МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждения образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

Направление специальности 1-40 01 01 10 Программное обеспечение информационных технологий (программирование интернет-приложений)

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ:

по дисциплине «Операционные системы»

Исполнитель

студент (ка) 3 курса группы 6 Розель Станислав Александрович

(Ф.И.О.)

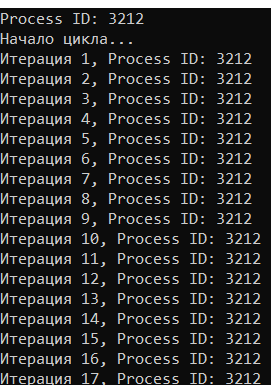
Минск 2024

Задание 1

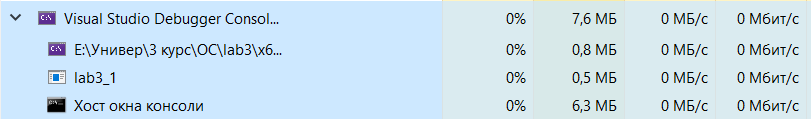
Листинг OS03\_01

|  |
| --- |
| #include <iostream>  #include <windows.h>  #include <process.h>  int main()  {  setlocale(LC\_ALL, "Russian");  DWORD pid = GetCurrentProcessId();  std::cout << "Process ID: " << pid << std::endl;  std::cout << "Начало цикла..." << std::endl;  for (int i = 0; i < 100; i++)  {  Sleep(100);  std::cout << "Итерация " << i + 1 << ", Process ID: " << pid << std::endl;  }  std::cout << "Цикл завершен" << std::endl;  system("pause");  return 0;  } |

Вывод консоли



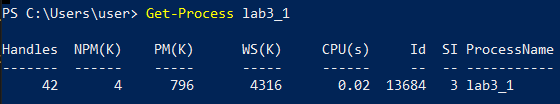
Task manager



Tasklist



PowerShell ISE



Performance monitor



Задание 2

Листинг OS03\_02

#define \_CRT\_SECURE\_NO\_WARNINGS

#define WIN32\_LEAN\_AND\_MEAN

#include <Windows.h>

#include <processthreadsapi.h>

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

int main()

{

setlocale(LC\_ALL, "");

STARTUPINFO si1, si2;

PROCESS\_INFORMATION pi1, pi2;

ZeroMemory(&si1, sizeof(si1));

si1.cb = sizeof(si1);

ZeroMemory(&si2, sizeof(si2));

si2.cb = sizeof(si2);

ZeroMemory(&pi1, sizeof(pi1));

ZeroMemory(&pi2, sizeof(pi2));

if (!CreateProcess(

L"C:\\Users\\Павел Пиво\\Desktop\\лабы\\OC\\лаба 3\\lab\_3\\x64\\Debug\\OS03\_02\_1.exe",

NULL,

NULL,

NULL,

FALSE,

0,

NULL,

NULL,

&si1,

&pi1

)) {

printf("Ошибка при создании первого процесса: %d\n", GetLastError());

return 1;

}

printf("Первый дочерний процесс создан с ID: %lu\n", pi1.dwProcessId);

if (!CreateProcess(

L"C:\\Users\\Павел Пиво\\Desktop\\лабы\\OC\\лаба 3\\lab\_3\\x64\\Debug\\OS03\_02\_2.exe",

NULL,

NULL,

NULL,

FALSE,

0,

NULL,

NULL,

&si2,

&pi2

)) {

printf("Ошибка при создании второго процесса: %d\n", GetLastError());

CloseHandle(pi1.hProcess);

CloseHandle(pi1.hThread);

return 1;

}

printf("Второй дочерний процесс создан с ID: %lu\n", pi2.dwProcessId);

WaitForSingleObject(pi1.hProcess, INFINITE);

WaitForSingleObject(pi2.hProcess, INFINITE);

CloseHandle(pi1.hProcess);

CloseHandle(pi1.hThread);

CloseHandle(pi2.hProcess);

CloseHandle(pi2.hThread);

return 0;

