Daniel Garcia

ECE 565

Homework 2

1. Three child processes are created. When fork is called the first time, the parent and the child execute the second fork so the parent calls fork twice and the child calls fork once.
2. When a thread is stopped, it has values in the registers which must be saved. Just as when the process is stopped, the registers must be saved. Multiprogramming threads are no different than multiprogramming processes, so each thread needs its own set of register values to be saved.

3.

1. Round Robin:

The finishing times for the five jobs are 10, 18, 24, 28, and 30, for an average of 22 minutes (= 110/5).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | ~~C~~ | D | E | A | B | ~~D~~ | E | A | ~~B~~ | E | A | ~~E~~ | ~~A~~ |  |
| 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 | 20 | 22 | 24 | 26 | 28 | 30 |

1. Priority Scheduling:

B is run first. After 6 minutes it is finished. The other jobs finish at 14, 24, 26, and 30, for an average of 20 minutes (= 100/5).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| B | | | E | | | | A | | | | | C | D | |  |
| 0 |  |  | 6 |  |  |  | 14 |  |  |  |  | 24 | 26 |  | 30 |

1. First-Come, First-Serve (run in order 10, 6, 2, 4, 8):

If the jobs run in the order A through E, they finish at 10, 16, 18, 22, and 30, for an average of 19.2 minutes (= 96/5).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | | | | | B | | | C | D | | E |  |  |  |  |
| 0 |  |  |  |  | 10 |  |  | 16 | 18 |  | 22 |  |  |  | 30 |

1. Shortest Job First:

It yields finishing times of 2, 6, 12, 20, and 30, for an average of 14 minutes (= 70/5).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| C | D | | B | | | E | | | | A | | | | |  |
| 0 | 2 |  | 6 |  |  | 12 |  |  |  | 20 |  |  |  |  | 30 |

Therefore, shortest job first is the best algorithm for scheduling algorithms because it guarantees that process spend the least time waiting, however due to penalizing long processes, it is likely that a long process will not execute which may deny some critical function.

4.

* Condition 1. Mutual Exclusion:

The solution satisfies Mutual Exclusion Condition since it is not possible for both processes to be in their critical sections at the same time. That is, when turn is 0, P0 can execute its critical section, but not P1. Likewise, when turn is 1.

* Condition 2. Progress with No Deadlock:

The solution does NOT satisfy Progress with No Deadlock Condition since this system of two processes will not make any progress. For example, if P1 produces something and wants to enter its critical section for placing it in their shared buffer, then while P0 can get into its critical section, it will find the shared buffer empty and will be blocked by P1 who runs outside its critical section. Meanwhile, P1 is already blocked by P0 running inside its critical section, waiting to consume what is produced by P1. P0 and P1 stand deadlock.

* Condition 3. Bounded Waiting:

The solution does NOT satisfy Bounded Waiting Condition either. Since P0 is assumed to run first with turn initialized to 0, P1 would wait forever for being granted its request to enter its critical section since it has to pass its Critical Section before it can set turn to 1 for it to enter its critical section.

* Condition 4. Unrestricted Processors:

The solution satisfies Unrestricted Processors Condition since it does not post any assumption on CPUs, and speeds or the number of CPUs do not affect the outcomes of the solution discussed before.

5. The employees communicate by passing messages: orders, food, and bags in this case. In UNIX terms, the four processes are connected by three pipes.

6. Code:

# Author: Daniel Garcia

# Student ID: 01411723

# ECE 565 2017 HW#2 Problem 6

param (

[Parameter(Mandatory=$false)][string]$filename = "ece565hw02.txt"

)

function main{

param (

[Parameter(Mandatory=$true)][string]$filename

)

Write-Host "Starting Content Switching Simulation Program"

$CPUQueue = parsedata $filename

firstComeFirstServe $CPUQueue

}

function firstComeFirstServe{

param (

[Parameter(Mandatory=$true)][pscustomobject[]]$CPUQueue

)

$CPUTime = 0

foreach($process in $CPUQueue){

if("Blocked" -ne $process.State){

Write-Host "Loading CPU with PID $($process.PID)"

$timeRemaining = [int]$process.CPU

while($timeRemaining -gt 0){

Write-Host "Running... time remaining: $timeRemaining"

$timeRemaining--

$CPUTime++

}

Write-Host "Process $($process.PID) Complete"

Write-Host "CPU Time: $CPUTime"

}else{

Write-Host "Process $($process.PID) is blocked and will not be executed"

}

}

Write-Host "No more processes in queue... Exiting"

}

function parsedata{

param (

[Parameter(Mandatory=$true)][string]$filename

)

if(Test-Path $filename){

$fileContents = Get-Content $filename

}else {

return $null

}

$processes = @()

$process = New-Object pscustomobject

foreach($line in $fileContents){

if($line -eq ''){

$processes += $process

$process = New-Object pscustomobject

}else{

$keyValuePair = ($line -split ':\s+')

$process | Add-Member -MemberType NoteProperty -Name "$($keyValuePair[0])" -Value "$($keyValuePair[1])"

}

}

return $processes

}

main $filename

Output:

Starting Content Switching Simulation Program

Loading CPU with PID 12

Running... time remaining: 6

Running... time remaining: 5

Running... time remaining: 4

Running... time remaining: 3

Running... time remaining: 2

Running... time remaining: 1

Process 12 Complete

CPU Time: 6

Loading CPU with PID 11

Running... time remaining: 2

Running... time remaining: 1

Process 11 Complete

CPU Time: 8

Loading CPU with PID 111

Running... time remaining: 4

Running... time remaining: 3

Running... time remaining: 2

Running... time remaining: 1

Process 111 Complete

CPU Time: 12

Process 1 is blocked and will not be executed

No more processes in queue... Exiting

How to Run:

Open a powershell window (Powershell 5.0)

Run the script “.\Homework2-ContextSwitching.ps1 ece565hw02.txt”

The script will output the same output above