Hassaan Saeed (hs770)

Haneef Pervez (hp408)

Systems Programming, Section 07

Assignment 1

Data Structures:

- Linked List:
- memEntry Node:
    - int size
    - int isFree
    - mem Entry * next

Implementation:

To implement our program, we used a static array called myblock and a struct to hold our metadata called memEntry. The fields of the struct memEntry were an int field called *size* to keep track of the size, an int field called *isFree* to keep track of if the corresponding space was free or not, and pointer to the next memEntry in the array. We kept track of the first malloc by keeping a short, called *magic* in the beginning of the array. If the magic number was not present in the array, then it was the first malloc, and the first metadata will be added. If it is not the first malloc, then we add the magic number in the beginning of myblock, allocate the space and add a metadata after the allocated space that points to the rest of the array. We also check if the amount to be malloced is valid before allocating space by checking if the size to be allocated is nonnegative and can actually fit in the array. We implemented *myfree* by taking a pointer and going to the corresponding metadata and changing the *isFree* field to 1. We also check to make sure the pointer given is a valid pointer. To keep track of the workloads we ran them 100 times and stored the total times in a double array and then divided each number by 100 to find the mean.

Findings:

- *Work Load A: 17.66 milliseconds*
- *Work Load B: 46.71 milliseconds* *Slowest Test Case*
- *Work Load C: 0.22 milliseconds*
- *Work Load D: 9.44 milliseconds*
- *Work Load E: 0.10 milliseconds* *Fastest Test Case*
- *Work Load F: 0.27 milliseconds*