МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

**«Московский технический университет связи и информатики»**

Кафедра «Системное программирование»

Лабораторная работа №9

**«Исследование механизмов выделения памяти»**

по дисциплине

Операционные системы

Выполнил: студент гр. БЭИ2202

Кулешов А. С.

Вариант 16

Проверил: Королькова Т.В.

Москва, 2025 г

1.

Напишите программу, работающую в одном из 2-х режимов, режим

работы задается при запуске в качестве аргумента командной строки:



режим1: выделение памяти malloc() равно освобождению памяти free()



режим2: выделение памяти malloc() больше, чем освобождение памяти

free()

Выделение и освобождение памяти выполняйте параллельно, с

задержкой 100 – 1000 нс в бесконечном цикле, для второго режима работы

выделение памяти может быть, например, в 2 раза чаще, чем освобождение.

Предусмотрите прерывание процесса по сигналу, направленному

пользователем программы асинхронно (ctrl+c, для освобождения всей

динамически выделенной памяти.

2.

Напишите программу для создания карты виртуальной памяти процесса,

идентификатор которого передается первым аргументом, а полный путь до

директории сохранения файла карты памяти – вторым аргументом. Именовать

файлы карты по шаблону map\_<pid>\_<date&time>, где <pid> - идентификатор

процесса, переданный в программу и соответствующий процессу, для которого

создается карта, <date&time> - дата и время создания карты в формате yyyy-mm

dd\_h:m:s. Наполнение карты возьмите из соответствующего процесса в /proc/

или используйте утилиту pmap. Проверьте работу программы для процесса

запущенной программы из п. 1.

3.

Включите в отчет ответы на вопросы в свободной форме:

 Какую информацию содержит карта виртуальной памяти процесса?

 Что такое стек и куча?

 Как происходит статическое и динамическое выделение памяти?

 Каковы плюсы и минусы статического и динамического выделения

памяти?

**ВЫПОЛНЕНИЕ**



Рисунок 1 – Задание 1. Запуск без аргумента

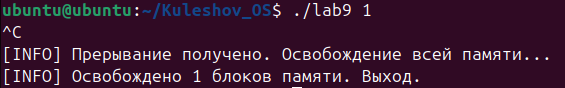


Рисунок 2 – Задание 1. Запуск в обычном режиме

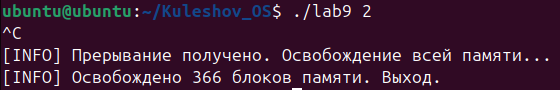


Рисунок 3 – Задание 1. Запуск с утечкой памяти



Рисунок 4 – Задание 2. Запуск без аргумента



Рисунок 5 – Задание 2. Получение pid



Рисунок 6 – Задание 2. Запуск. Успешный

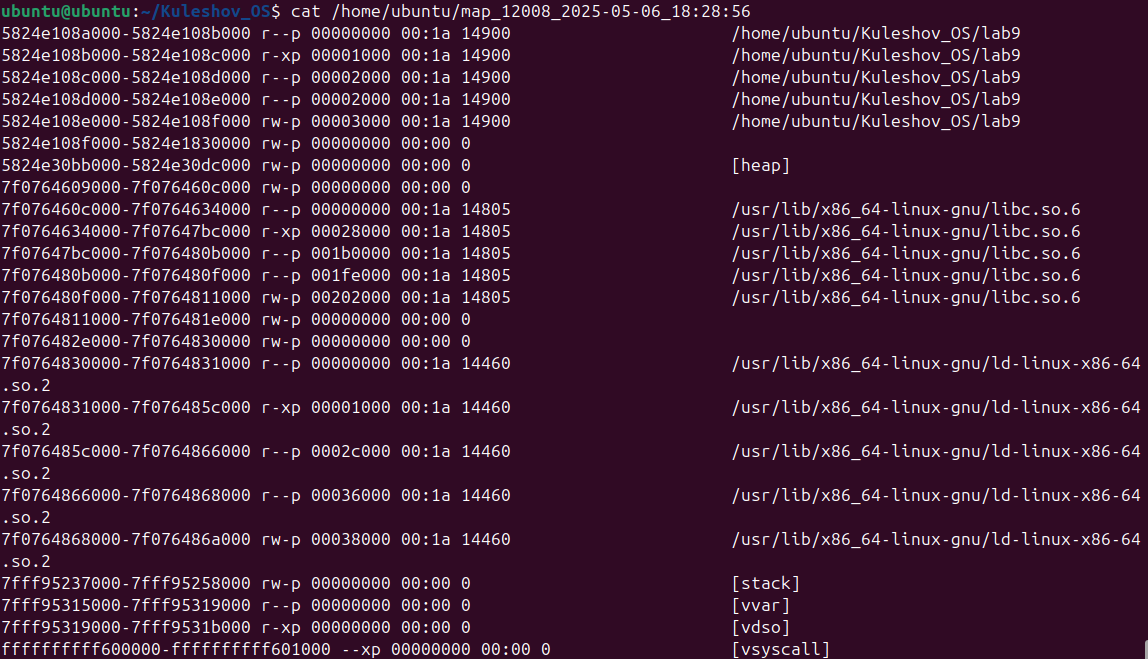


Рисунок 7 – Задание 2. Полученная карта

Вопрос 1. Какую информацию содержит карта памяти?

