Project 1 – CPU Scheduler

Some Notes from Project 0

- Grading is in the works – should be done by Saturday night ☺
- Don’t do everything in main()
- Start early
- Feel free to develop on whatever OS you choose
  - Running on Linux isn’t as essential on this project
  - (But we are still testing on it, so make sure it runs)

A brief interlude for the writeup…

- Note: Some stuff may not be clear yet – will be discussed in class in coming weeks

Simulator in Action!

Deliverable 1 – Feb 20th, 11:59PM

- ./simulator ~~[flags]~~ simulation_file.txt
- Create Process, Thread, Event, & Burst Classes or Structs
- Read the simulation files into appropriate data structures (see above), sets up the priority queue for events
- Iterate over Event Queue and output THREAD_ARRIVED events in the correct format.
  - This format is the verbose format. Feel free to implement this with the flag or not. Flags are not required for this deliverable.

Class/Struct Overview

- Set this up in a reasonable way, the next steps depend on this
  - You probably want one class or struct per header file
  - May need to use forward declaration depending on your setup
- Use pointers – ex: Event* event = new Event(Type, time, …)
  - Don’t forget to clean up – ex: delete event;
- Example on whiteboard
  - Update: see next page

Whiteboard Example

- This represents a general structure you may find helpful implementing deliverable 1
  - For Process.h, I've given you what I think is necessary.
  - This represents a guideline - feel free to implement it in any way you choose.

Process.h:

```
class/struct Process{
    enum Type{...};
    int pid;
    Type ptype;
    vector<Thread*> threads;
    Process(...); //constructor
};
```

Event.h

```
class/struct Event{
    //fill in your variables here
};
```

Thread.h:

```
class/struct Thread{
    //fill in your variables here
};
```

Burst.h:

```
class/struct Burst{
    //fill in your variables here
};
```

The Event Priority Queue

- template <class T, class Container = vector<T>, class Compare = less<typename Container::value_type> > class priority_queue;
  - http://www.cplusplus.com/reference/queue/priority_queue/
- Gives us access to elements in sorted order
- What might we want to sort by?

Deliverable 2 – Mar 6th, 11:59PM

- ./simulator [flags] simulation_file.txt
- FCFS algorithm implemented
- Program metrics are displayed

Deliverable 3 – Mar 20th, 11:59PM

- --algorithm flag available
- All algorithms implemented
  - Custom algorithm implemented & described in README

Class/Struct/Makefile Example

Activities Sublime Text Fri 00:56

~/abc.cpp - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

makefile

```
all: COMPILE

COMPILE:
    g++ -Wall -o simulator abc.cpp

run: COMPILE
    ./simulator simulation_file.txt
```

abc.cpp

```
#include <iostream>

struct Hello{
    //public by default
    std::string greeting = "Hello from the Hello struct!";
};
class Hi{
//not public by default
public:
    std::string greeting = "Hi from the Hi class";
};


int main(int argc, char* argv[]){
    std::cout<<"Hello, World"<<std::endl;

    Hello* hello = new Hello();
    std::cout<<hello->greeting<<std::endl;



    Hi* hi = new Hi();
    std::cout<<hi->greeting<<std::endl;


    delete hi; //always remember to clean up
    delete hello; //always remember to clean up

    return EXIT_SUCCESS;
}
```

Line 10, Column 38; Saved ~/abc.cpp (UTF-8) Tab Size: 4 C++

Additional Suggestions

- You may want to use boost to help with output formatting
  - https://www.boost.org/doc/libs/1_66_0/libs/format/doc/format.html
  - …or not
- You may want to use <getopt.h> for flag handling
  - …or not
- Feel free to include additional libraries as needed, just make sure:
  - You update your makefile accordingly
  - The library is available on Alamode
- Feel free to use higher C++ standards, just make sure:
  - You update your makefile accordingly
  - The version is available on Alamode

Questions?