Учреждение образования

«Белорусский государственный технологический университет»

**Лабораторная работа №8**

Выполнил:

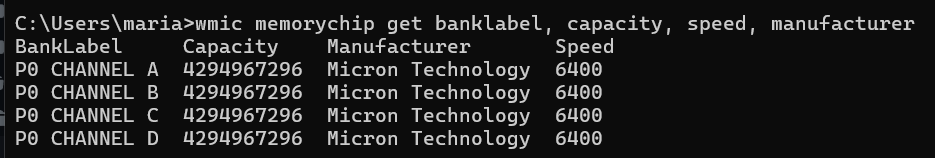
Студент 3 курса 4 группы ФИТ

Сосновец Мария Игоревна

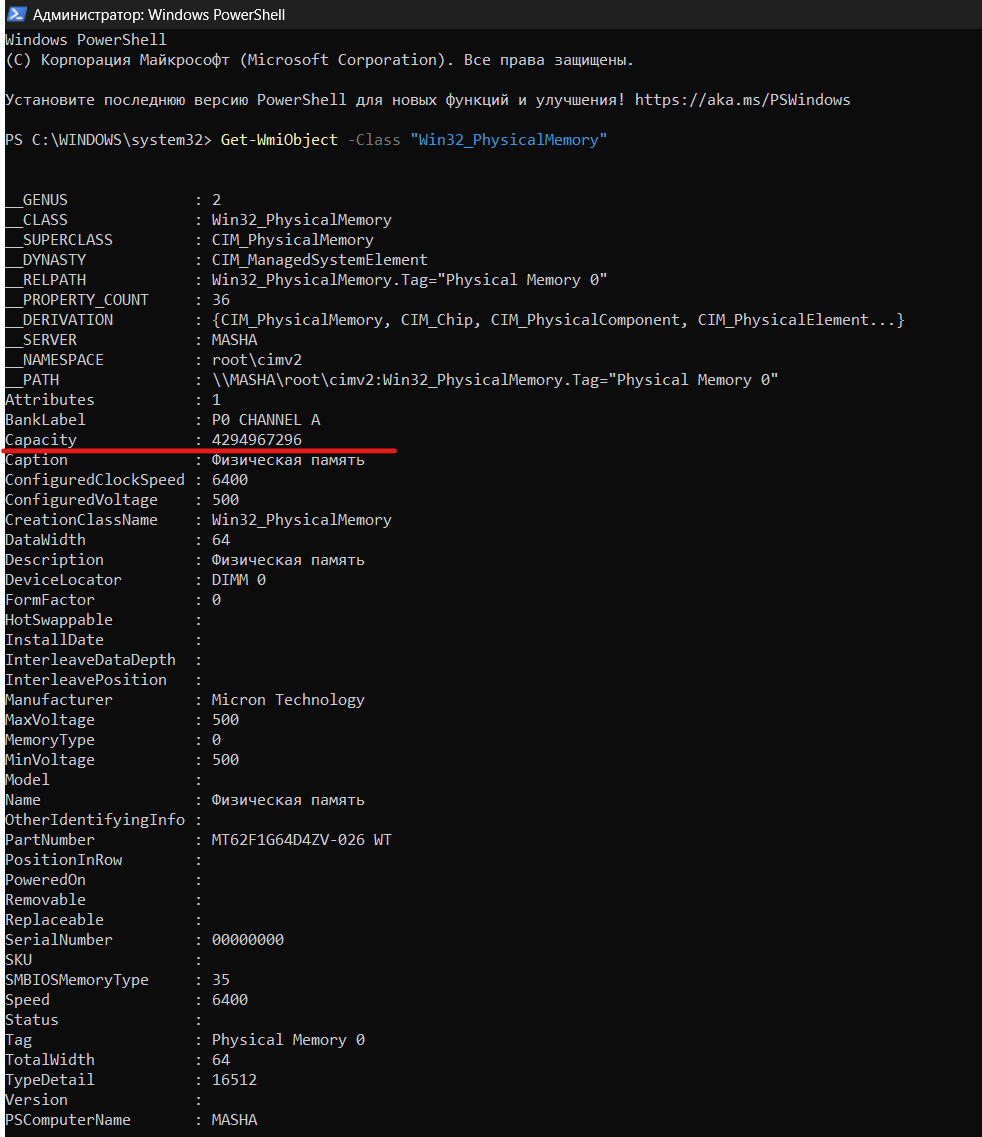
2024 г.

