**Министерство науки и высшего образования Российской Федерации**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Факультет безопасности информационных технологий**

**Дисциплина:**

«Операционные системы»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6**

«Функция malloc»

Содержание

[Введение 3](#_Toc182062976)

[1 Malloc 4](#_Toc182062977)

[1.1 Задание 4](#_Toc182062978)

[1.2 Ход работы 4](#_Toc182062979)

[1.2.1 Функции выделения памяти 4](#_Toc182062980)

[1.2.2 Тестирование malloc 5](#_Toc182062981)

[1.2.3 График зависимости для malloc 7](#_Toc182062982)

[1.2.4 Тестирование calloc 7](#_Toc182062983)

[1.2.5 График зависимости для calloc 9](#_Toc182062984)

[1.2.6 Тестирование jemalloc 9](#_Toc182062985)

[1.2.7 График зависимости для jemalloc 10](#_Toc182062986)

[1.2.8 Тестирование tcmalloc 11](#_Toc182062987)

[1.2.9 График зависимости для tcmalloc 13](#_Toc182062988)

[1.2.10 Общее сравнение 13](#_Toc182062989)

[Заключение 14](#_Toc182062990)

[Список использованных источников 15](#_Toc182062991)

Введение

Цель работы – протестировать функцию malloc/free и построить график зависимости времени выделения от размера запрашиваемой памяти. Сравнить с другими малоками

Для достижения поставленной цели необходимо решить следующие задачи:

* познакомиться с malloc;
* написать тесты;
* построить графики зависимость;
* сравнить малоки.

# Malloc

## Задание

Протестировать функцию malloc/free и построить график зависимости времени выделения от размера запрашиваемой памяти. Сравнить с другими малоками.

## Ход работы

### Функции выделения памяти

Функции выделения памяти — это функции, которые динамически выделяют память в соответствии со значениями параметров и возвращают адрес начала выделенного участка памяти.

malloc — это функция, которая выделяет блок памяти и возвращает указатель на его начало. Через полученный указатель можно помещать данные в выделенную память. Если выделить память не удалось, функция возвращает NULL. Так как указатель всегда имеет тип void, его необходимо явно приводить к нужному типу. malloc выделяет память в области, называемой "куча" (heap) в оперативной памяти, которая используется для динамического хранения данных. Это позволяет избегать использования статически выделенной памяти, что может быть недостаточно в некоторых ситуация. Память, выделенная с помощью malloc, не инициализируется, т.е. в ней могут находиться произвольные значения. Это важно учитывать, так как перед использованием выделенной памяти необходимо либо инициализировать ее, либо удостовериться, что она не содержит старых данных.

calloc – это функция, которая выделяет блок памяти для массива размером — num элементов, каждый из которых занимает size байт, и инициализирует все свои биты нулями. В результате выделяется блок памяти размером number \* size байт, причём весь блок заполнен нулями. Возвращает данная функция указатель на выделенный блок памяти. Тип данных, на который ссылается указатель всегда void\*, поэтому это тип данных может быть приведен к желаемому типом данных. Если функции не удалось выделить требуемый блок памяти, возвращается нулевой указатель.

jemalloc – это реализация функций malloc, оптимизированная для решения проблем фрагментации и оптимальной утилизации ресурсов CPU при выделения памяти в нескольких потоках на многопроцессорных системах. Использование jemalloc позволяет увеличить производительность, сократив при этом фрагментацию и в связи с этим требуя для своей работы меньше памяти.

tcmalloc - в основе данной функции лежит идея разделения памяти на несколько уровней ради уменьшения фрагментации памяти. Внутри TCMalloc управление памятью делится на две части: работа с памятью потоков и работа с кучей. Память потоков разбивается на последовательность фрагментов определённых размеров, выбираемых в соответствии с классами размеров. Это позволяет эффективно выделять память под объекты, размер которых меньше или равен 32 Кб. Куча, управляемая TCMalloc, представляет собой коллекцию страниц, в которой набор последовательных страниц может быть представлен в виде диапазона страниц (span). Когда нужно выделить память под объект, размер которого превышает 32 Кб, для выделения памяти используется куча.

