Joe Uzdzinski
CIS3207 - Project 1 - Giorgio's Discrete Event Simulator
TESTING

The first part of the project I focused on were the queues. I used a menu for debugging. This is a demonstration of the priority queue being tested:

```
Joes-MacBook-Pro:Discrete-Event-Simulator-UZDZINSKI joeyu$ ./priority
1. enq
2. dq
3. quit
Enter choice: 1
Type of event? 1
Time of event? 5
type: 1 time: 5 --> NULL
1. enq
2. dq
3. quit
Enter choice: 1
Type of event? 2
Time of event? 15
type: 1 time: 5 --> type: 2 time: 15 --> NULL
1. enq
2. dq
3. quit
Enter choice: 1
Type of event? 3
Time of event? 2
type: 3 time: 2 --> type: 1 time: 5 --> type: 2 time: 15 --> NULL
1. enq
2. dq
3. quit
Enter choice: 2
type: 1 time: 5 --> type: 2 time: 15 --> NULL
Removed: 2 3
1. enq
2. dq
3. quit
Enter choice: 3
QUITTING
```

The fifo queue was tested in the same manner.

The rest of the project was tested after and during each handler was implemented. This was done with a series of print statements that displayed the state of the simulation with each iteration:

```
printf("\nCurrent Time: %d\n", currentTime);

printf("-------------------------\n");

printf("Next Processed Event:\n\n");
printf("job %d ", nextEvent -> jobNo);
printEventType(nextEvent -> type);
printf(" %d \n", nextEvent -> time);

printf("-------------------------\n");

printf("CPU status: %d\n", CPU -> status);
printf("DISK1 status: %d\n", DISK1 -> status);
printf("DISK2 status: %d\n", DISK2 -> status);

printf("-------------------------\n");

printf("Event Q\n\n");
printq(&EVENT_Qptr);

printf("-------------------------\n");

printf("CPU Q\n\n");
printq(&CPU_Qptr);

printf("-------------------------\n");

printf("DISK1 Q\n\n");
printq(&DISK1_Qptr);

printf("-------------------------\n");

printf("DISK2 Q\n\n");
printq(&DISK2_Qptr);

printf("_________________________________________________\n");
```

```
Queue is empty
-------------------------
DISK2 Q

Queue is empty
_________________________________________________

Current Time: 80
-------------------------
Next Processed Event:

job 2 DISK1_ARRIVAL 80
-------------------------
CPU status: 1
DISK1 status: 0
DISK2 status: 0
-------------------------
Event Q

job 4 CPU_BEGIN 80 -->
job 7 PROCESS_ARRIVAL 86 -->
job -1 SIMULATION_FINISH 100 -->
NULL
-------------------------
CPU Q

job 1 DISK1_FINISH 61 -->
job 5 PROCESS_ARRIVAL 63 -->
job 3 DISK1_FINISH 73 -->
job 6 PROCESS_ARRIVAL 75 -->
NULL
-------------------------
DISK1 Q

Queue is empty
-------------------------
DISK2 Q

Queue is empty
_________________________________________________

Current Time: 80
-------------------------
Next Processed Event:

job 4 CPU_BEGIN 80
-------------------------
```

In order to print the events for testing, this function was created to print the event types:

```
//used for debugging, not implemented in actual simulation
void printEventType (eventType type) {
    switch (type) {
            case PROCESS_ARRIVAL: {
                printf("PROCESS_ARRIVAL");
                break;
            }
            case CPU_BEGIN: {
                printf("CPU_BEGIN");
                break;
            }
            case CPU_FINISH: {
                printf("CPU_FINISH");
                break;
            }
            case PROCESS_EXIT: {
                printf("PROCESS_EXIT");
                break;
            }
            case DISK1_ARRIVAL: {
                printf("DISK1_ARRIVAL");
                break;
            }
            case DISK1_BEGIN: {
                printf("DISK1_BEGIN");
                break;
            }
            case DISK2_ARRIVAL: {
                printf("DISK2_ARRIVAL");
                break;
            }
            case DISK2_BEGIN: {
                printf("DISK2_BEGIN");
                break;
            }
            case DISK1_FINISH: {
                printf("DISK1_FINISH");
```