## My malloc and my free:

usage: to use my malloc and free you must name the program you want to compile with it main.c and include the following 2 lines inside the header of your main.c:

```
#define malloc(x) mymalloc(x, __FILE__,__LINE__)
#define free(x) myfree(x,__FILE__,__LINE__)
```

## Implementation:

In my implementation I created 2 different static char arrays for small and big chunks of memory being allocated. If the size of the block is greater than 300, its stored in the large array and if its less than 300 it is stored in the small array in order to handle fragmentation

## Error Checking:

To check make sure your program does not free a pointer that was never allocated, my myfree function checks to see if the pointer exits in the memory, if it does not then it prints an error and returns. To check to make sure that the user is passing in a pointer to free that was allocated by my malloc function, It goes through all of the MemEnt structs and makes sure the pointer maches some of the structs. In order to make sure that a block that is being free'd has not already been free'd, my MemEnt structs hold a boolean value. If a user tries to free a chunk of memory that has already been free'd, my myfree function catches the mistake a prints and error and returns. In order to avoid saturation, I check that the amount of memory that the user is trying to allocate is less than the amount left in the static arrays and if it is not, it prints an error telling the user they ran out of memory.