Hasib Ar Rafiul Fahim

2019-1-60-036

Lab Tasks

Lab 4:

Task 1:

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/wait.h>

int main()

{

int pipedes1[2], pipedes2[2];

int num;

char \*message;

if(pipe(pipedes1) == -1) {

printf("Pipedes1 error\n");

exit(1);

}

if(pipe(pipedes2) == -1) {

printf("Pipedes2 error\n");

exit(1);

}

pid\_t pid = fork();

if(pid < 0) {

printf("Failed to create child\n");

return 1;

}

if(pid == 0) {

close(pipedes1[0]);

printf("Inside child process %d\n", (int)getpid());

printf("Enter an integer: ");

scanf("%d", &num);

printf("\nSending to the parent process ...\n");

write(pipedes1[1], &num, sizeof(num));

close(pipedes1[1]);

wait(NULL);

printf("Inside child process %d\n", (int)getpid());

close(pipedes2[1]);

char \*result = calloc(256, 4);

read(pipedes2[0], result, sizeof(result));

printf("Received result: %d is an %s number\n", num, result);

close(pipedes2[0]);

printf("Child terminated\n");

exit(1);

}

else {

sleep(5);

close(pipedes1[1]);

printf("Inside parent process %d\n", (int)getpid());

read(pipedes1[0], &num, sizeof(num));

printf("Received an integer: %d\n", num);

close(pipedes1[0]);

if(num%2 == 0) {

message = "even";

}

else {

message = "odd";

}

close(pipedes2[0]);

printf("\nSending to the child process ...\n");

write(pipedes2[1], message, sizeof(message));

close(pipedes2[1]);

wait(NULL);

printf("Parent terminated\n");

}

return 0;

}

Task 2:

#include<stdio.h>

#include<pthread.h>

void \* runner(void \* n){

printf("\nChild Thread\n");

long int nn=(long)n;

int sum=0,i;

for(i=0;i<=nn;i++) sum=sum+i;

printf("\n%d\n",sum);

}

int main()

{

pthread\_t tid,tid2;

pthread\_create(&tid,NULL,&runner,(void \*)3);

pthread\_create(&tid2,NULL,&runner,(void \*)4);

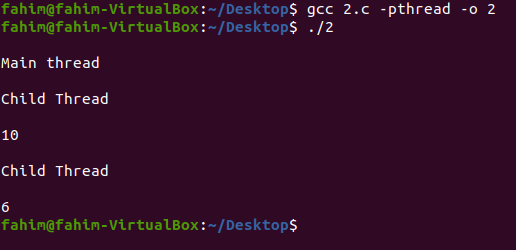
printf("\nMain thread\n");

pthread\_join(tid,NULL);

pthread\_join(tid2,NULL);

return 0;

}



Task 3:

#include<stdio.h>

#include<pthread.h>

void \* runner(void \* n)

{

printf("\nChild Thread\n");

long int nn=(long)n;

long int sum=0;

int i;

for(i=0;i<=nn;i++) sum=sum+i;

return (void \*) sum;

}

int main()

{

void \*sum1,\*sum2;

pthread\_t tid,tid2;

pthread\_create(&tid,NULL,&runner,(void \*)3);

pthread\_create(&tid2,NULL,&runner,(void \*)4);

printf("\nMain thread\n");

pthread\_join(tid,&sum1);

pthread\_join(tid2,&sum2);

printf("%ld\n",(long)sum1);

printf("%ld\n",(long)sum2);

return 0;

}