}

Листинг OS03\_02\_1

#define WIN32\_LEAN\_AND\_MEAN

#include <Windows.h>

#include <processthreadsapi.h>

#include <stdio.h>

#include <stdlib.h>

int main()

{

for (size\_t i = 0; i < 50; i++)

{

printf("Process1 ID (PID) of the new process: %lu\n", GetCurrentProcessId);

Sleep(1000);

}

}

Листинг OS03\_02\_1

#define WIN32\_LEAN\_AND\_MEAN

#include <Windows.h>

#include <processthreadsapi.h>

#include <stdio.h>

#include <stdlib.h>

int main()

{

for (size\_t i = 0; i < 125; i++)

{

printf("Process2 ID (PID) of the new process: %lu\n", GetCurrentProcessId);

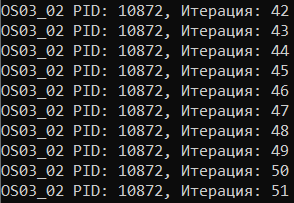
Sleep(1000);

}

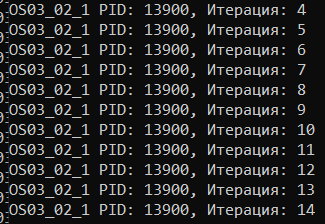
}

Вывод консоли

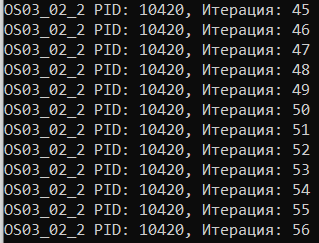
OS03\_02.exe



OS03\_02\_01.exe



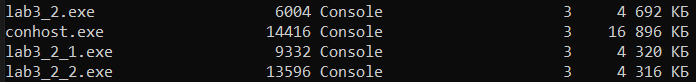
OS03\_02\_02.exe



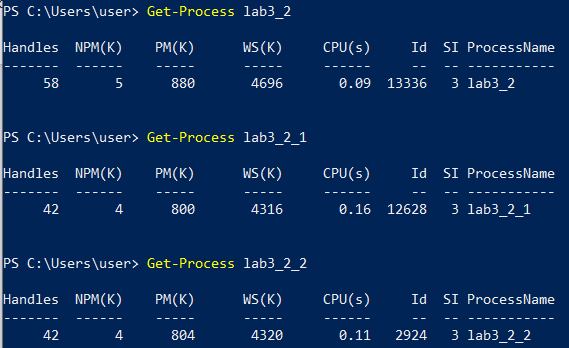
Task manager



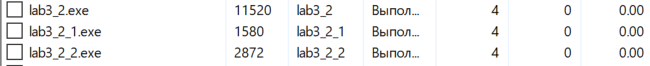
Tasklist



PowerShell ISE



Performance monitor

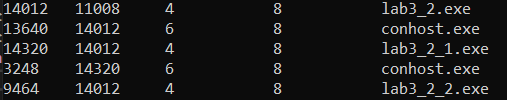


Задание 3

Листинг OS03\_03

|  |
| --- |
| #include <windows.h>  #include <tlhelp32.h>  #include <iostream>  #include <iomanip>  int main()  {  setlocale(LC\_ALL, "Russian");  HANDLE hProcessSnap = CreateToolhelp32Snapshot(TH32CS\_SNAPPROCESS, 0); |
| if (hProcessSnap == INVALID\_HANDLE\_VALUE)  {  std::cout << "CreateToolhelp32Snapshot failed. Error: " << GetLastError() << std::endl;  return 1;  }  PROCESSENTRY32W pe32;  pe32.dwSize = sizeof(pe32);  if (!Process32FirstW(hProcessSnap, &pe32))  {  std::cout << "Process32First failed. Error: " << GetLastError() << std::endl;  CloseHandle(hProcessSnap);  return 1;  }  std::cout << std::left << std::setw(8) << "PID"  << std::setw(10) << "PPID"  << std::setw(12) << "Threads"  << std::setw(12) << "Priority"  << "Process Name" << std::endl;  std::cout << std::string(60, '-') << std::endl;  do  {  std::wcout << std::left << std::setw(8) << pe32.th32ProcessID  << std::setw(10) << pe32.th32ParentProcessID  << std::setw(12) << pe32.cntThreads  << std::setw(12) << pe32.pcPriClassBase  << pe32.szExeFile << std::endl;  } while (Process32NextW(hProcessSnap, &pe32));  CloseHandle(hProcessSnap);  std::cout << "\nНажмите любую клавишу для выхода...";  std::cin.get();  return 0;  } |

