Operating Systems

21/09/2016

1. Show **the process generation tree and the output lines** produced by the execution of this program executed with command argument equal to **1**.

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <unistd.h>**

**int main (int argc, char \*argv[]) {**

**int i, n;**

**char str[50];**

**n = atoi (argv[1]);**

**printf ("run with n=%d\n", n); fflush (stdout);**

**for (i=0; i<n; i++) {**

**if (fork () > 0) {**

**sprintf (str, "%d", n-1);**

**execlp (argv[0], argv[0], str, NULL);**

**} else {**

**sprintf (str, "%s %d", argv[0], n-1);**

**system (str);**

**}**

**}**

**}**

1. Write a **bash** script that runs the program of exercise **n. 3** to tests its correctness.
2. Write a **program** that is able to handle signals **SIGUSR1,** **SIGUSR2** and **SIGALRM** as follows:

* Every 2 seconds controls the sequence of signals received, and
* outputs an **error** message if it received signals **SIGUSR1** followed by signal **SIGUSR2** or viceversa,
* outputs a **success** message if **two** successive **SIGUSR1** (or **SIGUSR2**) signals,
* **terminates** if it has received at least **three** successive **SIGUSR1** (or **SIGUSR2**) signals.

1. Write the pseudo-code for **N producers** and **1 consumer** processes that communicates through a common **buffer of unlimited size**.

**Exam questions 5 and 6 on the back side**

1. Write a **bash** script that takes a filename as argument. The file contains a list of directory names, one per line. Suppose that each directory has not sub-directories.

The script must execute the program **tester** on each file of the given directory list, having extension **.test**. Program **tester** outputs a line with format **<result>**

Where **result** is either **PASS** or **FAIL**, and its execution time, as an integer value. (e.g., **PASS** **108**). Recall that command **date +%s** returns a time as an integer.

The script must save all **.test** files that passed or failed the test in file **pass.txt** or **fail.txt**, respectively, with format **<filename time>**

Finally, the script must output the **total** **number** of PASSED files and their **total** execution time.

1. A Web Server keeps an access log file with this format:

**178.1.192.33 goodguy [10/07/2015:13:55:36] GET index.html 200**

**34.52.1.33 badguy [10/07/2015:14:04:10] POST /services/ask 200**

**34.52.1.37 badguy [10/07/2015:14:04:10] POST /intranet/login 401**

**34.52.177.48 okguy [10/07/2015:14:32:00] POST /intranet/login 401**

**178.1.192.41 nastyguy [10/07/2015:18:29:01] POST /intranet/login 401**

**178.1.192.42 nastyguy [10/07/2015:18:56:01] POST /intranet/login 200**

**178.1.192.32 goodguy [11/07/2015:15:21:43] POST /intranet/login 200**

**123.154.48.1 worstguy [11/07/2015:00:21:32] GET /services/list 200**

Every file line indicates: the host IP address issuing the access request, the username of the user issuing the request, the date and time of the request, the request type (**GET** or **POST**), the name of the resource, and the server answer (**200** for “access granted”, **401** for “access denied”).

Write an **awk** script that takes as its argument a log file and read a file **access.check**, which contains in its first line a date, and in the other lines a list of usernames. Example:

**10/07/2015**

**badguy**

**nastyguy**

**worstguy**

The script must output the list of the IP addresses from which one of the users of the given list has performed a **POST** access request on the given date, and the access has been granted (**200**).

The script must not output duplicate line information.

For the example files, the output is:

**34.52.1.33**

**178.1.192.42**