**Задание 01. Windows**

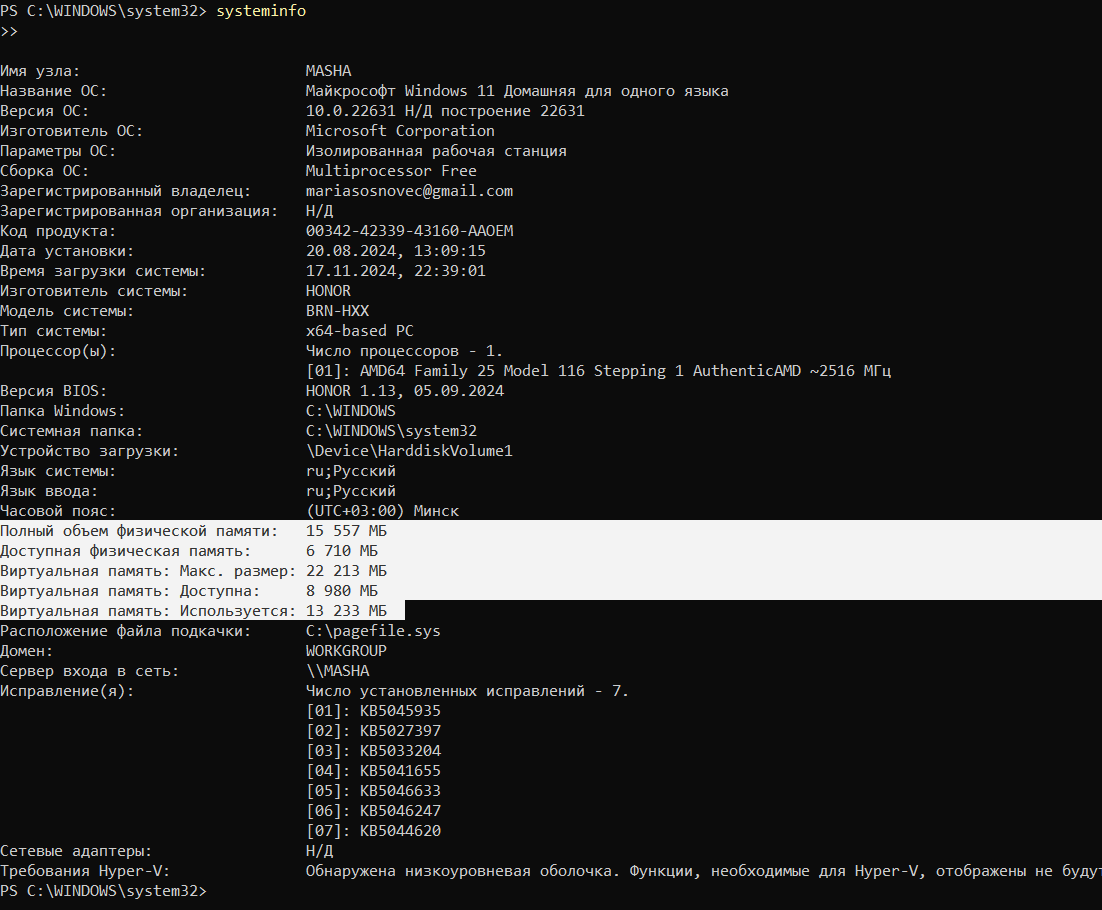
1. Получите с помощью утилиты wmic информации об физической оперативной памяти компьютера, поясните эту информацию.