Вывод консоли



Задание 4

Листинг OS03\_04

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h>

int main() {

pid\_t pid = getpid();

for (int i = 0; i < 20; i++) {

printf("Cycle[%d] process %d \n", i,pid);

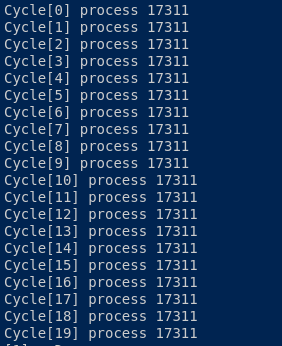
sleep(1);

}

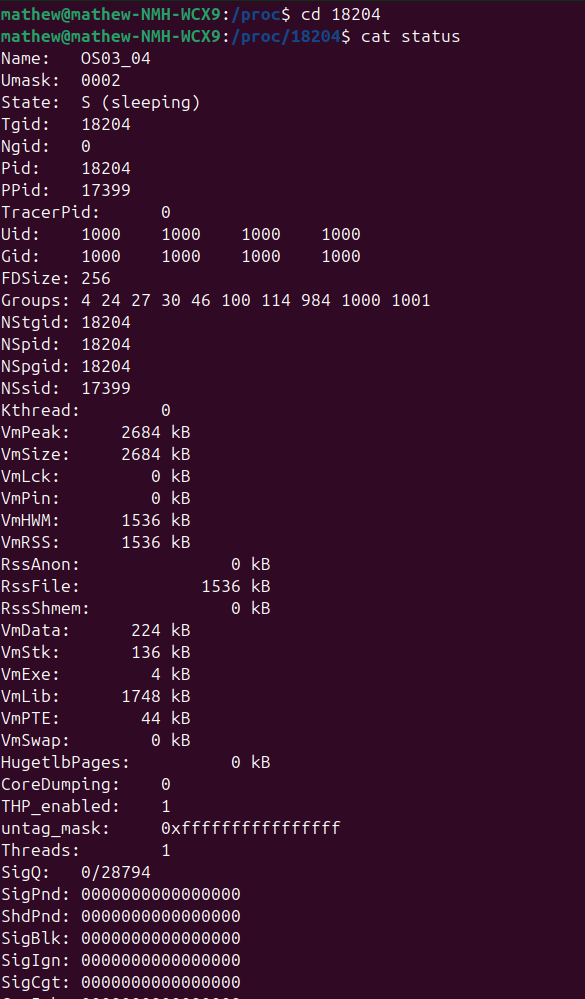
return 0;

}

Вывод терминала



/proc



/ps



Задание 5

Листинг OS03\_05

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/wait.h>

int main() {

pid\_t pid = getpid();

printf("OS03\_05 (Parent) Process ID: %d\n", pid);

pid\_t child\_pid = fork();

if (child\_pid == 0) {

pid\_t child\_pid = getpid();

printf("OS03\_05\_1 (Child) Process ID: %d\n", child\_pid);

for (int i = 0; i < 50; i++) {

printf("OS03\_05\_1 (Child) Cycle %d\n", i);

sleep(1);

}

} else if (child\_pid > 0) {

for (int i = 0; i < 100; i++) {

printf("OS03\_05 (Parent) Cycle %d\n", i);

sleep(1);

}

waitpid(child\_pid, NULL, 0);

} else {

printf("Error: Failed to create child process.\n");

