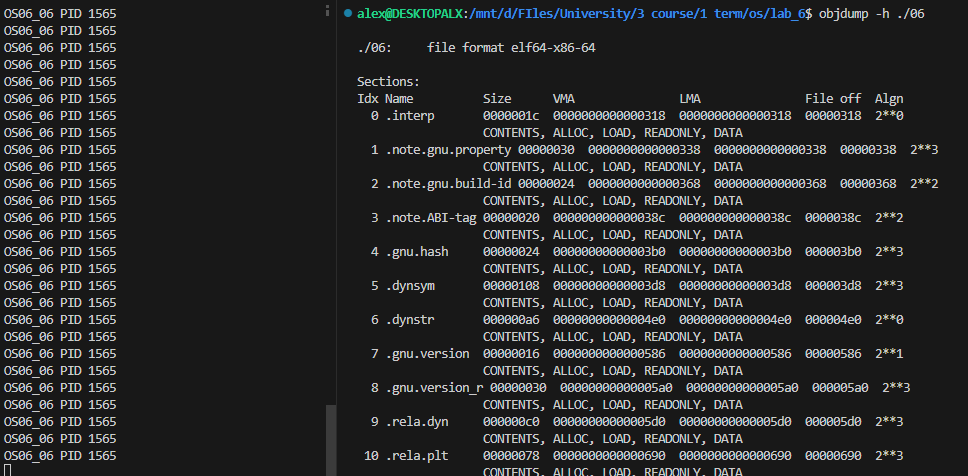
|  |
| --- |
| #include <iostream>  #include <Windows.h>  #define KB (1024)  using namespace std;  void sh(HANDLE pheap);  int main() {  HANDLE heap = HeapCreate(HEAP\_NO\_SERIALIZE | HEAP\_ZERO\_MEMORY, 4096 \* KB, 0);  cout << "---------------Before---------------\n";  sh(heap);  int\* m = (int\*)HeapAlloc(heap, HEAP\_NO\_SERIALIZE | HEAP\_ZERO\_MEMORY, 300000 \* sizeof(int));  cout << "-- m = " << hex << m << "\n";  cout << "\n------------After--------------\n";  sh(heap);  HeapFree(heap, HEAP\_NO\_SERIALIZE, m);  cout << "\n------------HeapFree--------------\n";  sh(heap);  HeapDestroy(heap);  system("pause");  }  void sh(HANDLE pheap) {  PROCESS\_HEAP\_ENTRY phe;  phe.lpData = NULL;  while (HeapWalk(pheap, &phe)) {  cout << "-- address = " << hex << phe.lpData  << ", size = " << dec << phe.cbData  << ((phe.wFlags & PROCESS\_HEAP\_REGION) ? " R" : "") // начало непрерывной области  << ((phe.wFlags & PROCESS\_HEAP\_UNCOMMITTED\_RANGE) ? " U" : "") // нераспределенная область  << ((phe.wFlags & PROCESS\_HEAP\_ENTRY\_BUSY) ? " B" : "") // распределенная область  << "\n";  }  cout << "-----------------------------------\n\n";  } |

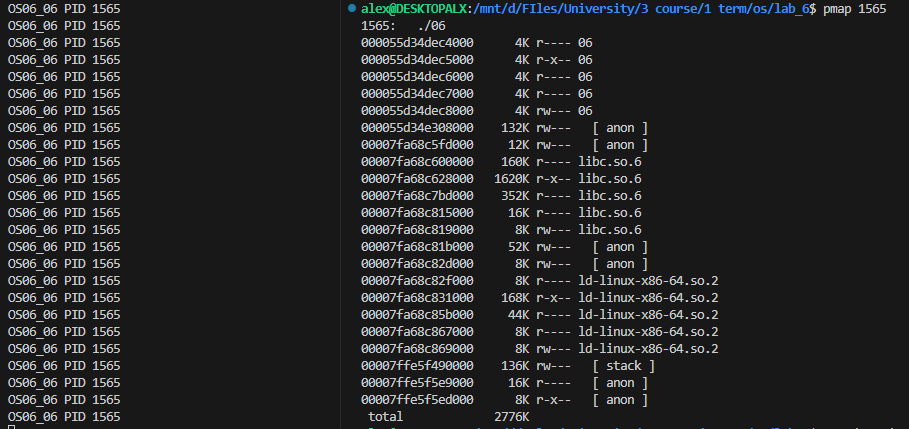


**Задание 06.Linux**

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.
4. 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.

**Задание 08** Ответьте на следующие вопросы

1. Поясните понятие «виртуальная память».
2. Поясните понятие «свопинг».
3. Поясните понятие «страничная память».
4. Поясните понятие MMU.
5. Поясните понятие TLB.
6. Какая информация содержится в строке таблицы страниц
7. Поясните принцип применения хэш-таблиц.
8. Поясните применение «инвертированной таблицы физических» страничной памяти.
9. Поясните понятие «рабочий набор страниц».
10. Поясните принцип работы алгоритма LRU.
11. Windows: поясните назначение сервиса SysMain.
12. Windows: поясните назначение файла hiberfil.sys.
13. Windows: поясните назначение файла pagefile.sys.
14. Windows: поясните назначение файла swapfile.sys.
15. Windows: перечислите области адресного пространства (от младших к старшим адресам) и поясните их назначения.
16. Windows: какой стандартный начальный размер области heap?
17. Windows: каким образом можно изменить начальный размер области памяти heap приложения?
18. Windows: какой стандартный размер области памяти stack?
19. Windows: каким образом можно изменить размер области памяти stack приложения?
20. Windows: поясните назначение функции Windows API: GlobalMemoryStatus.
21. Windows: поясните назначение функции Windows API: VirtualQuery; перечислите значения атрибутов Protect, State и Type.
22. Windows: что такое «рабочее множество»? поясните принцип управления рабочим множеством с помощью OS API.
23. Windows: что означает «страница заблокирована»? с помощью каких функций OS API можно установить блокировку страниц и снять блокировку? Какое максимальное количество страниц можно заблокировать?
24. Windows: что такое «heap»? Что такое «heap процесса»? Что такое «пользовательская heap»? Поясните принцип устройства heap.
25. Linux: перечислите области адресного пространства (от младших к старшим адресам) и поясните их назначения.

Linux: в какой части адресного пространства выделяется памяти с помощью функций malloc, calloc?