Fork Pipe

This assignment gives the programmer experience with forking to create another process. Pipes are used to communicate between the parent process and the child process. This assignment will specify what the parent process and the child process requirements are in the program.

General Program Requirements

Use the LineInfo.h exception handler and the associated throws described in the Assignment Guides try-catch mechanism for calling all operating system services through the Unix APIS.

All arrays that require a size declarator, like the pipe message and quote string array should be of size declarator of 1000.

Use global constants for use by the program.

Use FILE and the fopen file commands for all file operations.

Name the source code program forkpipe.cpp.

Name the run execution file forkpipe.

Main Program

Program Command Line

The running program will accept from the command line one argument number:

```
./forkpipe <number>
```

Where number is the number of quotes to request.

example:

```
./forkpipe 2
```

The program must check to make sure that one number argument is provided. If this is not the case then an appropriate error message using a LineInfo.h throw.

See the end of this document to see a sample run.

Quotes File

The quotes file is located at:

~smd013000/courses/3377/assigns/03/quotes.txt

Use a function with the following prototype header to create an array that holds quote lines read in from a provided text file named quotes.txt.

Create a Quotes Array

```
void getQuotesArray(char *lines[], unsigned &no Lines)
```

```
lines   : the array that holds cstring quote lines from the quotes.txt file.
nolines : passes back the number of lines in the array
```

Pipes

Create 2 pipes used for parent child communication that return a status variable.

Fork Pipe

Use LineInfo.h throws to check the return status to make sure the pipes were created without error.

Create Another Process

Use fork();

Based on the pid returned by fork() determine if the process is the parent process or the child process or the fork failed.

Processes

Display all message activities as shown in the run sample at the end of this document.

Parent Process

```
void executeParentProcess(int pipeParentWriteChildReadfds[],
                          int pipeParentReadChildWritefds[],
                          int noOfParentMessages2Send)
{

}
```

You are to design and code up the above process that is invoked after the fork pid is determined to be the parent pid.

As usual the process must close unnecessary part of pipes that are not used.

Create a character array for the Parent Read Child Message to hold messages from the child.

Design a loop that uses the command line argument number set at the program beginning to indicate how many Get Quote messages to send to the child.

Inside the loop:

- Send using a write to the Parent Write Child Read pipe a Get Quote message to the child.
- Declare a character array of the correct size and clear it.
- Use a read of the Parent Read Child Write pipe to get a Quote message from the child.

When the loop is done write to the Parent Write Child Read pipe an Exit message.

Close the pipes the parent used.

Display that the parent is done.

Fork Pipe

Child Process

```
void executeChildProcess(int      pipeParentWriteChildReadfds[],
                         int      pipeParentReadChildWritefds[],
                         char*    lines[],
                         unsigned noLines)
{

}
```

You are to design and code up the above process that is invoked after the fork returned pid is determined to be the child pid.

As usual the process must close unnecessary part of pipes that are not used.

Seed a random number generator.

Design an infinite loop.

Inside the Loop:

- Read the received message from the parent.
- Determine what the message is:
  - If the message is an Exit
    - Leave the infinite loop.
  - If the message is Get Quote
    - Use random number generator to get a random quote line from the quote array.
    - Declare and clear a char send array using correct size
    - Copy to the send array from quote array.
    - Write this quote to the parent.
  - If the message is invalid
    - Display to the screen that the message is invalid and sent back to the parent
    - Write this invalid message to the parent.

Close the pipes the child used.

Display to the screen a that the child is done.

Activities File

After done designing and testing, capture the run results in an activities text file named forkpipe.txt.

Use the copy and paste to local editor text file for creating the forkpipe.txt file.

Do not use script or terminal session to create the forkpipe.txt file.

See the sample run below.

Submittal

Put in a zip folder that you name:
forkpipe.cpp
forkpipe.txt
quotes.txt

Submit whatever you called the .zip folder to blackboard.

Fork Pipe

```
{cslinux1:~/courses/3377/assigns/03/forkpipe}  g++ forkpipe.cpp -o forkpipe
{cslinux1:~/courses/3377/assigns/03/forkpipe} ./forkpipe

Usage: ./forkpipe <number>

{cslinux1:~/courses/3377/assigns/03/forkpipe} ./forkpipe 2


In Parent: Write to pipe getQuoteMessage sent Message: Get Quote
In Child : Read from pipe pipeParentWriteChildMessage read Message: Get Quote
In Child : Write to pipe pipeParentReadChildMessage sent Message:
“Art is not what you see, but what you make others see.” ~ Edgar Degas

In Parent: Read from pipe pipeParentReadChildMessage read Message:
“Art is not what you see, but what you make others see.” ~ Edgar Degas


----------------------------------
In Parent: Write to pipe getQuoteMessage sent Message: Get Quote
In Child : Read from pipe pipeParentWriteChildMessage read Message: Get Quote
In Child : Write to pipe pipeParentReadChildMessage sent Message:
“No act of kindness, however small, is ever wasted” ~Aesop

In Parent: Read from pipe pipeParentReadChildMessage read Message:
“No act of kindness, however small, is ever wasted” ~Aesop


----------------------------------
In Parent: Write to  pipe ParentWriteChildExitMessage sent Message: Exit
Parent Done
In Child : Read from pipe pipeParentWriteChildMessage read Message: Exit
Child Done

{cslinux1:~/courses/3377/assigns/03/forkpipe}
```