Project 7: Contiguous Memory Allocation

Name: 韩冰

Number: 516030910523

This project will involve managing a contiguous region of memory of size MAX where addresses may range from 0 ... MAX - 1. Your program must respond to four different requests:

1. Request for a contiguous block of memory
2. Release of a contiguous block of memory
3. Compact unused holes of memory into one single block
4. Report the regions of free and allocated memory

Your program will be passed the initial amount of memory at startup. For example, the following initializes the program with 1 MB (1,048,576 bytes) of memory: ./allocator 1048576

The Memory Allocation

Program file tree structure.

```
.
├── allocate.c   main function
└── report.pdf  readme file
```

Solution:

1. Initial the array of the all jobs, and initial a job struct which includes the job's start and ends position.
2. Design the main function to interact with user. Read the instructions and print the result.
3. After reading the instructions, if request. then iterate the array to find the interval and put the job into array.
4. The detail can see the codes.

Screenshot

RQ for some memory, then show the state

```
oskernel@ubuntu:~/allocate$ ./a.out 10240
allocator>@RQ P0 100 F
request for memory
allocator>@RQ P1 200 F
request for memory
allocator>@RQ P3 100 F
request for memory
allocator>@RQ P4 300 F
request for memory
allocator>@RQ P5 100 F
request for memory
allocator>@STAT
Addresses [0:100] Process P0
Addresses [100:300] Process P1
Addresses [300:400] Process P3
Addresses [400:700] Process P4
Addresses [700:800] Process P5
allocator>@
```

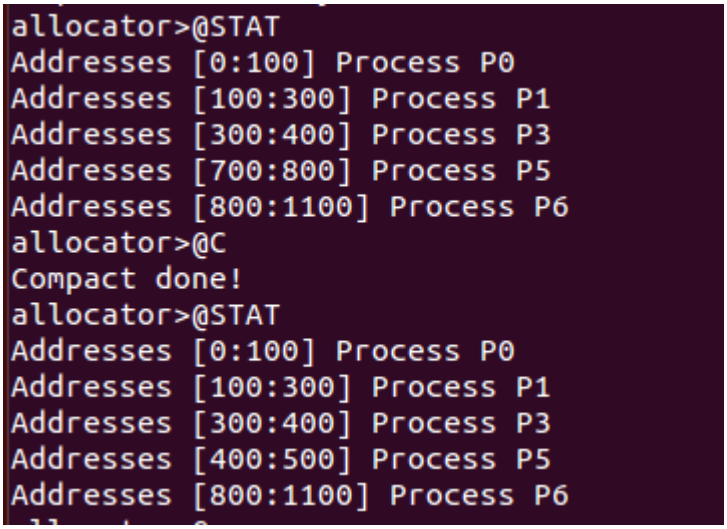
RL for some jobs, then show the state

```
allocator>@STAT
Addresses [0:100] Process P0
Addresses [100:300] Process P1
Addresses [300:400] Process P3
Addresses [400:700] Process P4
Addresses [700:800] Process P5
allocator>@RL P1
allocator>@RL P4
allocator>@STAT
Addresses [0:100] Process P0
Addresses [300:400] Process P3
Addresses [700:800] Process P5
allocator>@
```

RQ for other allocation

```
allocator>@STAT
Addresses [0:100] Process P0
Addresses [300:400] Process P3
Addresses [700:800] Process P5
allocator>@RQ P1 200 B
request for memory
allocator>@RQ P6 300 W
request for memory
allocator>@STAT
Addresses [0:100] Process P0
Addresses [100:300] Process P1
Addresses [300:400] Process P3
Addresses [700:800] Process P5
Addresses [800:1100] Process P6
```

Compact the allocation


allocator>@STAT
Addresses [0:100] Process P0
Addresses [100:300] Process P1
Addresses [300:400] Process P3
Addresses [700:800] Process P5
Addresses [800:1100] Process P6
allocator>@C
Compact done!
allocator>@STAT
Addresses [0:100] Process P0
Addresses [100:300] Process P1
Addresses [300:400] Process P3
Addresses [400:500] Process P5
Addresses [800:1100] Process P6