UDACITY

‹ Return to "C++" in the classroom

Process Monitor

REVIEW

CODE REVIEW 16

HISTORY

▸ src/linux_parser.cpp 9

▾ src/system.cpp 2

```
#include <unistd.h>
#include <cstddef>
#include <set>
#include <string>
#include <vector>
#include <algorithm>

#include "process.h"
#include "processor.h"
#include "system.h"
#include "linux_parser.h"
```

SUGGESTION

You can just use the line using `namespace std;`
It will take care of all the std:: that you use as a prefix of many STLs.

Important

But take care of the following issue addressed here.

https://stackoverflow.com/questions/11271889/global-variable-count-ambiguous

So always take care of this while using

```
using namespace std;
```

```

// Constructor
System::System() {
   cpu_ = Processor();
}

// Return the system's CPU
Processor& System::Cpu() {
   return cpu_;
}

// Return a container composed of the system's processes
std::vector<Process>& System::Processes() {
   processes_.clear();
   std::vector<int> pids = LinuxParser::Pids();
   for (std::size_t i = 0; i < pids.size(); i++) {
      Process p(pids[i]);
      processes_.push_back(p);
```

SUGGESTION

Try using `emplace_back` instead of `push_back`, it is much more efficient than `push_back`.
push_back constructs a temporary object which then will need to get moved into the vector v whereas em
In short, it is a good habit if you are using `emplace_back` instead of `push_back`

```
   }
   //sort vector using overloaded "less than" comparison operator
   std::sort(processes_.begin(), processes_.end());
   return processes_;
}

// Return the system's kernel identifier (string)
std::string System::Kernel() {
   return LinuxParser::Kernel();
}

// Return the system's memory utilization
float System::MemoryUtilization() {
   return LinuxParser::MemoryUtilization();
}

// Return the operating system name
std::string System::OperatingSystem() {
   return LinuxParser::OperatingSystem();
}

// Return the number of processes actively running on the system
int System::RunningProcesses() {
   return LinuxParser::RunningProcesses();
}

// Return the total number of processes on the system
```

```
int System::TotalProcesses() {
   return LinuxParser::TotalProcesses();
}

// Return the number of seconds since the system started running
long int System::UpTime() {
   return LinuxParser::UpTime();
}

```

▸ src/process.cpp 2

▸ src/processor.cpp 1

▸ src/format.cpp 1

▸ include/ncurses_display.h 1

▸ src/ncurses_display.cpp

▸ src/main.cpp

▸ include/system.h

▸ include/processor.h

▸ include/process.h

▸ include/linux_parser.h

▸ include/format.h

▸ README.md

▸ Makefile

▸ CMakeLists.txt

RETURN TO PATH