- Диапазоны адресов виртуальной памяти. Права доступа. Индекс файла в файловой системе.

Вопрос 2. Что такое стек и куча

- Стек хранит в себе как правило локальные данные при помощи механизма добавления и удаления по принципу LiFo (Last in, First out). Куча же предполагает ручное выделение и удаление памяти. Растёт как правило в положительную сторону адресов

Вопрос 3. Как происходит статическое и динамическое выделение памяти?

- Статическое выделение происходит на стадии компиляции. Данные существуют в весь период жизни программы. Динамические выделются во время работы программы

Вопрос 4. Какие плюсы и минусы статического и динамического выделения памяти?

- Статическая память быстрее, а ещё с ней меньше проблем с утечками. Однако выделить нефиксированное число памяти не выйдет. Также размер меньше чем у динамической

Код для задания 1.

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <signal.h>

#include <string.h>

#include <time.h>

#define MAX\_PTRS 1000000

**volatile** **sig\_atomic\_t** stop = **0**;

**void**\* allocated\_ptrs[MAX\_PTRS];

**size\_t** allocated\_count = **0**;

**void** **handle\_sigint**(**int** sig) {

stop = **1**;

printf("**\n**[INFO] Прерывание получено. Освобождение всей памяти...**\n**");

**for** (**size\_t** i = **0**; i < allocated\_count; ++i) {

free(allocated\_ptrs[i]);

}

printf("[INFO] Освобождено %zu блоков памяти. Выход.**\n**", allocated\_count);

exit(**0**);

}

**void** **nanosleep\_random**() {

**struct** timespec ts;

ts.tv\_sec = **0**;

ts.tv\_nsec = **100** + rand() % **901**; // от 100 до 1000 нс

nanosleep(&ts, NULL);

}

**int** **main**(**int** argc, **char**\* argv[]) {

**if** (argc != **2**) {

fprintf(stderr, "Использование: %s режим(1 или 2)**\n**", argv[**0**]);

**return** **1**;

}

**int** mode = atoi(argv[**1**]);

**if** (mode != **1** && mode != **2**) {

fprintf(stderr, "Неверный режим. Используйте 1 или 2.**\n**");

**return** **1**;

}

signal(SIGINT, handle\_sigint);

srand(time(NULL));

**size\_t** alloc\_rate = (mode == **1**) ? **1** : **2**;

**size\_t** free\_rate = **1**;

**while** (!stop) {

// Выделяем память

**for** (**size\_t** i = **0**; i < alloc\_rate; ++i) {

**if** (allocated\_count < MAX\_PTRS) {

**void**\* ptr = malloc(**64**); // Выделим 64 байта

**if** (ptr != NULL) {

allocated\_ptrs[allocated\_count++] = ptr;

}

}

nanosleep\_random();

}

// Освобождаем память

**for** (**size\_t** i = **0**; i < free\_rate; ++i) {

**if** (allocated\_count > **0**) {

free(allocated\_ptrs[--allocated\_count]);

}

nanosleep\_random();

}

}

**return** **0**;

}

Код для задания 2.

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <time.h>

#include <string.h>

#include <sys/stat.h>

#include <dirent.h>

**void** **create\_memory\_map**(**pid\_t** pid, **const** **char** \*output\_dir) {

**time\_t** now = time(NULL);

**struct** tm \*tm\_info = localtime(&now);

**char** timestamp[**20**];

strftime(timestamp, **sizeof**(timestamp), "%Y-%m-%d\_%H:%M:%S", tm\_info);

**char** filename[**256**];

snprintf(filename, **sizeof**(filename), "%s/map\_%d\_%s", output\_dir, pid, timestamp);

**char** proc\_path[**64**];

snprintf(proc\_path, **sizeof**(proc\_path), "/proc/%d/maps", pid);

**FILE** \*src = fopen(proc\_path, "r");

**if** (!src) {

perror("Не смогли открыть /proc/[pid]/maps");

exit(**1**);

}

**FILE** \*dst = fopen(filename, "w");

**if** (!dst) {

perror("Не смогли создать выходной файл");

fclose(src);

exit(**1**);

}

**char** buffer[**4096**];

**size\_t** bytes;

**while** ((bytes = fread(buffer, **1**, **sizeof**(buffer), src)) > **0**) {

fwrite(buffer, **1**, bytes, dst);

}

fclose(src);

fclose(dst);

printf("Карта памяти создана: %s**\n**", filename);

}

**int** **main**(**int** argc, **char** \*argv[]) {

**if** (argc != **3**) {

printf("Использование: %s <pid> <output\_directory>**\n**", argv[**0**]);

**return** **1**;

}

**pid\_t** pid = atoi(argv[**1**]);

**const** **char** \*output\_dir = argv[**2**];

**DIR** \*dir = opendir(output\_dir);

**if** (!dir) {

perror("Выходная директория не существует");

**return** **1**;

}

closedir(dir);

create\_memory\_map(pid, output\_dir);

**return** **0**;

}