**Ex. No: 10**

**Date:**

**IPC USING PIPES**

**Problem Statement:**

To implement inter process communication using pipes.

**Problem Description:**

To implement IPC using pipes a) one pipe, where parent process writes into the pipe and child reads from the pipe b) two pipes where child 1 reads from parent 2 and child 2 reads from parent 1.

**Algorithm:**

**One pipe with one parent and children**

1. Create a pipe
2. Create a child process pid using fork
3. Check if pid == 0 and close the write end of the pipe
4. Read the data from parent process using read
5. In parent process close the read end of the pipe
6. Write the data to the pipe
7. Wait for child process to finish

**Two pipe with parent and two children**

1. Create two pipes
2. Create a parent pid2 with fork
3. If pid2 = 0. Close the read end of pipe 1 and write end of pipe 2 in parent 2
4. Exit the process without wait
5. Write data to pipe2
6. If pid1 = 0. Close the read end of pipe 2 and write end of pipe 1 in parent 1
7. Write data to pipe 1
8. Exit the process
9. Print the data in the pipes without wait
10. Terminate.

**Code:**

**One pipe with one parent and children**

import os

# Create a pipe

pipe\_read, pipe\_write = os.pipe()

# Create a child process

pid = os.fork()

if pid == 0:

# This is the child process

    os.close(pipe\_write) # Close the write end of the pipe in the child

    child\_data = os.read(pipe\_read, 1024)

    print(f"Child received: {child\_data.decode()}")

else:

# This is the parent process

    os.close(pipe\_read) # Close the read end of the pipe in the parent

    data\_to\_send = "Hello from Parent!"

    os.write(pipe\_write, data\_to\_send.encode())

    os.wait() # Wait for the child process to finish

**Two pipes with two parent and children**

import os

# Create two pipes

pipe1\_read, pipe1\_write = os.pipe()

pipe2\_read, pipe2\_write = os.pipe()

# Create Parent 2

pid2 = os.fork()

if pid2 == 0:

# This is Parent 2

    os.close(pipe1\_read) # Close the read end of Pipe 1 in Parent 2

    os.close(pipe2\_write) # Close the write end of Pipe 2 in Parent 2

    data\_to\_send2 = "Hello from Parent 2 to Child 11!"

    os.write(pipe1\_write, data\_to\_send2.encode())

    os.\_exit(0) # Exit the child process without waiting

else:

# Create Parent 1

    pid1 = os.fork()

if pid1 == 0:

    # This is Parent 1

    os.close(pipe2\_read) # Close the read end of Pipe 2 in Parent 1

    os.close(pipe1\_write) # Close the write end of Pipe 1 in Parent 1

    data\_to\_send1 = "Hello from Parent 1 to Child 21!"

    os.write(pipe2\_write, data\_to\_send1.encode())

    os.\_exit(0) # Exit the child process without waiting

else:

    # This is the main parent process

    # Wait for both child processes to finish

    os.waitpid(pid1, 0)

    os.waitpid(pid2, 0)

    message\_from\_child1 = os.read(pipe1\_read, 1024).decode()

    message\_from\_child2 = os.read(pipe2\_read, 1024).decode()

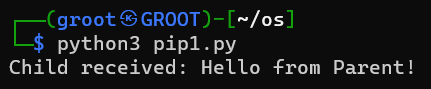
    # Print the messages

    print(message\_from\_child1)

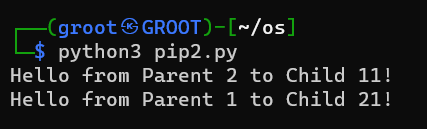
    print(message\_from\_child2)

**Output**

**One pipe with one parent and children**

****

**Two pipes with two parent and children**

****

**Result:**

Thus, inter process communication between processes using pipes has been executed successfully.