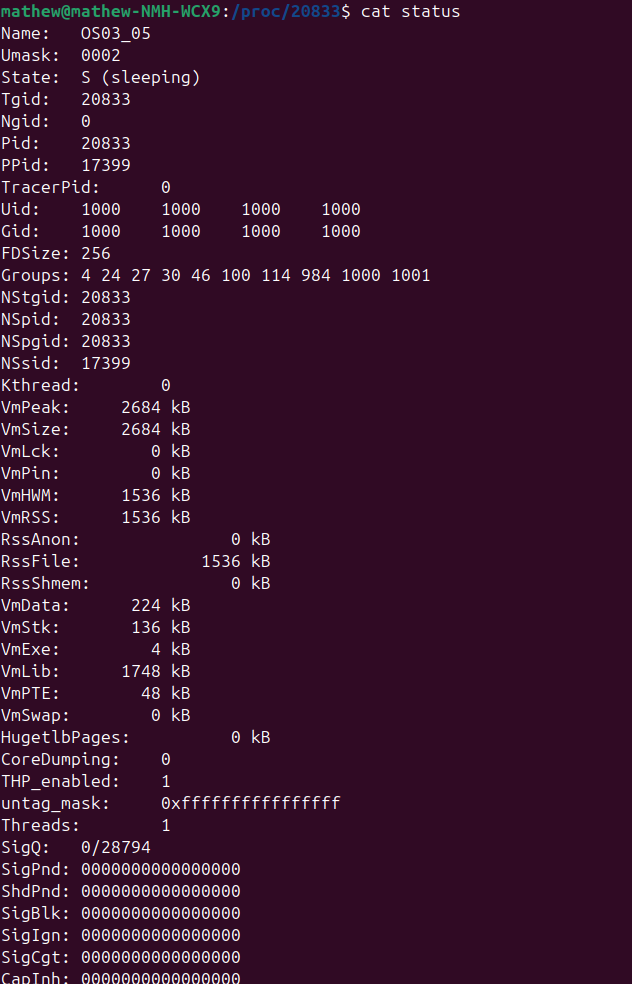
return 1;

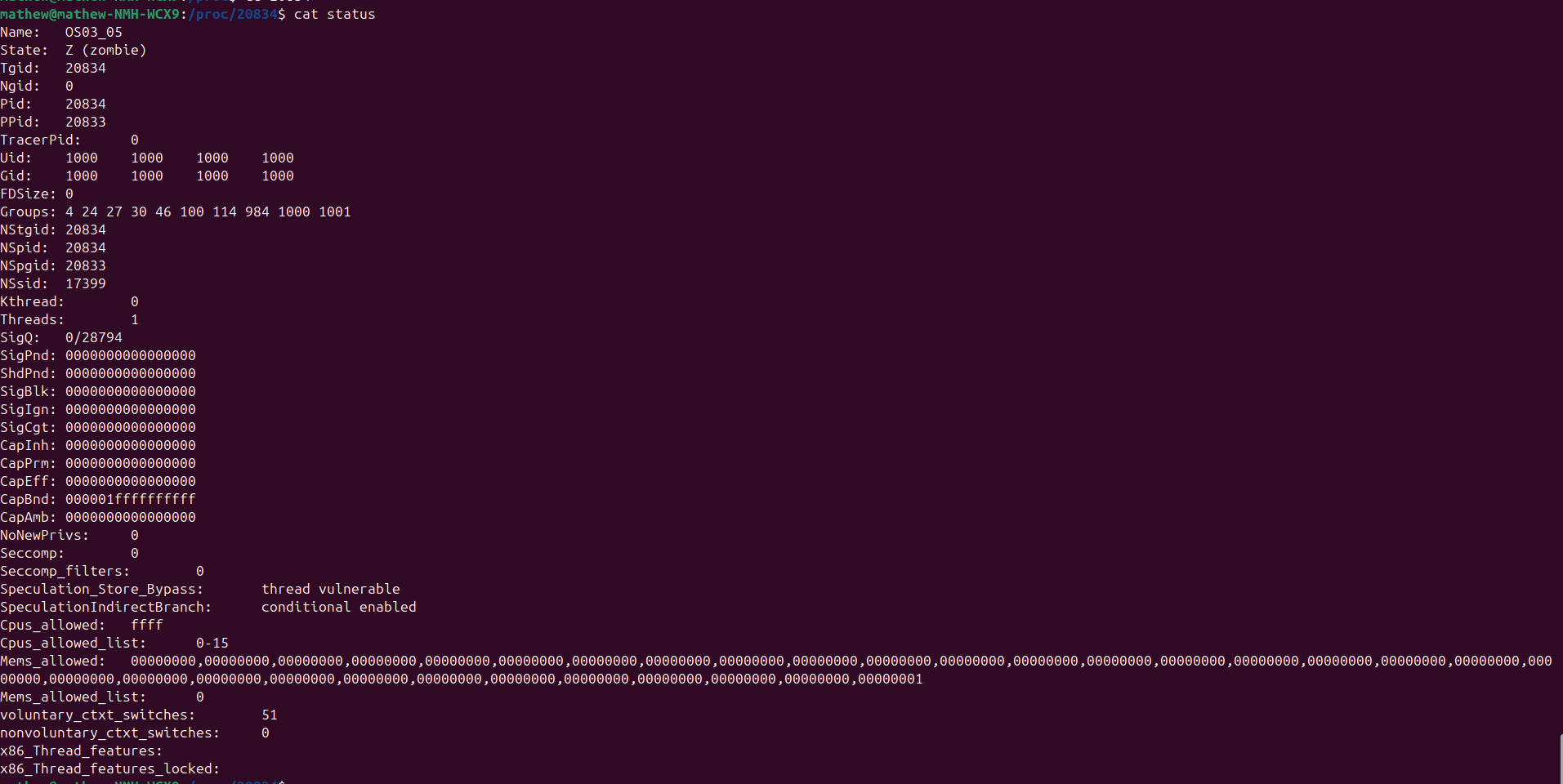
}

return 0;

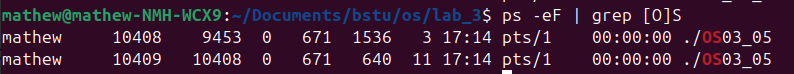
}

/proc





ps



Задание 6

Листинг OS03\_06

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/wait.h>

int main(){

for (int i = 0; i < 50; i++) {

printf("OS03\_05\_1 (Child) Cycle[%d] %d\n",i , getpid());

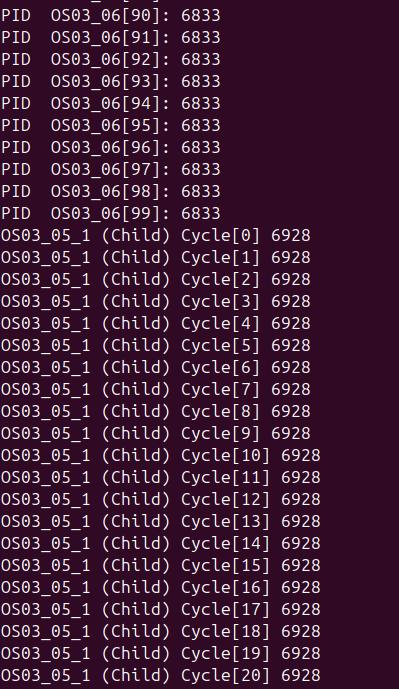
sleep(1);

