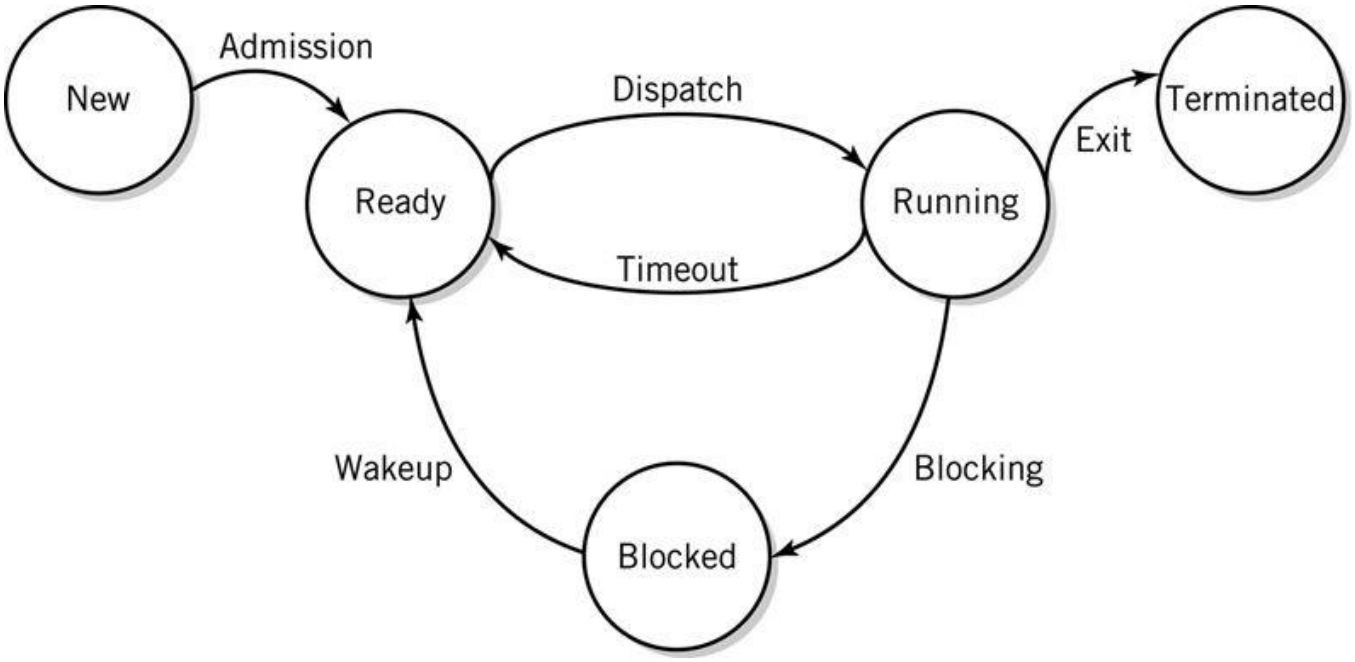
MSU CSC 344 Fall 2016
Assignment 4 – Simulate process queues
Given: Mon. Nov. 14, version 1
Due: Wed. Dec. 7 11:59pm

You will simulate the process queues in an OS, following the diagram below.



The data that MAY be used in this simulation is below. Not all this data will be used in all phases of the assignment.

```
int Priority = 0;  // Priority of the process, NEVER CHANGED BY
POLICY.
int SpawnTime = 0;  // Time the process is spawned, never updated.
int RunLength = 0;  // CPU length of process, never updated.
int BlockingModel = 0;  // Model for determining Blocking of a
process, never updated.

int NextEventTime = 0; // Time of the next event of this process,
updated as needed.
int ReadyLength = 0;  // Extent of time this process spent in Ready
state, updated as needed.
int BlockedLength = 0;   // Extent of time this process spent in
Blocked state, updated as needed.
int ExitTime = 0;  // Time this process completed, updated as
needed.
```

***(Important:** The "time slice" in the Running state is initially 5 units. You may change that time slice later.)*

New is a priority queue based on process initialization time. The process moves to the Ready queue when that time arrives.

Ready is a priority queue based on both process priority and time. You may implement a priority queue based on both process priority and time using an array (indexed by process priority) of priority queues (each one based on time). *Note: initially, implement only ONE priority queue that ignores the process priority and is only based on time.*

Running is one, single process.

Blocked is a priority queue based on only process blocking time.

Terminated is a data structure of your choice that contains a collection of processes. You may need to go through those processes several times to compute averages, etc.

The key operation of the process simulation is a loop that updates TIME.

```
DO // Move processes from state to state
      if (next process event is a Running process)
            if that process is finished
                  move process to Terminated
            else if that process is Blocked
                  determine the amount of blocked
                  push process into BlockedPQ
            else
                  push process into ReadyPQ
            update Time
      else if (next process event is a New process)
            pop process off NewPQ
            push process into ReadyPQ
            update Time
      else if (next process event is a Blocked process)
            etc.
      else // next process event must be a Run process
            etc.
While (time < TotalSimulationTime)
Report the simulation outcome
```

Essential for this assignment:

a. Put all your code in one file named Assn4.cpp
b. Read from a data file the following:
    i. Amount of time the simulation is to continue. This is an integer unit.
    ii. The four essential characteristics of each process, `Priority`, `SpawnTime`, `RunLength`, and `BlockingModel`, into a class object. The remainder of the data file will consist of some number of repetitions of those four integer values.
        `Priority` is an integer value 1..5, where 1 means the highest (first) priority, and 5 means the lowest (last) priority.
        `BlockingModel` is an integer value 1..5, where each number designates a different model for blocking time. For the initial implementation, use only NO BLOCKING. Other blocking models are provided below.
c. Implement the simulation of those processes up to the limit of time specified.
d. Report the following data on processes completed in the limit of time specified:

i. Number of processes completed
ii. Average time spent in the Ready queue

To obtain full credit for this assignment, do any one of the following:

a. Implement blocking models. Create input data files that demonstrate these blocking models, and create a Word document named Assn4_Block that describes the simulation outcome differences when a process does not block and when it does. **(Two decent paragraphs.)**
BlockingModel value 1: Process does not block
BlockingModel value 2: Process has a 5% chance of blocking for 5 time units
BlockingModel value 3: Process has a 20% chance of blocking for 10 time units
BlockingModel value 4: Process does not block
BlockingModel value 5: Process does not block

b. Create a "live action" graphic display of the simulation as it operates. Read the BearPlot documentation provided in the trace Assignment folder.

c. Change the value of "time slice" – the amount of time a process spends in the Running state, and re-run the simulation. Create a Word document named Assn4_Time that describes the effect that the time slice has on the throughput of the simulation. Explain why we can't simply increase time slice to an arbitrarily large value to complete processes quicker. **(Two decent paragraphs.)**

TURN IN TO TRACE…344:

- This edited word document
- BMP_in_hex.txt