### Тестирование malloc

Для тестирования malloc был написан следующий код си:

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

int main() {

FILE \*file = fopen("malloc.txt", "w");

for (int count = 2; count <= 268435456; count = count \* 2) {

clock\_t start = clock();

int\* ptr = (int\*) malloc(count \* sizeof(int));

clock\_t end = clock();

double time = (double)(end - start) / CLOCKS\_PER\_SEC;

printf("malloc %lu %f ", count \* sizeof(int), time);

fprintf(file, "malloc %lu %f ", count \* sizeof(int), time);

start = clock();

free(ptr);

end = clock();

time = (double)(end - start) / CLOCKS\_PER\_SEC;

printf("free %lu %f\n", count \* sizeof(int), time);

fprintf(file, "free %lu %f\n", count \* sizeof(int), time);

}

fclose(file);

return 0;

}

Листинг 1 – тестирование malloc

После запуска программы в терминале получаем результаты тестирования.

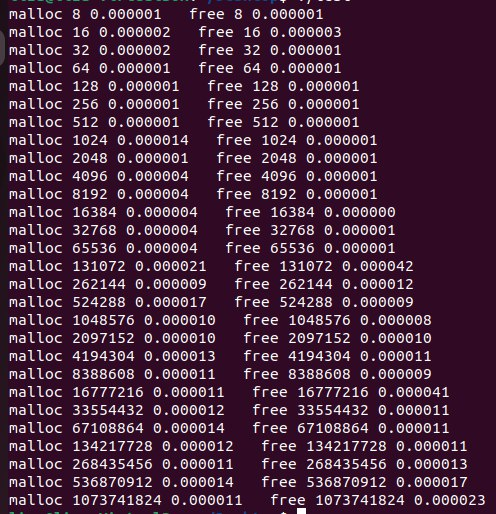


Рисунок 1 – результат тестирования malloc

### График зависимости для malloc

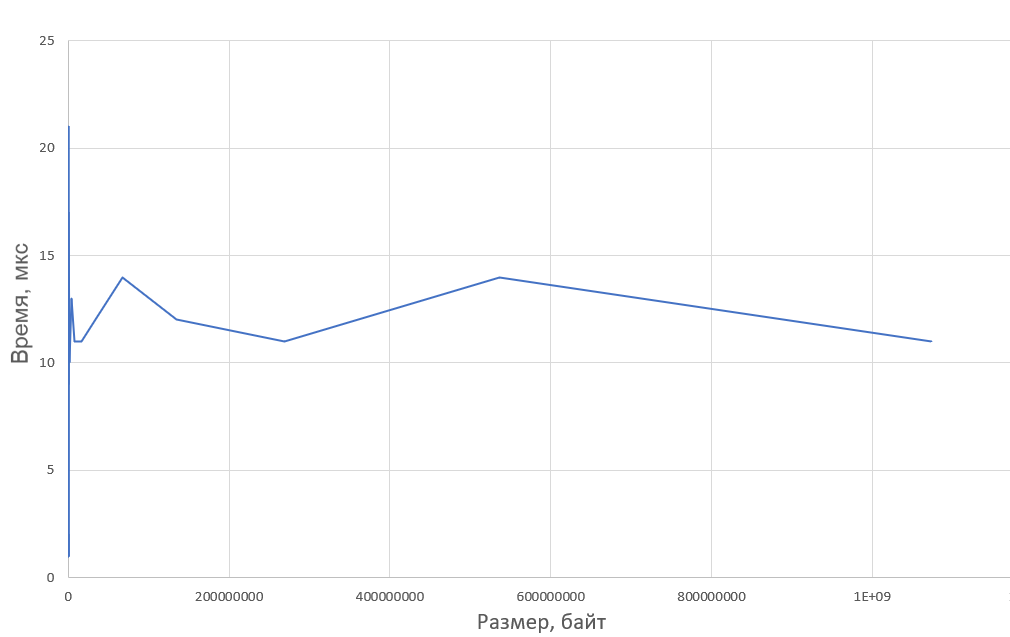


Рисунок 2 - график зависимости времени выделения от размера запрашиваемой памяти для malloc

### Тестирование calloc

Для тестирования calloc был написан следующий код си:

#include <stdlib.h>

#include <time.h>

int main() {

FILE\* file = fopen("calloc.txt", "w");

for (int count = 2; count <= 268435456; count \*= 2) {

clock\_t start = clock();

int\* ptr = (int\*)calloc(count, sizeof(int));

clock\_t end = clock();

double time = (double)(end - start) / CLOCKS\_PER\_SEC;

printf("calloc %lu %f\n", count \* sizeof(int), time);

fprintf(file, "calloc %lu %f\n", count \* sizeof(int), time);

free(ptr);

}

fclose(file);

return 0;

}

Листинг 2 – тестирование calloc

После запуска программы в терминале получаем результаты тестирования.

