Process Monitor

▸ src/linux_parser.cpp 14

▸ src/system.cpp 2

▸ src/processor.cpp 2

▾ src/process.cpp 2

```cpp
#include "process.h"

#include <unistd.h>

#include <cctype>
#include <fstream>
#include <regex>
#include <sstream>
#include <string>
#include <vector>

using std::string;
using std::to_string;
using std::vector;

Process::Process(int pid) : pid_(pid) {}

// TODO: Return this process's ID
int Process::Pid() { return pid_; }

// DONE: Return this process's CPU utilization
float Process::CpuUtilization() {
  long tot_time = LinuxParser::ActiveJiffies(pid_);
  float seconds = LinuxParser::UpTime() - UpTime();

  return (seconds != 0) ? (cpu_ = tot_time / sysconf(_SC_CLK_TCK)) / seconds
                        : cpu_ = 0;
}

// TODO: Return the command that generated this process
string Process::Command() { return LinuxParser::Command(pid_); }
```

SUGGESTION

Here you can make sure that you do not send command more than of 50 characters (or 40 characters - It is your choice).
You can append `...` in the end if the command exceeds more than 40-50 characters and all.

```cpp

// TODO: Return this process's memory utilization
string Process::Ram() { return LinuxParser::Ram(pid_); }

// TODO: Return the user (name) that generated this process
string Process::User() { return LinuxParser::User(pid_); }

// TODO: Return the age of this process (in seconds)
long int Process::UpTime() { return LinuxParser::UpTime(pid_); }

// TODO: Overload the "less than" comparison operator for Process objects
// REMOVE: [[maybe_unused]] once you define the function
bool Process::operator<(Process &a) {
  return CpuUtilization() < a.CpuUtilization();
}
```

AWESOME

It is so nice to see that you have overloaded less than comparison operator for process objects.

Similarly, you can do so for any operator using the correct logic and application

Note : Following Operators cannot be overloaded:

1. `.` (Member Access or Dot operator)

2. `?:` (Ternary or Conditional Operator)
3. `::` (Scope Resolution Operator)
4. `.*` (Pointer-to-member Operator)
5. `sizeof` (Object size Operator)
6. `typeid` (Object type Operator)

▸ **src/format.cpp** 1

▸ **Makefile** 1

▸ **include/ncurses_display.h** 1

▸ **include/linux_parser.h** 1

▸ **src/ncurses_display.cpp**

▸ **src/main.cpp**

▸ **README.md**

▸ **include/system.h**

▸ **include/processor.h**

▸ **include/process.h**

▸ **include/format.h**

▸ **CMakeLists.txt**

Learn the best practices for revising and resubmitting your project.

RETURN TO PATH