Name: <u>ERIC KAHINDI KAZUNGU</u>
Reg Number: <u>SCT211-0028/2021</u>
Course: <u>B.Sc COMPUTER SCIENCE</u>
Unit Code: <u>ICS 2305</u>
Unit Name: <u>SYSTEMS PROGRAMMING</u>

<u>ASSIGNMENT 1</u>

a. **Write a C program that prints the process ID, priorities, and parent ID of all programs currently in the RAM.**
   - For this to work we need to enlist the help of a special Linux directory called the proc directory. It has special files that represent the current state of the Kernal allowing us to peer into the system to see what is going on.
   - These include directories for processes named by their process IDs. The Stats file of these directories is what we need to retrieve their process ID, priorities, and parent ID

b. **Create a Shell Script called "NYONGA", while the Shell is open, Write a program in C that kills the open shell Script.**
   The program source code is in the attached folder.

c. **Write a C program that uses the fork () API to create a child process.**
   Fork() is used to create a child process by splitting a process into two parallel processes. One is the child process and the other is the Parent process

d. **Describe in prose how waitpid() and wait() works.**
   - The first function, waitpid(), is a function that is used to wait for a specific child process especially when the parent process has multiple child processes. It takes in parameters like the PID of the child to wait for, a pointer where the exit status of a terminated child process will be stored, and lastly, an Options parameter to control the behavior.
   - The second function, wait(), is a function that's less flexible but simpler than the waitpid() function. It doesn't allow us to specify the PID of the child process to be waited for. It is useful in situations where the order of child process completion is not a priority. However, like the waitpid() function it allows for a pointer parameter that stores the exit status of the process

e. **In Linux tools such as sysstat , sar and Nmon can be used to monitor CPU performance (Try them)**
   In debian for instance to install sysstat you will run in terminal "sudo apt-get install sysstat").
   We want to graph various CPU usage. We want to capture the CPU usage for few minutes and graph it.

Write a C program that incorporates a Graphical User Interface (GUI) that graphs CPU usage in real-time
Program source code in the attached folder

f. **Describe users defined signals with two working examples (Do not copy internet examples ..Create your own!**
These are unique Signals in Linux that allow us to build our own special functionality to a Signl in the terminal

g. **Write a Program that creates a text file called JUJU in drive C and create a signal that deletes the file 5 seconds after the creation and reports on prompt ..Hint : Knowledge of SIGALRM will be handy**
Program source code in the attached folder