#include <stdio.h>

#include <stdlib.h>

#include <sys/types.h>

#include <unistd.h>

void forkexample()

{

int p;

p = fork();

if(p<0)

{

perror("fork fail");

exit(1);

}

// child process because return value zero

else if ( p == 0)

printf("Hello from Child!\n");

// parent process because return value non-zero.

else

printf("Hello from Parent!\n");

}

int main()

{

forkexample();

return 0;

}

/tmp/LC8bcoBzzu.o

Hello from Parent!

Hello from Child!

=== Code Execution Successful ===

#include<stdio.h>

#include<unistd.h>

#include<iostream>

using namespace std;

int main() {

int pipefd[2];

char msg1[20] = "Message 1";

char msg2[20] = "Message 2";

char readmsg[20] = "";

int returnstatus = pipe(pipefd);

if (returnstatus == -1)

{

cout<<"Pipe cannot be created\n";

return 1;

}

int pid = fork();

if (pid < 0)

{

cout<<"Unsuccessful fork command\n";

return 1;

}

if (pid == 0)

{ //child process

close(pipefd[1]);

read(pipefd[0], readmsg, 20);

cout<<"Child read: "<< readmsg<<endl;

read(pipefd[0], readmsg, 20);

cout<<"Child read: "<< readmsg<<endl;

close(pipefd[0]);

}

else

{ //parent process

close(pipefd[0]);

cout<<"Parent writing for the first time \n";

write(pipefd[1], msg1, 20);

cout<<"Parent writing for the second time \n";

write(pipefd[1], msg2, 20);

close(pipefd[1]);

}

return 0;

}

Parent writing for the first time

Parent writing for the second time

Child read: Message 1

Child read: Message 2

=== Code Execution Successful ===