National University of Computer and Emerging Sc...

Operating Systems (CS2006)

Date: February 27 2024

Course Instructor(s)

Mr. Razi Uddin

Mr. Mubashar Hussain

Ms. Rubab Anam

Ms. Namra Absar

Sessional-I Exam

Total Time: 1 Hours

Total Marks: 35

Total Questions: 05

Pages: 07

Semester: SP-2024

Campus: Lahore

Dept: Computer Science

Solution

Student Name _____ Roll No _____ Section _____ Student Signature _____

Vetted by _____ Vetter Signature _____

***CLO #: 2* Implement solutions employing concepts of Processes and Threads**

Q1: Imagine a scenario where a parent process interacts with a child process to create a guessing game. Initially, the parent process prompts the user to input an integer number. This number is then transmitted to the child process through a pipe. Subsequently, the child process will compare that number with an already stored secrete number. If the number match, then the child process communicates this information back to the parent process through the pipe. In the case of mismatch, the child process will also inform parent about failure and then the parent process prompts the user again to input a number and transmits it to the child process. This iterative process continues until the child process either successfully guesses the number or exhausts the maximum limit of 5 attempts. Given the skeleton code below you are required to complete the missing portion of the code. **[Marks: 15]**

```
int main() {
int number, tries = 0;
//write your code here
```

int fd[2];
int out[2];

error handling of both

```
while (tries < 5)
{   pid = fork();
    if (pid == -1)
        "error.
}

else if( pid == 0 ){
// Child process
int targetNumber;
int guess=10; //Secrete Number
// write your code here
        close (fd[1]);
        read (fd[0], &targetNumber, sizeof(int));
        close (fd[0]);
        int flag = 0;
if (guess == targetNumber) {
// Notify parent about success (write your code here)
        flag = 1
        write (out[1], &flag, sizeof(int));
        exit(0);
} else {
// Notify parent about the failed attempt (write your code here)
        write (out[1], &flag, sizeof(int));
}
```

```
//write your code here

return 0;



} //Child Process End

else {
// Parent process

 cout << "Enter a number for the child to guess (or enter 'quit' to exit): ";
 string input;
 cin >> input;

  if (input == "quit") {
     break;
  }

  try {
  number = stoi(input);
   } catch (...) {
       cout << "Invalid input. Please enter a valid number or 'quit'." << endl;
       continue;
    }
//write your code here
    write (fd[1], &number, sizeof(int));
    int temp;
    read (out[0], &temp, sizeof(int));
    if (temp){
        // congratsmessage
      } break;

 }//Parent End
    else {
        cout << " could not guess";
        if (tries < 5)
            //try again
        else
            // no more tries
        }
    }
    tries++;
} else { fd[1]); wait (NULL);
return 0;
}
```

***CLO #: 2* Implement solutions employing concepts of Processes and Threads**

Q2: Consider a parent process P that has forked a child process C in the program below. **[Marks:5]**

```
int a = 5;
int fd = open(...) //opening a file
int ret = fork();
if(ret >0) {
close(fd);
a = 6;
...
}
else if(ret==0) {
printf("a=%d\n", a);
read(fd, something);
}
```

After the new process is forked, suppose that the parent process is scheduled first, before the child process. Once the parent resumes after fork, it closes the file descriptor and changes the value of a variable as shown above. Assume that the child process is scheduled for the first time only after the parent completes these two changes.

(a) What is the value of the variable **a** as printed in the child process, when it is scheduled next? Explain.

//Solution

Value changed in parent only.

(b) Will the attempt to read from the file descriptor succeed in the child? Explain.

//Solution

Yes: it will closed by parent

CLO #: 1 Describe services provided by the modern Operating Systems
***CLO #: 2* Implement solutions employing concepts of Processes and Threads**

Q3: Consider a system with two processes P and Q, running a Unix-like operating system as studied in class. Consider the following events that may happen when the OS is concurrently executing P and Q, while also handling interrupts. **[Marks: 5]**

(A) The CPU program counter moves from pointing to kernel code in the kernel mode of process P to kernel code in the kernel mode of process Q.
(B) The CPU stack pointer moves from the kernel stack of P to the kernel stack of Q.
(C) The CPU executing process P moves from user mode of P to kernel mode of P.
(D) The CPU executing process P moves from kernel mode of P to user mode of P.
(E) The CPU executing process Q moves from the kernel mode of Q to the user mode of Q.
(F) The interrupt handling code of the OS is invoked.
(G) The OS scheduler code is invoked.
For each of the two scenarios below, list out the chronological order in which the events above occur. Note that all events need not occur in each question.

