

1º Relatório

Projeto 2020

Pedro Carrasco, Afonso Simões, Pedro Lopes, Luís Espírito Santo| Sistemas Operativos | 28/04/2020

Introdução

Neste relatório vamos mencionar o nosso conteúdo no trabalho até a data.

Todos os requisitos pedidos para a primeira entrega (12 maio) estão feitos como a implementação do algoritmo FIFO.

O nosso programa está a ler os programas e executar os seus processos e também a reportar a informação dessa mesma execução no ecrã.

Vamos explicar as funcionalidades do programa, as suas componentes e demonstrar que os requisitos pedidos até a data, encomtra-se no trabalho.

Estruturas de dados “processo”



Esta estrutura contém os elementos em baixo referidos:

* nome que representa o nome do processo;
* pid que representa o identificador de processo gestor pid = 0;
* ppid que representa o pai da pid, se este for filho;
* prioridade que representa a prioridade de execução 1 > 2 > 3;
* tempoVida que representa o tempo a ser executado;
* PC que representa o número de instruções;
* processValue que representa o valor do processo;
* quantidadeDeIntrucoes que representa o número de instruções do programa.

Esta estrutura premite de forma simples guardar toda a informação que define um programa, no sentido em que se for preciso consultar alguma informação por exemplo o pid ou ppid de um programa apenas será preciso procurar pelo nome do programa a estrutura associada a informação.

Estruturas de dados “programa”



Esta estrutura contém os elementos em baixo referidos:

* nomeProg que contem o nome do programa;
* processo que contem a struct com informações do programa;
* listaDeInstrucoes que contem a lista com as operações do programa;
* estado que contem todos os estados do programa RUNNING 0 / PARADO 1 / MORTO 2.

Desta forma fica facil manipular quando um programa deve ou não ser executado, por exemplo se um programa de maior prioridade aparecer simplesmente será necessário trocar o programa em execução sem haver perda da posição da execução do programa anterior.

Optou-se por usar um array strings(array de array de caracteres, C) para a organização das instrucções por este simbulizar o método mais facil de implementar e provavelmente manipular no tipo de estrutruas possiveis, como por exemplo filas,pilhas,arvores,listas que requerem uma maior capacidade de abstração na sua implementação.

A nossa interpretação e logica do projeto

Como o enunciado do projeto é ambíguo em relação a como deve este ser implementado e qual o caminho a ter, abaixo encontra-se explicado a metodologia que terá sido usada.

Na atual situação do projeto encontra-se disponivel um sistema de escalonamento não preemptivo, ou seja, não flexivel na execução de programas, isto é cada programa é lido e executado, ignorando a existência de outros programas que possam ou não ter maior prioridade que ele. Para que tal assim fosse implementado através da função "programaRunnerFifo" é executada uma sequência que apos a leitura através da função "atribuidorDeIntrucoes" e após ser unificada pela função "juntor" definidas abaixo, é de imediato executado a função "percorrerIntrucoes", o que leva a que seja impossivel um processo, impedir a execução de outro, visto que o segundo apenas será definido após a conclusão do primeiro.







Deforma a poder-se simular o fork(), no caso instrução C, utilizou-se uma função auxiliar "filho" de forma a que se possa evitar confusões na execução de instruções apos o fork(), numa possivel implementação esta função irá retornar o pid do filho para esta seja mais fidígna á implementação do fork em C.

Para simular a instrução exec() do C, no caso intrução L voltou-se a chamar a função"atribuidor de Intruções".

Execução do metodo de escalonamento FIFO(não preemptivo)

