LAB 6

CS 361: Systems Programming / Spring 2023

Description

In this lab session, you will inspect how malloc works internally.

Please read this document carefully and follow the instructions on the last section to complete this lab session. When you answered all the questions, please show your work to the TA.

Guide

1. Accept the invitation for Lab 6 on Github classroom: https://classroom.github.com/a/x7EsT5lR
2. Import the Github repository created to your machine using vscode, as explained in Assignment 0
3. Make sure that you can launch a terminal inside vscode via menus: Terminal > New Terminal
4. Read this guide and answer the questions as they appear. You should answer a total of 8 questions.

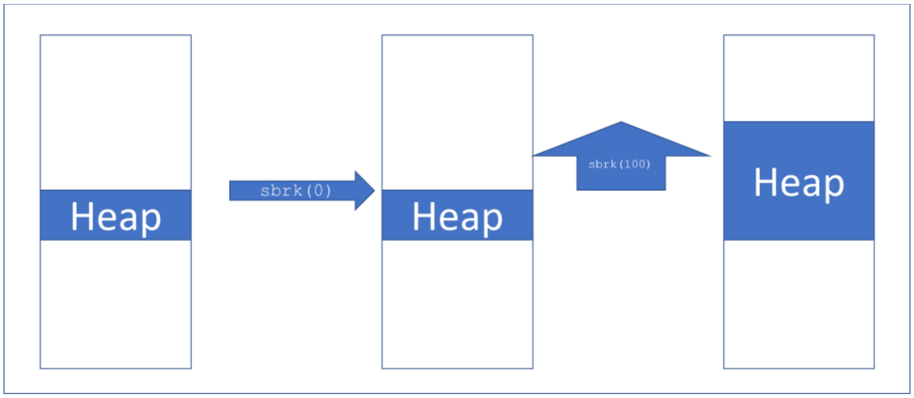
Running each program

In this lab, you will use the command `make` to compile all programs, just as in earlier labs.

However, you will use the provided script `run.sh` to run your programs. For instance, to run program `lab6-1` you will use the command `./run.sh ./lab6-1`

sbrk

Function `sbrk` is used internally by `malloc` to grow the size of the heap. It can also be used to discover the address that is the current limit of the heap.



In the example in the figure above, a call to sbrk with the argument zero simply returns the current program break (i.e., the upper limit of the heap). A call to sbrk with a non-zero argument (e.g., 100) grows the heap by that amount of bytes (e.g., the right-most heap is 100 bytes larger than the other two).

Question 1: Modify program lab6-1.c's for loop limits, and the amount of memory allocated per iteration, until you observe malloc grow the heap. By how many bytes did the heap grow?

Function hexdump

This lab has a handy function called hexdump:

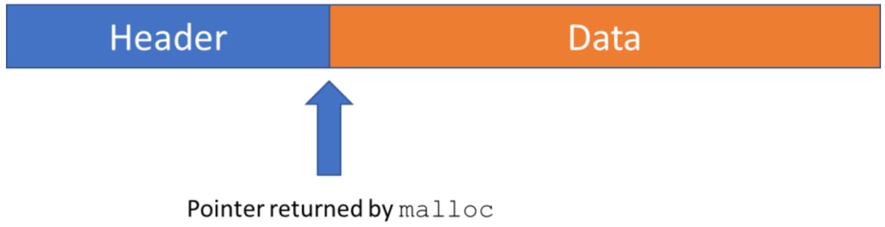
```
void hexdump(void * start, int len);
```

Function hexdump prints a dump of memory starting from pointer start and with len bytes.

Question 2: Modify program lab6-2.c to fill the arrays with the character 'a', 'b', and 'c'. Then call function hexdump to inspect the memory of all 3 arrays at once. Are the arrays allocated contiguously, or is there any memory separating each array?

`malloc`

Function `malloc` allocates memory on the heap, and returns a pointer to a chunk. Chunks have a header as follows:

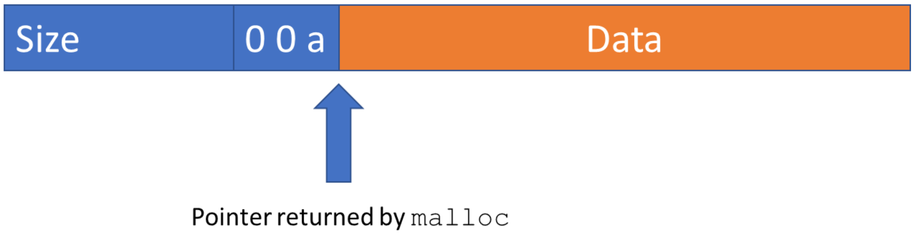


Question 3: Modify program `lab6-3.c` to fill the allocated memory with the character ‘a’, ‘b’, and ‘c’. Then use function `hexdump` to inspect the memory contents. How many bytes is each header between chunks?

Question 4: Use pointer arithmetic from Lab 5 with your answer to Question 3 to print the header of `mem1` with function `hexdump`.

Header

The internal structure of each chunk’s header is as follows:



The header starts with a 32bit integer, which denotes the size of the chunk. The least-significant 3 bits are flags:

- 3rd least significant bit (0): Always zero
- 2nd least significant bit (0): Always zero
- Least significant bit (a): Status of the **previous** chunk (1 means used, 0 means free)

Question 5: Modify `lab6-3.c` to print the size of the chunk. Sizes are never odd numbers, only even. Is the number you printed even or odd?

Question 6: Modify `lab6-3.c` to use a bitmask (Lab 5) to ignore the flags and print the correct size (should be an even number).

Question 7: Program `lab6-4.c` takes one optional argument:

- `./run.sh ./lab6-4 free` it will free `mem1`.
- `./run.sh ./lab6-4` it will not free `mem1`.
-

Print the header of `mem1` and `mem2` with and without the argument. What is different between runs?

Question 8: Modify program `lab6-4.c` such that it prints whether `mem1` is free or not. Confirm that it works by running lab6-4 as explained in Question 7. Did you use mem1's or mem2's header to determine the status of mem1?

Extra / Optional

Can you write a function that starts on one pointer and traverses the heap until the end, jumping over each chunk?

Grading

Show your UIC card to the TA when you enter the lab, or type your UIN on the chat when joining remotely. Stay in the session until you show your work, or until the TA announces that the lab is over.

- You have to remain present for the whole lab to get attendance, which you can then use to resubmit Assignment 3.
- You can leave early after showing your work to the TA (answers to all questions). In this case, you will get a 5% bonus in Assignment 3.