

```
struct Process {
```

```
    int priority;
```

```
};
```

```
vector<Process> processTable;
```

```
class readyQueueComparator {
```

```
public: static void setProcPointer(Process *p) {  
    proc = p;  
    return; }  
bool operator () (Process *p1,
```

```
    Process *p2) {
```

```
    return p1->priority > p2->priority;
```

```
    }  
    }  
};
```

```
{
```

```
    Process p; Process *ptr;
```

```
    p.priority = 42;
```

```
    processTable.push_back(p);
```

```
    int pid = processTable.size() - 1;
```

```
    priority_queue<Process *, vector<Process * >
```

```
        , readyQueueComparator > readyQueue;
```

```
    readyQueue.push(ptr);
```

```
    ptr = &processTable  
        [pid];
```

```
double time, increment;  
unsigned char charIn;  
ifstream fin;
```

```
⋮
```

```
fin.read((char*) &charIn, 1);
```

```
increment = charIn / 10.0;
```

```
Event e;
```

```
e.time = time + increment;
```

```
e.type = Event::ARRIVAL;
```

```
eventQueue.push(e);
```

```
fin.read((char*) &increment,  
         sizeof(double));
```