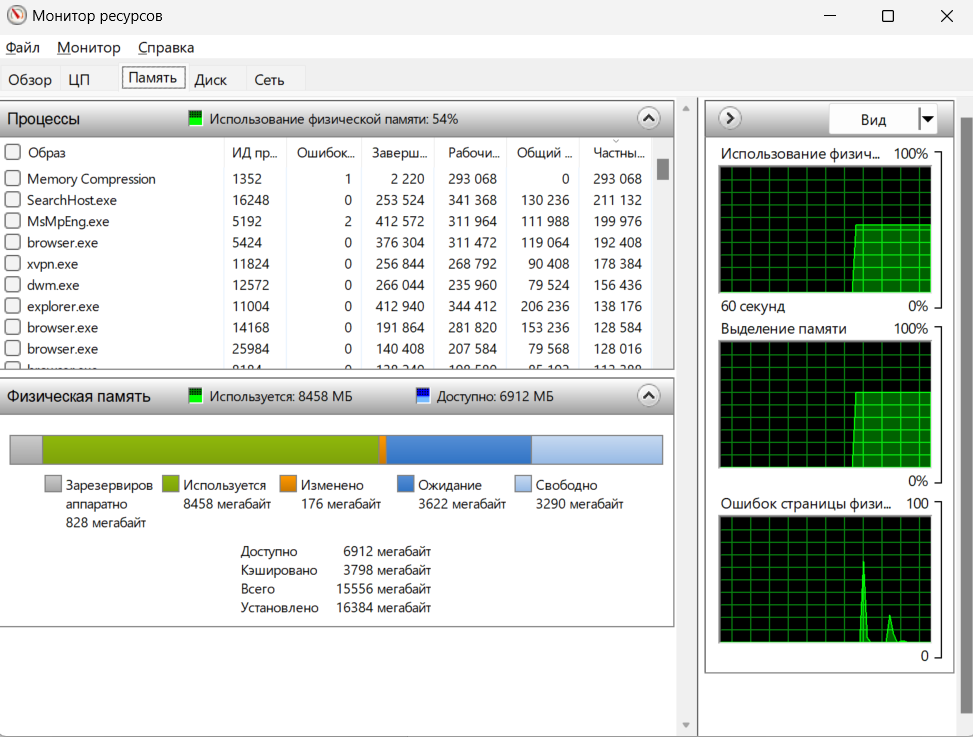
1. Получите с помощью утилиты powershell информации об физической оперативной памяти компьютера, поясните эту информацию.



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



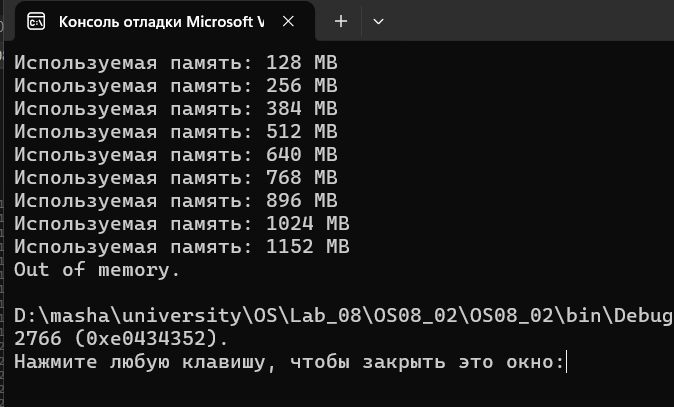
1. Получите с помощью утилиты performance monitor информации об оперативной памяти компьютера, поясните эту информацию.

****

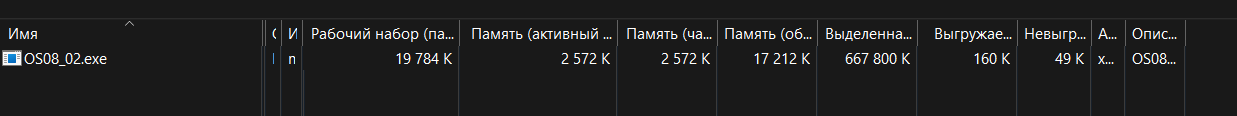
**Задание 02.Windows**

1. Разработайте на языке программирования C# консольное приложение OS08\_02, которое каждые 5 секунд создает новый объект размером 128 МБ и выводит объем используемой памяти, установите Platform Target x86.

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Threading;  class Program  {  static void Main(string[] args)  {  List<Big> bigObjects = new List<Big>();  while (true)  {  bigObjects.Add(new Big());  long memoryUsed = GC.GetTotalMemory(false) / (1024 \* 1024);  Console.WriteLine($"Используемая память: {memoryUsed} MB");  Thread.Sleep(5000);  }  }  }  class Big  {  public byte[] Data;  public Big()  {  Data = new byte[128 \* 1024 \* 1024];  }  } |

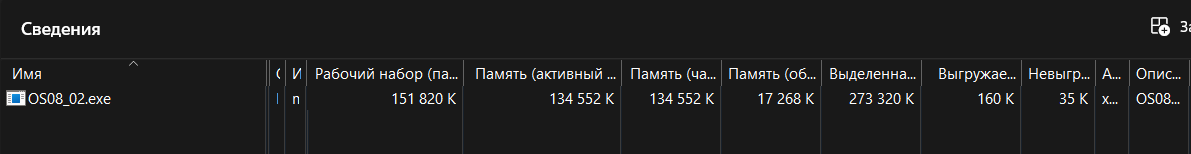


1. Понаблюдайте за выделенной памятью рабочим набором памяти приложения OS08\_02 в Диспетчере задач. Будет ли приложение работать бесконечно?