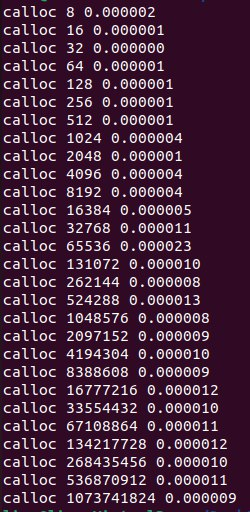


Рисунок 3 - результат тестирования calloc

### График зависимости для calloc

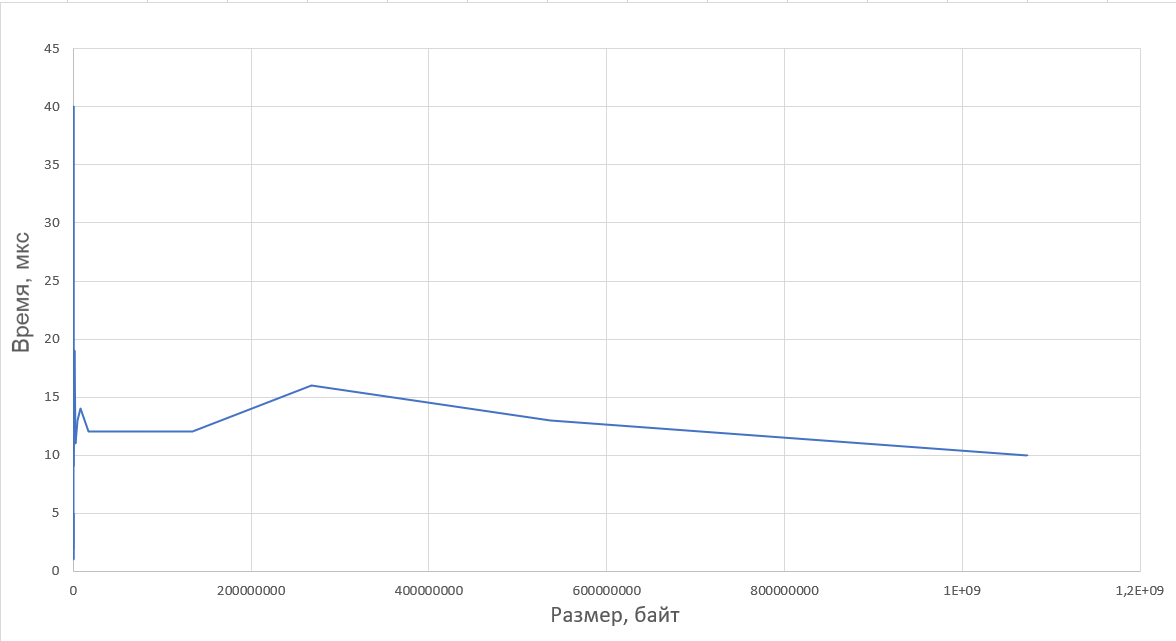


Рисунок 4 - график зависимости времени выделения от размера запрашиваемой памяти для calloc

### Тестирование jemalloc

Для тестирования jemalloc был написан следующий код си:

#include <time.h>

#include <jemalloc/jemalloc.h>

int main() {

FILE \*file = fopen("malloc.txt", "w");

for (int count = 2; count <= 268435456; count = count \* 2) {

clock\_t start = clock();

int\* ptr = (int\*)malloc(count \* sizeof(int));

clock\_t end = clock();

double time = (double)(end - start) / CLOCKS\_PER\_SEC;

printf("jemalloc %lu %f\n", count \* sizeof(int), time);

fprintf(file, "jemalloc %lu %f\n", count \* sizeof(int), time);

free(ptr);

}

fclose(file);

return 0;

}

Листинг 3 – тестирование jemalloc

После запуска программы в терминале получаем результаты тестирования.