(a) A timer interrupt occurs when P is executing. After processing the interrupt, the OS scheduler decides to return to process P.

CFGD.

(b) A timer interrupt occurs when P is executing. After processing the interrupt, the OS scheduler decides to context switch to process Q, and the system ends up in the user mode.

CFGBAE

***CL* *2* Implement solutions employing concepts of Processes and Threads**

Q4: Consider a parent process P that has forked a child process C. Now, P terminates while C is still running. Answer yes/no, and provide a brief explanation. **[Marks: 5]**

(a) Will C immediately become a zombie?

No it will be adopted by init

(b) Will P immediately become a zombie, until reaped by its parent?

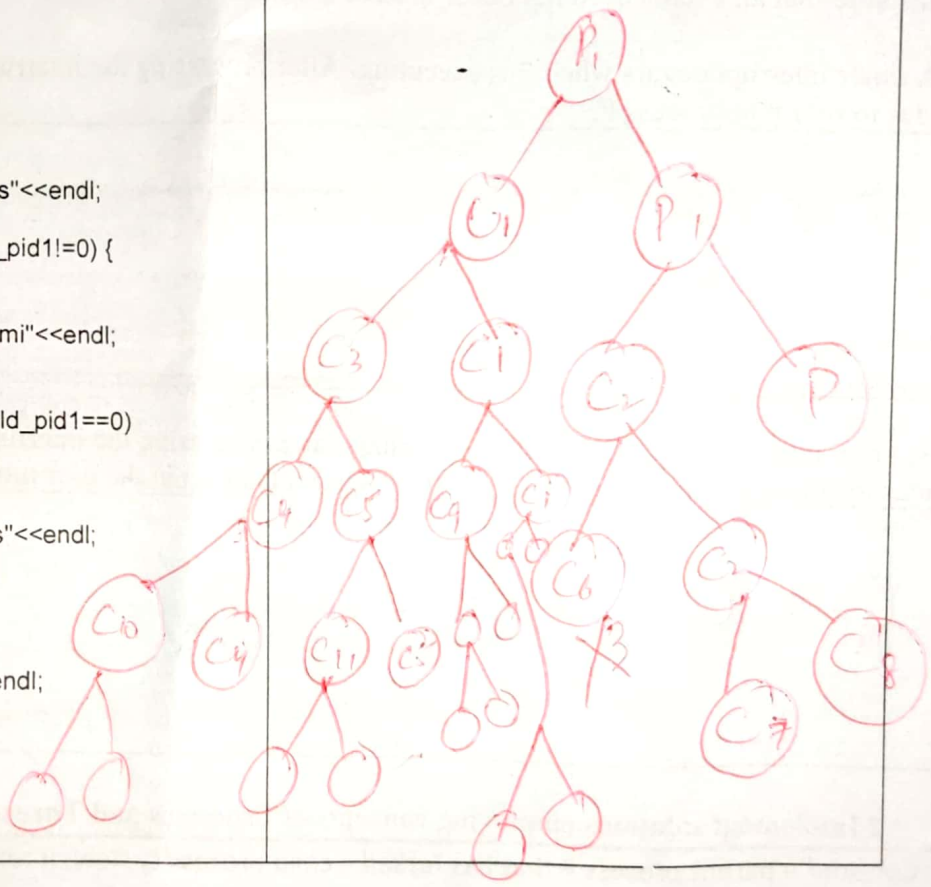
Yes

CLO #: 2 **Implement solutions employing concepts of Processes and Threads**

Q5: Assuming fork() never fails, draw the process tree of the following code, also write how many times "Lahore Qalandars", "Peshawar Zalmi", "Multan Sultans", "Krachi Kings" will be printed on screen. (just write the count). **[Marks: 5]**

```
int main() {
    pid_t child_pid1;
    pid_t child_pid2;
    child_pid1 = fork();
    child_pid2 = fork();
    if (child_pid1 == 0) {
        fork();
        cout<<"Lahore Qalandars"<<endl;
    }
    if (child_pid2 == 0 && child_pid1!=0) {
        if(fork()&& fork())
        {
            cout<<"Peshawar Zalmi"<<endl;
        }
    }
    else if(child_pid2 == 0 || child_pid1==0)
    {
        if( fork() || fork())
        {
            cout<<"Multan Sultans"<<endl;
        }
    }
    else
    {
        cout<<"Krachi Kings"<<endl;
    }
    return 0;
}
```

//Solution



LQ=4

PZ=1

MS=8

KK=1

-16 Total.