}

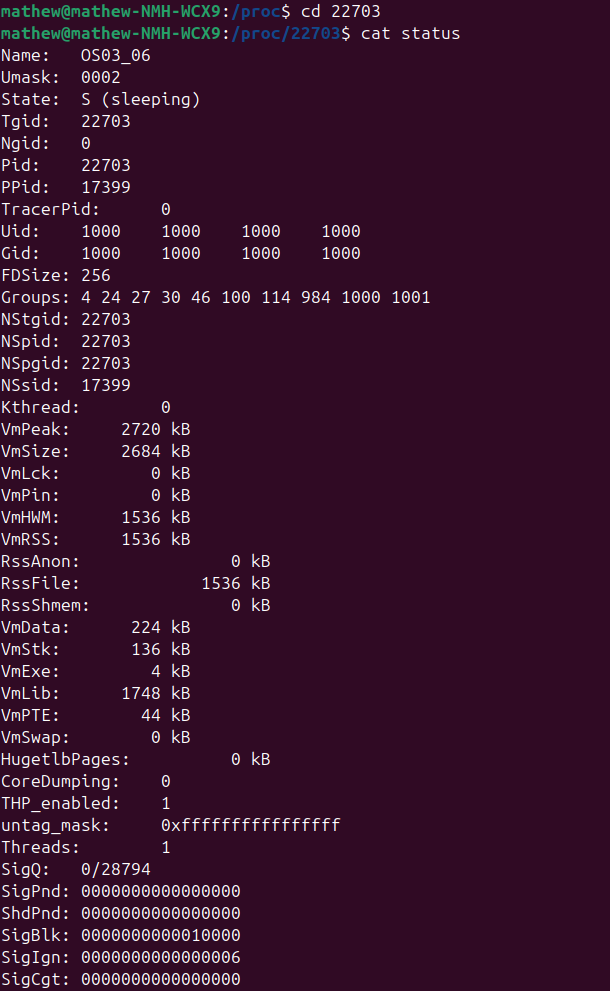
return 0;

