

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО  
ОБРАЗОВАНИЯ  
«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
УНИВЕРСИТЕТ ИТМО»**

**Факультет безопасности информационных технологий**

**Дисциплина:  
«Операционные системы»**

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6**  
Тестирование функций malloc/free

**Выполнила:**  
Студентка гр. №N3253  
Пастухова А.А.



**Проверил:**  
Ханов А.Р.

Санкт-Петербург  
2022 г.

### **Задачи:**

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

Сложный:

Сравнить с другими mallocками

### **Ход работы:**

**Malloc** – функция выделяет блок памяти, размером `size_t` байт, и возвращает указатель на начало блока. Содержание выделенного блока памяти не инициализируется, оно остается с неопределенными значениями. Если память выделить не удалось, то функция возвращает NULL. Так как malloc возвращает указатель типа `void*`, то его необходимо явно приводить к нужному нам типу.

**Calloc** – функция выделяет место для хранения массива числовых элементов, каждый из которых имеет размер в байтах. Каждый элемент инициализируется значением 0. Возвращает указатель на выделенное пространство. Пространство в хранилище, на которое указывает возвращаемое значение, гарантированно выровнено подходящим для хранения любого типа объектов образом.

**Dlmalloc** - одна из реализаций стандартного malloc, в котором есть некоторые особенности: группировка свободных блоков по размерам. Это означает, что есть фиксированное множество из 128 размеров блоков, по которым производится поиск наиболее подходящего блока. Кроме того, при хранении свободных блоков в dlmalloc задействован freelist, хранящий указатель на соседние для данного размера блока участки внутри самих блоков. Объединение свободных блоков в бóльший блок происходит не сразу, а через какое-то время, ведь велика вероятность, что раз такой блок только что удалили, то скоро попросят что-то такого же или близкого размера обратно. Иногда при нехватке блоков маленького размера большой свободный блок просто делится на блок нужного размера и оставшуюся часть. Потому такой блок стали делить сразу на много блоков поменьше, чтобы не тратить время на «откусывание» по чуть-чуть несколько раз в будущем, т.к., вероятно, пользователю понадобится ещё несколько блоков похожего размера.

**Free** – функция освобождения блоков памяти, после чего они помещаются в корень дерева свободных блоков. Число освобожденных байтов эквивалентно количеству байтов, запрошенному при выделении блока (или

выделении заново, если использовалась функция `realloc`). Если блок памяти имеет значение `NULL`, указатель не обрабатывается, и функция `free` немедленно возвращает управление.

### Код главной программы C++ (пример для `calloc`)

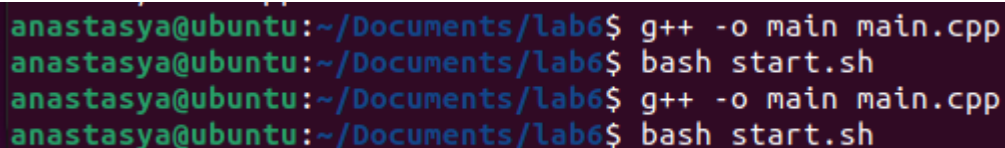
Для функции `dmalloc` нужно использовать заголовочный файл `#include "alloc.h"`

```
#include <iostream>
#include <ctime>
#include <stdlib.h>
#include <unistd.h>
#include <memory.h>
#include <fstream>
#include "alloc.h"
using namespace std;
int main(int argc, char *argv[]) {
    ofstream fout("calloc5.txt", ios::app);
    int size = atoi(argv[1]);
    clock_t timer = clock();
    char* bf = (char*) calloc(1, size);
    free(bf);
    fout << (1000.0 * (clock() - timer) / CLOCKS_PER_SEC) << "\n";
    fout.close();
    return 0;
}
```

### Код отдельного запуска циклов `bash`

```
#!/bin/bash
for (( i=1; c <= 4096; i++ ))
do
    .main $i
done
```

Запуск программ:



```
anastasya@ubuntu:~/Documents/lab6$ g++ -o main main.cpp
anastasya@ubuntu:~/Documents/lab6$ bash start.sh
anastasya@ubuntu:~/Documents/lab6$ g++ -o main main.cpp
anastasya@ubuntu:~/Documents/lab6$ bash start.sh
```

График зависимости времени выделения от размера запрашиваемой памяти:



Оранжевый – malloc

Зеленый – dlmalloc

Синий – calloc

Сравнивая malloc (оранжевую и зеленую линии), я точно могу сказать, что dlmalloc работает медленнее. Это можно объяснить тем, что помимо стандартных аспектов malloc в ней также присутствуют некоторые оптимизации, которые замедляют работу функции. В ее приоритете правильное использование памяти, а не быстрота выделения.

Также по графику видно, что на всем протяжении тестирования calloc работает медленнее, чем malloc, это характеризуется тем, что происходит дополнительная операция – инициализация памяти.

На графике заметен скачок во времени на уровне 3890 байт, что приблизительно к стандартному размеру кучи (4096 байт). После этой отметки продолжать тестирование не имеет смысла, поскольку для них будет применяться не malloc/dlmalloc/calloc, а mmap.

Помощь и консультации в выполнении работы оказывал **Шарифуллин И.А.**