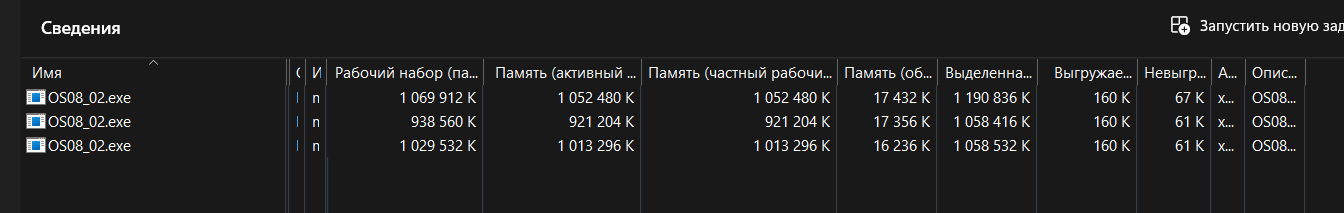


1. Добавьте в приложение OS08\_02 код, который заполняет произвольными значениями выделенную память, лучше всего в отдельном потоке. Изменилось ли соотношение выделенной памяти и рабочего набора?

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Threading;  class Program  {  private static readonly object lockObject = new object();  static void Main(string[] args)  {  List<Big> bigObjects = new List<Big>();  Thread fillerThread = new Thread(() =>  {  Random random = new Random();  while (true)  {  lock (lockObject)  {  foreach (var big in bigObjects)  {  FillMemory(big, random);  }  }  Thread.Sleep(1000);  }  });  fillerThread.IsBackground = true;  fillerThread.Start();  while (true)  {  lock (lockObject)  {  bigObjects.Add(new Big());  }  long memoryUsed = GC.GetTotalMemory(false) / (1024 \* 1024);  Console.WriteLine($"Используемая память: {memoryUsed} MB");  Thread.Sleep(5000);  }  }  static void FillMemory(Big big, Random random)  {  for (int i = 0; i < big.Data.Length; i++)  {  big.Data[i] = (byte)random.Next(0, 256);  }  }  }  class Big  {  public byte[] Data;  public Big()  {  Data = new byte[128 \* 1024 \* 1024];  }  } |



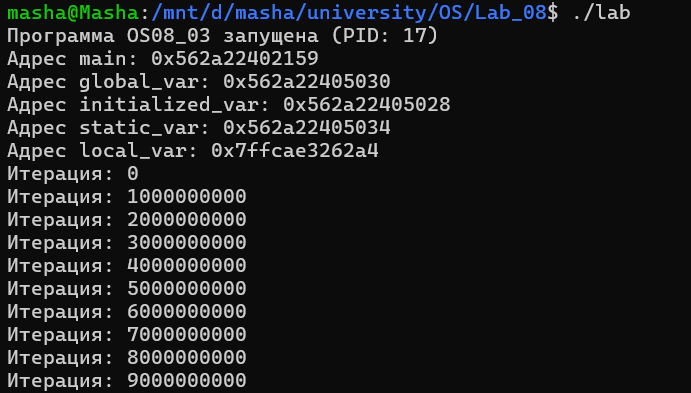
1. 8. Запустите несколько экземпляров приложения OS08\_02.Сравните частный и общий рабочие наборы.



