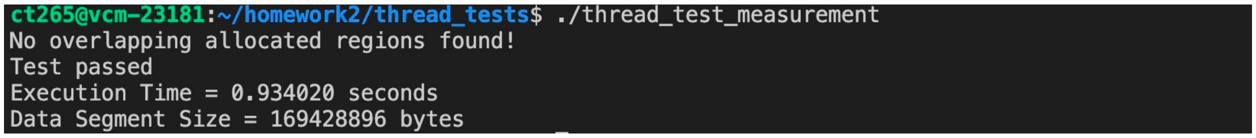
Report of Homework 2

Name: Chenglong Tang, NetID: ct265

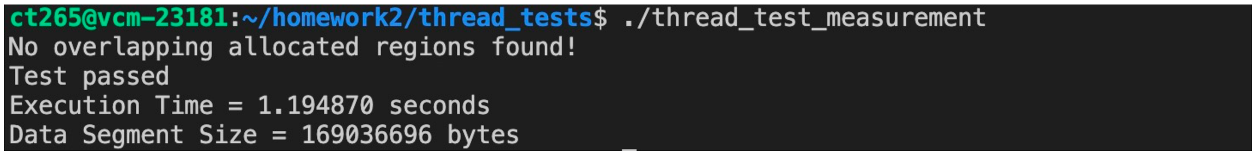
- Target: I am asked to implement 2 different thread-safe of versions of the malloc() and free() functions. Both of the thread-saft malloc and free functions are using **Best Fit** allocation policy.
- Implementation Description: For lock version, the **critical section** is the whole malloc() and the whole free() functions. So I just add one clock in the beginning and end of these two functions. For unlock version, I marked all the shared variables with label "__thread" so that each thread creates different memory for each variable so that **race conditions** will never happened.
- Performance Result Presentation: Below is the test results running in 16 threads.

For lock version, the result is:



```
ct265@vcm-23181:~/homework2/thread_tests$ ./thread_test_measurement
No overlapping allocated regions found!
Test passed
Execution Time = 0.934020 seconds
Data Segment Size = 169428896 bytes
```

For no-lock version, the result is:



```
ct265@vcm-23181:~/homework2/thread_tests$ ./thread_test_measurement
No overlapping allocated regions found!
Test passed
Execution Time = 1.194870 seconds
Data Segment Size = 169036696 bytes
```

- Result Analysis:

For lock version:

The **Data Segment Size** of **lock version** is similar(very close) to that of **no-lock version** because they call sbrk() in quite a similar way. For **Execution time**, **lock version** takes less time than **no-lock version**. Since the virtual machine only has 2 core and we have 16 threads, the thread overhead is pretty high for no-lock version, this contributing to more times spending.

- Recommendation: It's hard to tell which one is better. When we are in an environment which could support more cores comparative to the number of threads, it's better to apply no-lock version. Otherwise, we choose lock version.