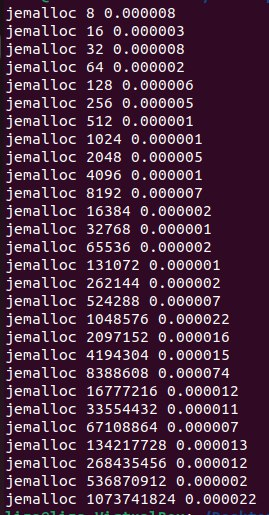


Рисунок 5 - результат тестирования jemalloc

### График зависимости для jemalloc

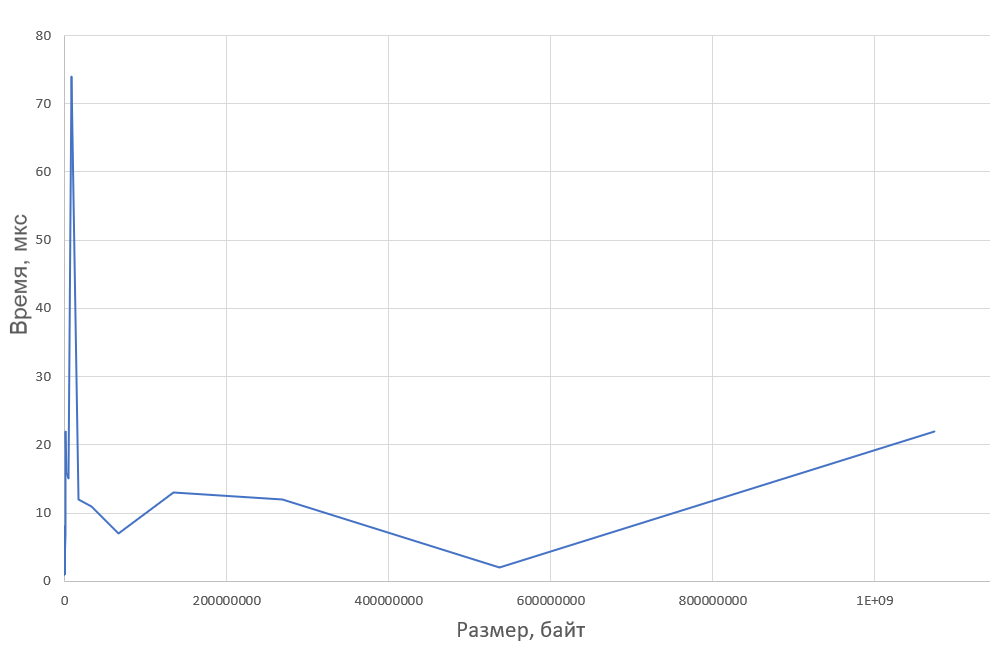


Рисунок 6 - график зависимости времени выделения от размера запрашиваемой памяти для jemalloc

### Тестирование tcmalloc

Для тестирования tcmalloc был написан следующий код си:

#include <stdio.h>

#include <time.h>

#include <gperftools/tcmalloc.h>

int main() {

FILE \*file = fopen("malloc.txt", "w");

for (int count = 2; count <= 268435456; count = count \* 2) {

clock\_t start = clock();

int\* ptr = (int\*) tc\_malloc(count \* sizeof(int));

clock\_t end = clock();

double time = (double)(end - start) / CLOCKS\_PER\_SEC;

printf("tcmalloc %lu %f\n", count \* sizeof(int), time);

fprintf(file, "tcmalloc %lu %f\n", count \* sizeof(int), time);

tc\_free(ptr);

}

fclose(file);

return 0;

}

Листинг 4 – тестирование tcmalloc

После запуска программы в терминале получаем результаты тестирования.

