Class Assignment #9

Due by May 1, midnight

Modify the template program in pp.166-167 to include the following additional functionalities:

* Instead of 5 children processes, 5 children processes need to be created.
* 5 children processes are connected to the parent process via 5 pipes, one for each pair, child1-parent, child2-parent, child3-parent, ....
* Child keeps writing to the pipe with 1 second apart between consecutive writes (as in child() in p. 167).
* Keyboard inputs need to be monitored as in the original code.
* Parent keeps reading from the 5 pipes and keyboard whenever any of them has data ready for reading.
* You have to use the “select” system call with a non-blocking feature as was done in the template program.

Run your program and submit the screen shots to show your programs source codes and running.

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <string.h>

#include <sys/types.h>

#include <sys/wait.h>

#define MSGSIZE 6

char \*msg1 = "hello";

char \*msg2 = "bye!!";

void parent(int [][2]);

int child(int []);

int main()

{

int pip[6][2]; // 6 children processes are connected to the parent process via 6 pipes

int i;

for(i = 0; i < 6; i++) //"Instead of 3 children processes, create 6 children processes."

{

if(pipe(pip[i]) == -1)

perror("pipe call");

switch(fork())

{

case -1:

perror("fork call");

exit(1);

case 0:

child(pip[i]);

exit(0);

}

}

parent(pip);

exit(0);

}

void parent(int p[6][2])

{

char buf[MSGSIZE], ch;

fd\_set set, master;

int i;

for(i = 0; i < 6; i++)

close(p[i][1]);

FD\_ZERO(&master);

FD\_SET(0, &master);

for(i = 0; i < 6; i++)

FD\_SET(p[i][0], &master);

while(set = master, select(p[5][0]+1, &set, NULL, NULL, NULL) > 0)

{

if(FD\_ISSET(0, &set))

{

printf("From standard input...");

read(0, &ch, 1);

printf("%c\n", ch);

}

for(i = 0; i < 6; i++)

{

if(FD\_ISSET(p[i][0], &set))

{

if(read(p[i][0], buf, MSGSIZE)>0)

{

printf("Message from child%d\n", i);

printf("MSG=%s\n", buf);

}

}

}

if(waitpid(-1, NULL, WNOHANG) == -1)

return;

}

}

int child(int p[2])

{

int count;

close(p[0]);

for(count = 0; count < 2; count++)

{

write(p[1], msg1, MSGSIZE); //Child keeps writing to the pipe with 1 second apart between consecutive writes

sleep(1);

}

write(p[1], msg2, MSGSIZE);

exit(0);

}

Output:

A screen shot of a computer

Description automatically generated with low confidence