CECS 326 - Operating Systems

Assignment 3 - Fork Program

Name: Eric Do
Professor: Ratana Ngo
Date: April 9th, 2019
Email: ericdo62497@gmail.com

Program Overview

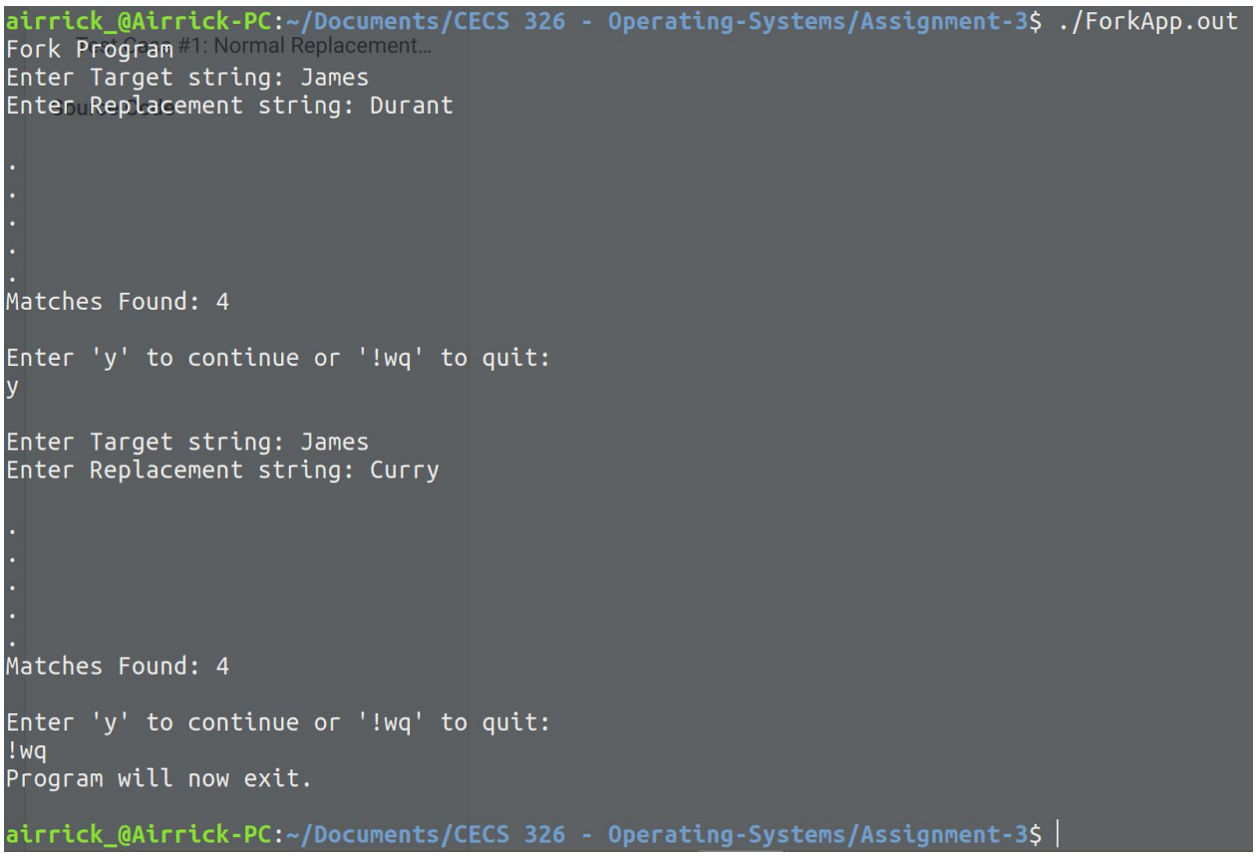
In this program, we experiment on how forking a parent and child process works.

A 300+ word plain text has been hard coded into the program as a string. When we run the program in the terminal, we prompt the user for a target word and a replacement word. After prompting the user, the program spawn a child process that will search through the plain text for the target word and replace all occurrences with the replacement word. The parent process will wait for the child process to finish before it starts running again. Every time the child finds and replaces a word, it will print out a dot (“.”). If the target word is found at least once, then it will print out the number of matches it found and replaced. Otherwise, an injected bug will occur which will cause the program to go in an infinite loop to try to find the target word.

This is where a second terminal comes into play. We will find the child process’s PID through the command, ‘ps aux’ and kill the child process. After killing the child process, the parent process will resume and the program will continue running as normal. The program will finally prompt the user to continue or quit. If continue, then the program will repeat, using the original plain text.