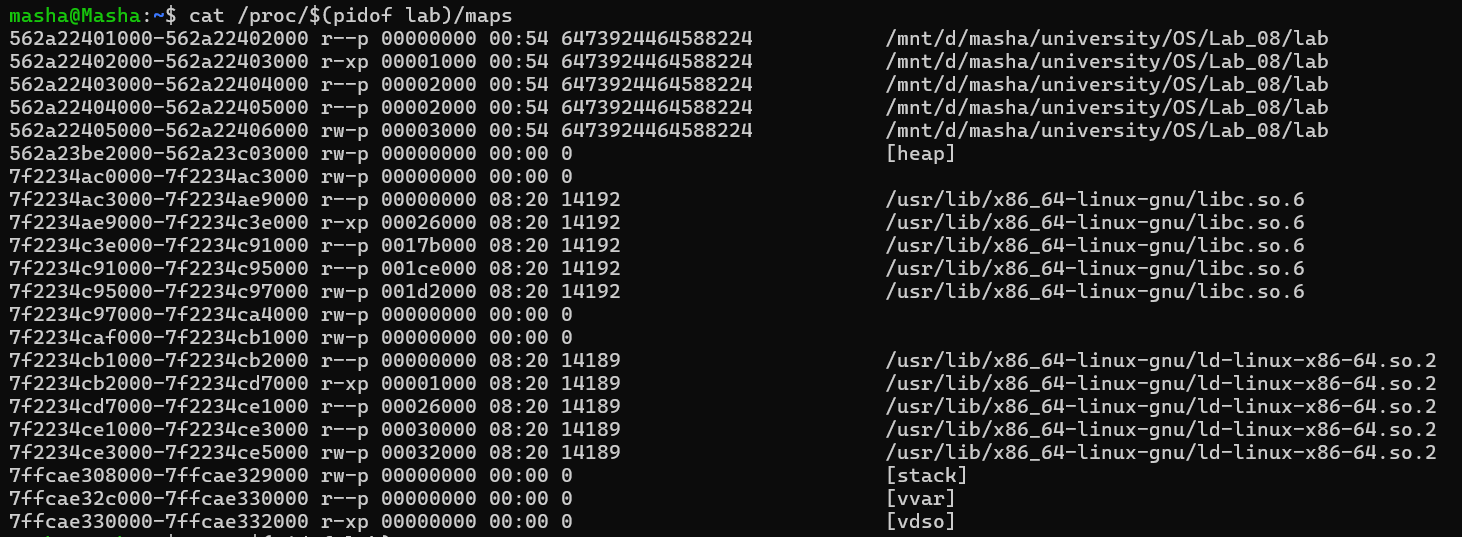
**Задание 03.Linux**

1. Разработайте консольное приложение OS08\_03, выполняющее длинный цикл.

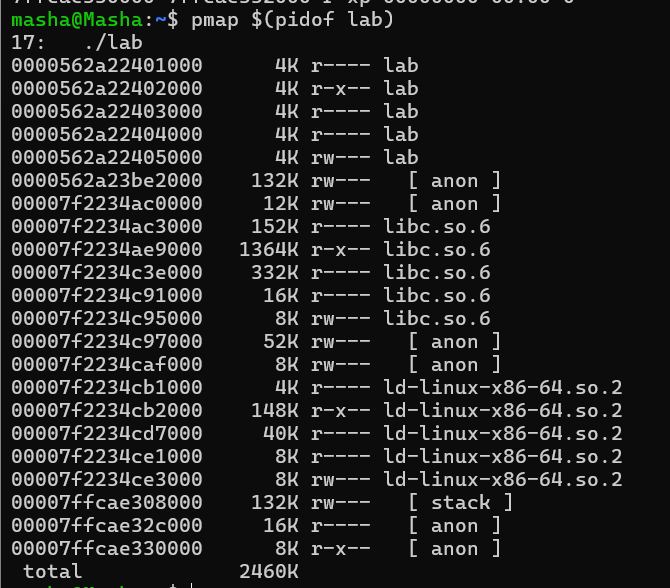
|  |
| --- |
| #include <stdio.h>  #include <unistd.h>  int global\_var = 0;  int initialized\_var = 42;  int main() {  static int static\_var = 0;  int local\_var = 0;  printf("Программа OS08\_03 запущена (PID: %d)\n", getpid());  printf("Адрес main: %p\n", main);  printf("Адрес global\_var: %p\n", &global\_var);  printf("Адрес initialized\_var: %p\n", &initialized\_var);  printf("Адрес static\_var: %p\n", &static\_var);  printf("Адрес local\_var: %p\n", &local\_var);  for (long long i = 0; i < 1e10; ++i) {  if (i % 1000000000 == 0) {  printf("Итерация: %lld\n", i);  sleep(1);  }  }  return 0;  } |



1. Продемонстрируйте с помощью файловой системы /proc структуру адресного пространства.



1. Продемонстрируйте с помощью pmap структуру адресного пространства.



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