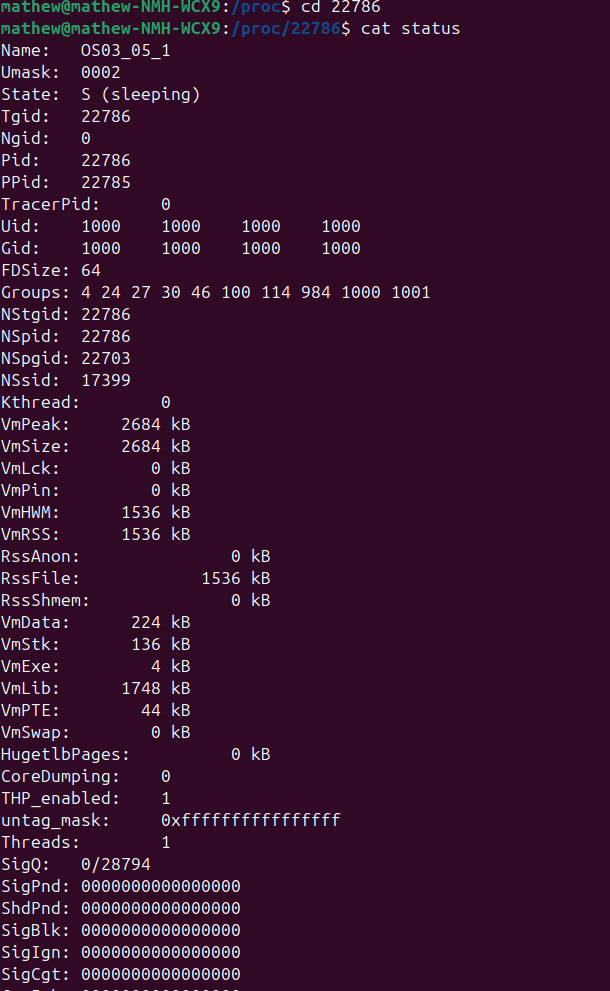
}

Вывод терминала

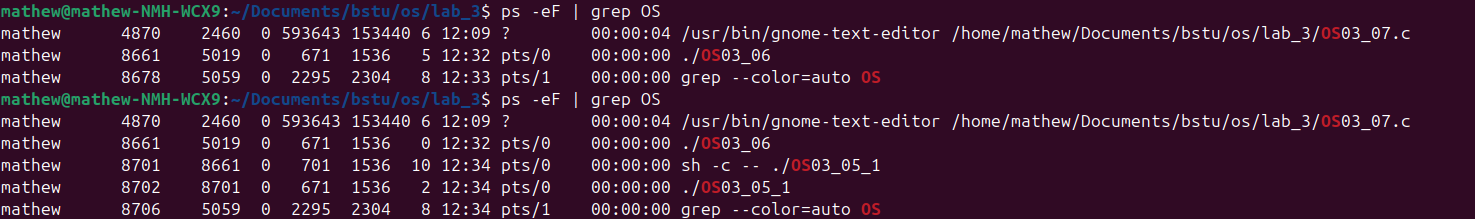


/proc





ps



Задание 7

Листинг OS03\_07

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h>

int main() {

for (int i = 0; i < 100; i++) {

printf("PID OS03\_06[%d]: %d\n",i, getpid());

execl("./OS03\_05\_1", "OS03\_05\_1", NULL);

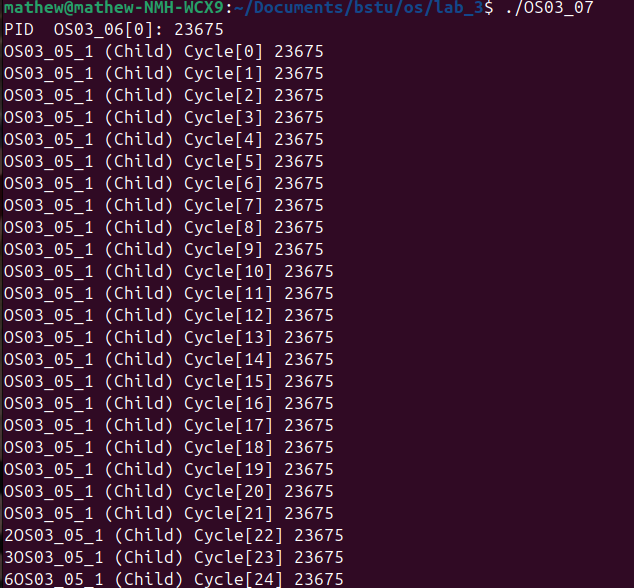
sleep(1);

}

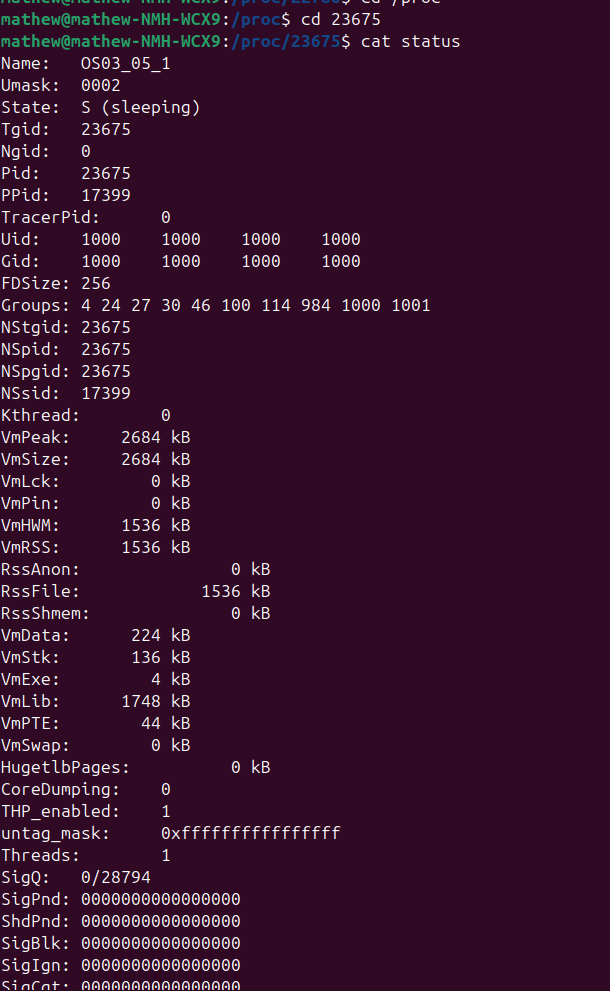
return 0;

