Report of Thread-Safe Malloc

Name: Peiru Liu | NetID: pl202

1 Implementation of Lock Version

In my previous code, I used a linked list to connect all freed blocks together. In this way, it may cause race condition. So to prevent race condition happen, I use the code below:

```
pthread_mutex_lock(&lock);
pthread_mutex_unlock(&lock);
```

I put the bf_malloc function between this two lines, and it is called critical section. The code will be locked and it can only be used by one thread. Also, I put the bf_free function between two lines. Another thread cannot access the code until it is unlocked.

2 Implementation of Non-lock Version

I used the code below to guarantee that each thread has its own linked list:

```
__thread block free_head_unlock = NULL;
__thread block tail_unlock = NULL;
```

__thread keyword enable that the values of each thread do not interfere with each other and it has independent head and tail pointer. I also add lock before sbrk because it is not thread-safe and unlock it after using the sbrk. The code is below:

```
void * ts_malloc_nolock(size_t size) {
  void * n = bf_malloc(size, &free_head_unlock, &tail_unlock, 1);
  return n;
}
void ts_free_nolock(void * ptr) {
  bf_free(ptr, &free_head_unlock, &tail_unlock);
}
```

3 Results and Analysis

The result of implementation is as below:

Lock/Non-lock	Average Execution Time/s	Average Data Segment Size
Lock	0.16 s	43471584 bytes
Non-lock	0.14 s	44131520 bytes

From the table above, first, we can conclude that the execution time of nolock is little small than the lock. I think it is because that the lock operation not only lock the sbrk but also lock the malloc and free function while the unlock operation just lock the sbrk, so the unlock operation will run faster. Second, according to the table above, we can notice that the segment size of lock is smaller than the unlock operation. For unlock version, because each thread has its own local scope, the memory utilization efficiency decreases when free is called. So the segment size is larger than the lock version.