Test Cases

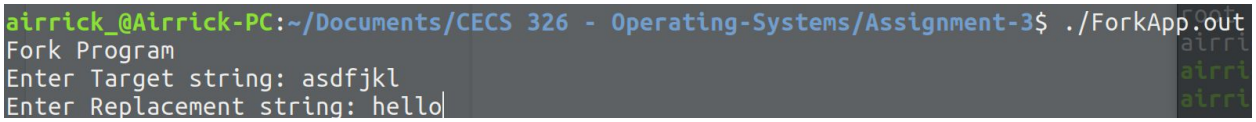
Test Case #1: Normal Replacement Task



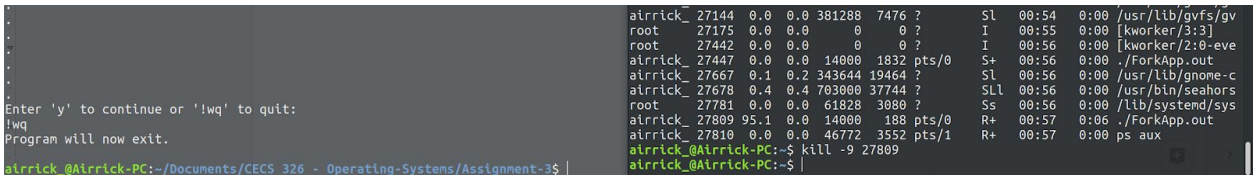
- "James" occurs 4 times in the document.
- Still able to find "James" 4 times in the 2nd try.
    - Each child works on the original document.

This test case shows that the find and replacement functionality works. It also shows that when the child finds and replaces the word, the original document that the parent has is still the same. Thus, everytime the child spawns to do it's replacement task, it will always work on the original document..

Test case #2: Killing Child Process



- infinitely prints dots



- Kill child process
- Stops printing dots and able to continue or exit program

This shows that even after killing the child process, the parent process will still continue running as normal and the user can still attempt to find and replace a word.

Source Code

```
#include <iostream>
#include <unistd.h>
#include <cstdlib>
#include <sys/types.h>
#include <sys/wait.h>

using namespace std;

int main()
{
  cout << "Fork Program" << endl;

  // Declare variables
  string target;
  string replacement;
  string document = "The Los Angeles Lakers and their medical staff made the decision for LeBron James. The team ended LeBron’s season on Saturday. The superstar had been dealing with a groin injury, a slight tear in his left groin, since Christmas Day. The issue held him out of 17 consecutive games, the longest streak James had ever missed with an injury. With next season in mind, the Lakers want LeBron to heal – and, in turn, they want him off the court. 'He wants to play,' Lakers coach Luke Walton said on Sunday, via ESPN. “My understanding from the medical staff, they finally said, ‘Look, it’s just not worth it anymore. Let’s make sure you have a healthy summer.’ So that’s the decision that was made, and we’ll move forward without him on the floor for the final six.” LeBron, 34, will continue to travel with the team to provide leadership and build team chemistry for the remainder of the season. Individually, his season was a successful one, even if the Lakers missed the playoffs and may not top 40 wins. In 55 games, LeBron put up 27.4 points per game, his second highest average since the 2009-10 season. He also averaged 8.5 rebounds per game, the third highest total in his career. And he had 8.3 assists per game, his fourth-highest total of his career. Why didn't Coach K give Zion Williamson the ball? There’s one obvious oddity as it pertains to Los Angeles’ decision. The Lakers were officially out of the playoff hunt after a loss on March 22, yet LeBron played in three of the games that followed and averaged 33 minutes in those technically meaningless appearances. Walton was asked why he and his staff didn’t end James’ season when Lakers knew they were out of the playoffs. “Eventually, as that time goes on and on, it’s easier to tell someone like that, ‘Let’s take care of your health right now,'” Walton said, via ESPN. It’s a bit of a cryptic answer that raises more questions. Were the Lakers trying to shut down LeBron sooner? Were the Lakers struggling to convince LeBron to stop playing? Were they uncomfortable broaching the issue sooner with their superstar? Whatever the reason for the delay, James’ season is done. His streak of eight consecutive NBA Finals appearances is also concluded. And the NBA playoffs won’t feel the same without him. His ability to thrive in the playoff drama, no matter the caliber of his teammates, has long been one of the most fun part of viewing the playoffs.";

  bool program = true;
  while (program)
```

```
    {
      // Prompt user for target & replacement string
      cout << "Enter Target string: ";
      cin >> target;
      cout << "Enter Replacement string: ";
      cin >> replacement;
      cout << endl;

      long childPID;
      childPID = fork();

      if (childPID > 0) //Parent waits for child
      {
        wait(0);
        cout << "Enter 'y' to continue or '!wq' to quit: " << endl;
        string userQuit;
        cin >> userQuit;
        if (userQuit == "!wq")
        {
          cout << "Program will now exit." << endl;
          program = false;
        }
      }
      else if (childPID == 0) //Child performs replacement task
      {
        int matchCount = 0;

        bool finding = true;
        while (finding)
        {
          cout << "." << endl;
          // Returns index of first match, or string::npos if not found
          size_t foundIndex = document.find(target);

          if (foundIndex != string::npos)
          {
            document.replace(foundIndex, target.size(), replacement);
            matchCount++;
          }
          else if (matchCount != 0) //Bug injection. When matchCount > 0 and cannot find
anymore, exit the loop.
          {
            finding = false;
          }
          else //Bug injection. Will keep on repeat when matchCount = 0
          {
            /*
            finding = false;
            */
          }
```

```
        }
        cout << "Matches Found: " << matchCount << endl;
        cout << endl;
        break;
      }
      cout << endl;
    }
    return 0;
  }
```