}

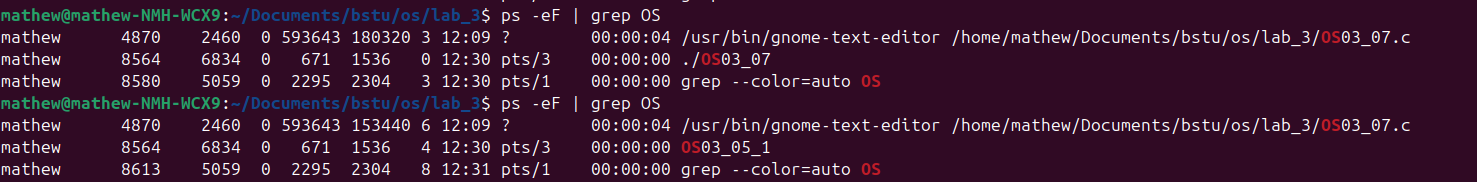
Вывод терминала



/proc



ps



Вопросы  
27. Что такое процесс?  
Процесс - это экземпляр выполняемой программы в операционной системе. Каждый процесс имеет свое собственное адресное пространство, ресурсы (память, файлы, сетевые соединения и т.д.) и контекст выполнения.

1. Что такое контекст процесса?

Контекст процесса - это совокупность всех данных, связанных с выполнением процесса. Это включает в себя:

Регистры процессора

Указатель стека

Счетчик команд

Открытые файлы и сетевые соединения

Информация о планировании (приоритет, время выполнения и т.д.)  
Контекст процесса позволяет ОС переключаться между различными процессами, сохраняя и восстанавливая их состояние.

1. Что такое родительский и дочерний процесс?

Родительский процесс - это процесс, который создал другой процесс (дочерний).

Дочерний процесс - это процесс, который был создан другим процессом (родительским). Дочерний процесс наследует некоторые ресурсы и свойства родительского процесса.

1. Что такое процесс инициализации OS?

(обычно называемый init или systemd) является первым процессом, запускаемым ядром операционной системы Linux при загрузке. Он отвечает за:

Запуск других системных процессов

Управление жизненным циклом других процессов

Обработку сигналов и событий, связанных с системой

Обеспечение взаимодействия между пользовательскими процессами и ядром ОС  
Процесс инициализации является родительским для всех других процессов в системе и играет ключевую роль в запуске и управлении ОС.

1. Перечислите области памяти процесса и поясните их назначение.

Код (Text) - содержит машинные инструкции, то есть исполняемый код программы.

Данные (Data) - хранит глобальные и статические переменные, инициализированные значения.

Стек (Stack) - используется для хранения локальных переменных функций и параметров при вызове функций.

Куча (Heap) - динамическая память, выделяемая программой во время выполнения (с помощью malloc(), new и т.д.).

Shared Libraries - область для загружаемых динамических библиотек, используемых процессом.

1. Чем отличаются системные процессы от пользовательских?

Системные процессы (например, ядро, драйверы, демоны) запускаются при загрузке ОС, выполняют важные системные функции и работают в привилегированном режиме ядра.

Пользовательские процессы запускаются по запросу пользователя или приложений, работают в пользовательском режиме с меньшим набором привилегий.

Системные процессы имеют больше прав доступа к аппаратным ресурсам, в то время как пользовательские процессы ограничены в правах.

Системные процессы обеспечивают основные сервисы ОС, в то время как пользовательские процессы выполняют конкретные задачи пользователя.

1. Что такое Windows-сервисы, Linux-демоны?

34. С помощью каких системных вызовов можно создать дочерний процесс в Windows? Поясните разницу.

35. С помощью каких системных вызовов можно создать дочерний процесс в Linux? Поясните разницу.

36. Какие потоки данных доступны любому процессу автоматически?

37. Поясните назначение системного вызова WaitForSingleObject в Windows-приложении.

38. Поясните назначение системного вызова wait в Linux-приложении.

39. Дайте развернутое определение процесса OS.