Michael Osei

5/22/2023

COSC 430

Week 14 Assignment

Code:

#include <stdio.h>

#include <stdlib.h>

#include <sys/time.h>

#include <sys/types.h>

#include <sys/wait.h>

#include <unistd.h>

#define MSGSIZE 6

char \*msg1 = "hello";

char \*msg2 = "bye!!";

void parent(int [][2]);

void child(int []);

int main()

{

int pip[5][2];

int i;

// Create five communication pipes and spawn five children

for (i = 0; i < 5; i++)

{

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

{

perror("pipe call");

exit(1);

}

switch (fork())

{

case -1:

perror("fork call");

exit(1);

case 0:

child(pip[i]);

exit(0);

}

}

parent(pip);

return 0;

}

// Parent sits listening on all five pipes

void parent(int p[5][2])

{

char buf[MSGSIZE];

char ch;

fd\_set set, master;

int i;

// Close all unwanted write file descriptors

for (i = 0; i < 5; i++)

{

close(p[i][1]);

}

// Set the bit masks for the select system call

FD\_ZERO(&master);

FD\_SET(0, &master);

for (i = 0; i < 5; i++)

{

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

}

while (1)

{

set = master;

// Select is called with no timeout, it will block until an event occurs

if (select(p[4][0] + 1, &set, NULL, NULL, NULL) == -1)

{

perror("select call");

exit(1);

}

// Check if there is input on standard input

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

{

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

read(0, &ch, 1);

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

}

// Check each pipe for input

for (i = 0; i < 5; 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);

}

}

}

// Check if all children have died

int status;

pid\_t pid;

while ((pid = waitpid(-1, &status, WNOHANG)) > 0)

{

if (WIFEXITED(status))

{

printf("Child %d terminated with exit status %d\n", pid, WEXITSTATUS(status));

}

else

{

printf("Child %d terminated abnormally\n", pid);

}

}

}

}

void child(int p[2])

{

int count;

close(p[0]);

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

{

write(p[1], msg1, MSGSIZE);

// Pause for a random amount of time

sleep(getpid() % 4);

}

// Send final message

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

exit(0);

}

Result:

