Description

The purpose of this project will be to reinforce you understanding of paging by building memory simulator.

Your simulator will:

1. Accept at least the following parameters:
    a. The size of the computer memory to simulate
    b. The size of a page in the computer's memory
        i. We won't be worrying about memory used by the kernel in the simulator, all memory will be for user programs.
        ii. The memory size must be an even multiple of the page size
    c. The number of jobs to run
    d. The minimum and maximum run times for each job
    e. The minimum and maximum memory needed for each job
2. Jobs will be scheduled using a round robin (RR) algorithm, with the time slice set to one time step.
3. The jobs will be generated at the start of the simulator and form a queue:
    a. The memory for the job will be randomly selected between the minimum and maximum memory sized specified on the command line.
    b. The run time for the job will be randomly selected between the minimum and maximum times (number of time slices) specified on the command line.
4. Loop, stepping through time and, at each time step:
    a. See if there are sufficient pages available to schedule the next job in the queue. If pages are available, assign the pages needed to the job and add the job to the process list for the round RR scheduler. Repeat until there aren't enough pages available to schedule the next job in the queue.
    b. Select the next available job in the RR process list to 'run' for the time step.
    c. If the job has completed (used all it's time), mark the jobs pages free so that they are available in the next time step.
5. When all jobs have been completed, end the program

The simulator must:

1. Run from the Linux command line (no graphical interfaces).
2. Be repeatable: you need to be able to specify a random seed (environment variables work well for this).
3. Output, at a minimum:
    a. The parameters the program is using.
    b. The contents of the job queue (after step 3 above).
    c. An indicator when a job is started and when a job is finished.
    d. An indication of the contents of the page table at each time step. In the example, free pages are indicated by dots (.) and pages in use by the job number owning the page (note this example only works for 10 jobs, you will need to support more than 10 jobs).
    e. At the completion of the simulation

    A sample of what the output MIGHT (but isn't required to) look like is provided on the last page.

The simulator may be written in any programming language you desire.

Due Date and Deliverables

The project is due by midnight, Tuesday, March 3rd[th]. Late work will be accepted according to the schedule shown in the syllabus.

By Thursday, February 13[rd], I will need to know the names of students that are working together (DM in slack preferably). The project is to be worked in groups of two students, not one, not three. If there are an odd number of students participating, it will be at the instructor's discretion to allow either one individual submission or one group of three. You are free to use the same partner as for the first project or choose a different person.

I will be expecting at a minimum a C source file and a `Makefile` to build the project. My preference would be a link to a GitHub (Bitlocker, GitLab, etc.) repository that I can clone to see the code but I will accept the aforementioned files submitted into the Oaks dropbox.

Rubric

The project will be worth 50 points in total allocated as follows:

1. 35 Points – Operation

    - (11 points) Does the program build correctly?
    - (12 points) Does the program run and produce useful, readable, output?
    - (12 points) Does the simulation work correctly

2. 15 Points – Programming Style

    - (5 points) Is the code well commented and readable?
    - (5 points) Are all needed files/libraries provided?
    - (5 points) Does the coding demonstrate an understanding of the features of the language used?

Sample Execution and Output

```
set RANDOM_SEED=13
./mysim 48000 1000 3 2 15 5000 25000

Simulator Parameters:
  Memory Size: 100000
  Page Size: 1000
  Random Seed: 13
  Number of jobs: 3
  Runtime (min-max) timesteps: 2-15
  Memory (min-max): 5000-25000

Job Queue:
   Job #   Runtime   Memory
       1         7    12000
       2         4    22000
       3        14     8000

Simulator Starting:

Time Step 1:
   Job 1 Starting
   Job 2 Starting
   Job 3 Starting
   Job 1 Running
   Page table:
       1111 1111 1111 2222
       2222 2222 2222 2222
       2233 3333 33.. ....
       .... .... .... ....

Time Step 2:
   Job 2 Running
   Page table:
       1111 1111 1111 2222
       2222 2222 2222 2222
       2233 3333 33.. ....
       .... .... .... ....
```

and so on...

```
Time Step 11:
   Job 2 Running
   Job 2 Completed
   Page table:
       1111 1111 1111 ....
       .... .... .... ....
       ..33 3333 33.. ....
       .... .... .... ....
```

and so on until all jobs are completed...

```
Job Information:

   Job #   Start Time   End Time
       1            1         21
       2            1         17
       3            1         25
```