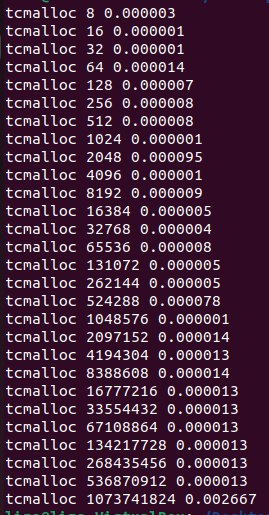


Рисунок 7 - результат тестирования tcmalloc

### График зависимости для tcmalloc

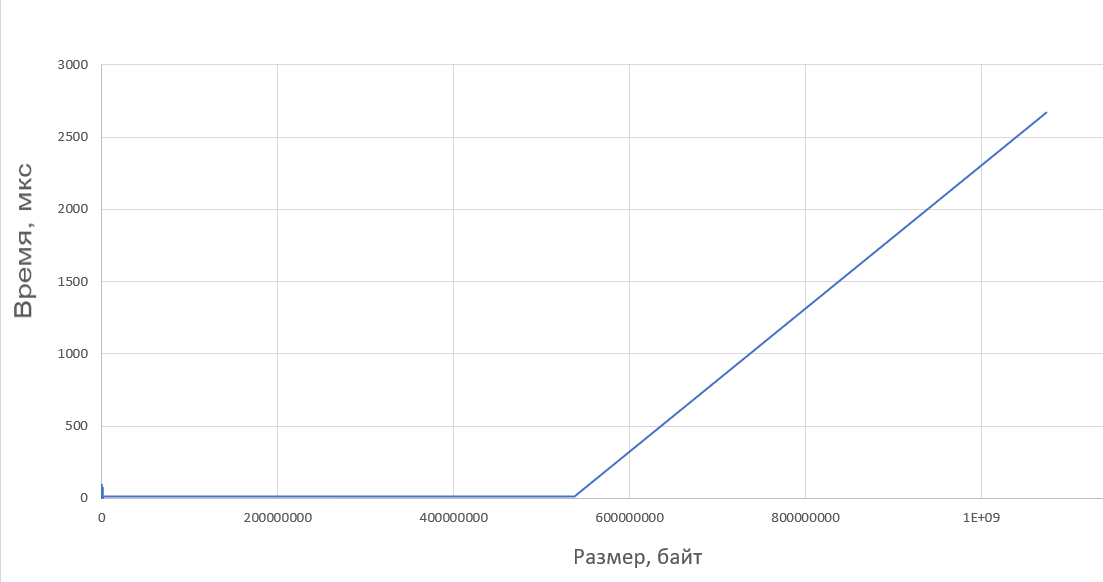


Рисунок 8 - график зависимости времени выделения от размера запрашиваемой памяти для tcmalloc

### Общее сравнение

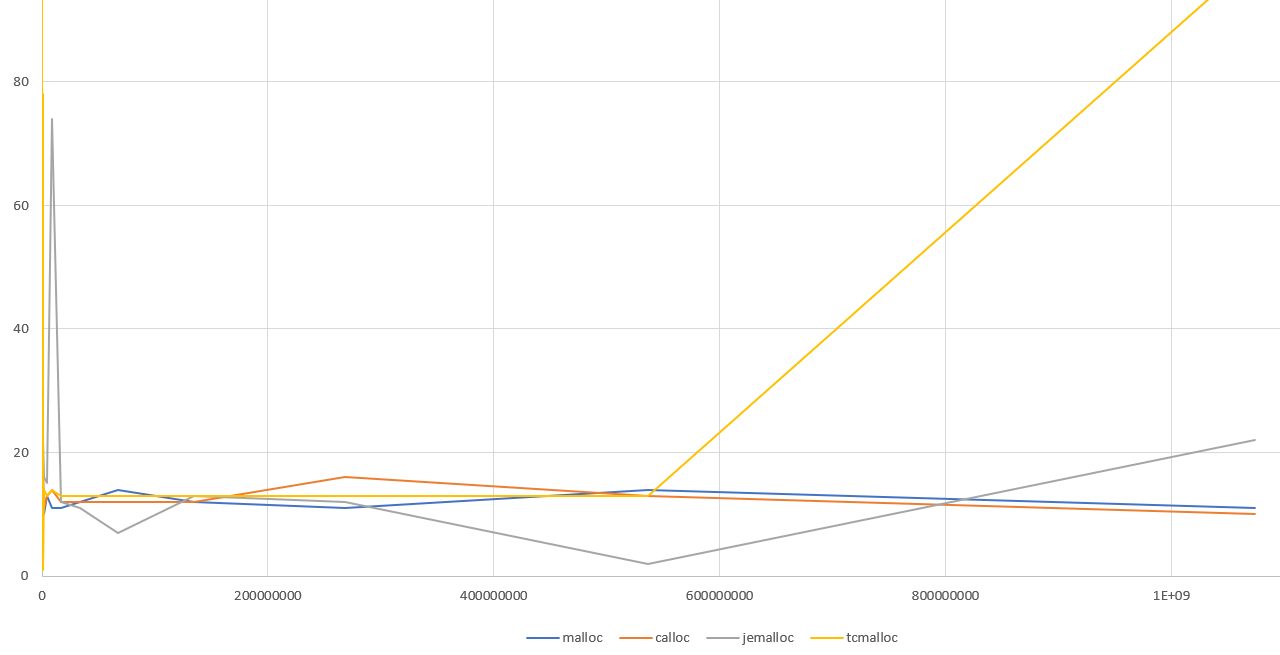


Рисунок 9 – сравнение функций выделения памяти.

Заключение

В ходе данной лабораторной работы мы познакомились с функциями выделения памяти, а именно malloc, calloc, jemalloc, tcmalloc. Также были написаны тесты для каждой из функций, на основе которых были построены графики зависимости времени выделения от размера запрашиваемой памяти. После этого было проведено сравнение данных функций. Это позволило закрепить полученные знания и навыки. В ходе выполнения лабораторной работы были выполнены все задачи и достигнуты поставленные цели.

Список использованных источников

1. Эндрю Таненбаум Современные операционные системы. — 4-е изд. — СПб. : Питер, 2015. — 1120 с.