Readme

In this assignment, we were tasked with creating a malloc and free function. The inbuilt malloc function in C checks to see if there is enough space on the stack to allocate the requested space for the users use. Our code inherits some of the attributes of the inbuilt malloc however, our program, *mymalloc*, is given a 5000 byte array and is tasked with allocating memory efficiently. If it encounters a specific error, it will output a specific error message and then exit the program.

Mymalloc uses a doubly linked list to check if there is adequate space to be allocated. If so, it proceeds to allocate the memory. There is another check to see if the metadata has been created. If not, the metadata linked list is defined. After that, a single pointer traverses through the linked list to check for a free space. If so, the space will be allocated and the pointer will be set accordingly. If the space is not available throughout the linked list of metadata, then user will get an error message.

The inbuilt free method in C looks for free recieves a memory address from the user, in the form of a pointer, and then frees the memory that was malloc'ed to that pointer. It has several error checks to make sure that the method does not free any memory that was not allocated to that pointer. Myfree also traverses the double linked lists to accomplish these tasks.

Memgrind Outputs

The memgrind outputs are as following:

A: The program ran out of memory because the metadata was taking up space for each memory allocation of 1 byte.

B:

```
Workload A Runtime: 0.000000 Microseconds
Workload B Runtime: 100.410004 Microseconds
Workload C Runtime: 0.000000 Microseconds
Workload D Runtime: 0.000000 Microseconds
Workload E Runtime: 428342792146649088.000000 Microseconds
Workload F Runtime: 0.000000 Microseconds
```

C: The output has a segmentation fault because the program runs out of memory very quickly due to the size of the required the metadata for allocating the memory in the array.

D:

```
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
ERROR: cannot free pointer
```

E:

```
Error is calling mymalloc
```

F: Segmentation fault: The number of times it mallocs is 500*5 times. This the reason why there are segmentation faults because the processor runs out of space really fast.