Relatório sobre o 1º trabalho

Henrique Rosa(51923) Alice Martins(52768)
Diogo Matos(54466)

30 de junho de 2023

Objetivo do Trabalho

O objetivo deste trabalho é desenvolver um simulador de um modelo de 5 estados: NEW, READY, RUN, BLOCKED, EXIT. Quando os processos estão em READY ou BLOCKED, estão em filas de espera do tipo FIFO (first in first out). A parte 2 do trabalho consiste em ler o input do simulador a partir de um ficheiro.

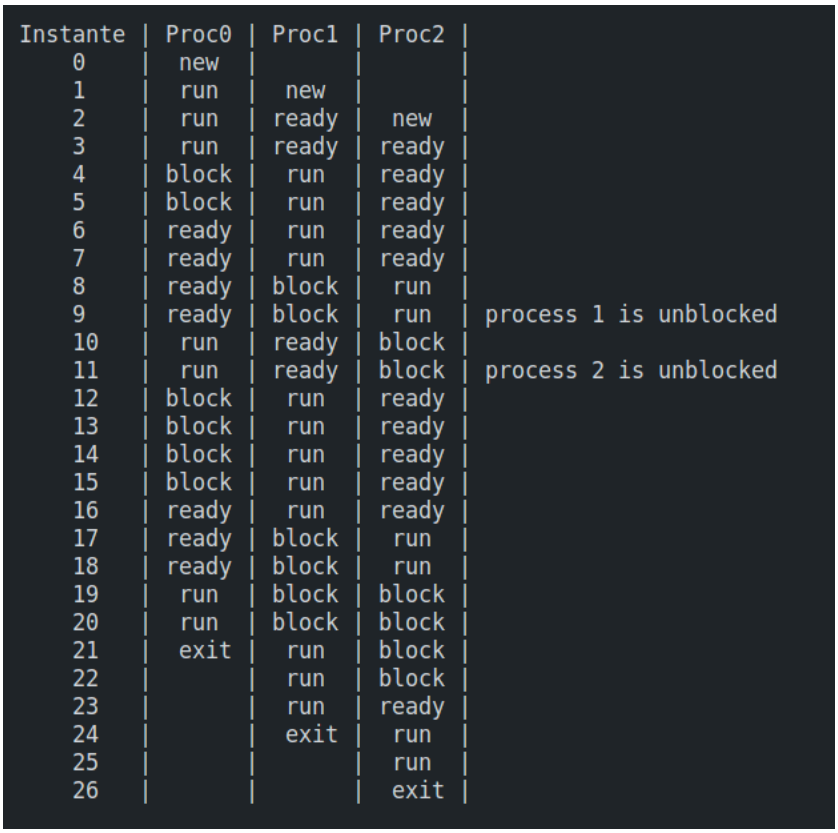
Resumo do Código

O código apresentado é um simulador de escalonamento de processos. Ele realiza a execução de programas em diferentes estados (new, ready, running, blocked, exit) com base num conjunto de dados fornecidos num ficheiro de texto chamado "proc.txt".

Neste código, são definidas algumas constantes para representar os diferentes estados dos programas. A função soma_instantes calcula a soma de todos os instantes de todos os programas. A função output é responsável por imprimir os estados de cada programa em cada instante. A função all_exit verifica se todos os programas estão no estado de saída (exit). A função main é o ponto de entrada do programa.

No início da função main, é aberto o arquivo "processes.txt"para leitura e são lidos o número de programas e o número máximo de mudanças. Em seguida, é criada uma fila do tipo QUEUE chamada "ready"para representar o estado ready dos programas. O estado inicial de todos os programas é definido como "antes de entrar"(bnos) e os iteradores são inicializados como 0. É impressa a primeira linha do output, indicando os estados dos programas em cada instante. Os programas que estão no estado "new"no instante zero são inicializados. O loop principal começa e continua até que todos os programas tenham atingido o estado de saída (exit) ou até que todos os instantes tenham sido executados.

Dentro do loop, são executadas várias ações de acordo com as transições de estado dos programas: Processos em estado "exit"são marcados como "sem estado"(nos). Processos em estado "run"podem transitar para "exit"se o tempo



Instante	Proc0	Proc1	Proc2	
0	new			
1	run	new		
2	run	ready	new	
3	run	ready	ready	
4	block	run	ready	
5	block	run	ready	
6	ready	run	ready	
7	ready	run	ready	
8	ready	block	run	
9	ready	block	run	process 1 is unblocked
10	run	ready	block	
11	run	ready	block	process 2 is unblocked
12	block	run	ready	
13	block	run	ready	
14	block	run	ready	
15	block	run	ready	
16	ready	run	ready	
17	ready	block	run	
18	ready	block	run	
19	run	block	block	
20	run	block	block	
21	exit	run	block	
22		run	block	
23		run	ready	
24		exit	run	
25			run	
26			exit	

Figura 1: Output do Exemplo do Enunciado

de execução for zero. Processos em estado "run"podem transitar para "blocked"se o tempo de execução for zero e o próximo estado for diferente de zero. Processos em estado "blocked"podem transitar para "ready"se o tempo de bloqueio for zero. Processos em estado "new"podem transitar para "ready". Processos em estado "ready"podem transitar para "run"se não houver nenhum processo em execução. Processos bloqueados que precisam ser desbloqueados são colocados no estado "ready". Os tempos de execução e bloqueio são atualizados. O instante é incrementado. É feita a impressão dos estados no output para o instante atual.

Após o loop, o programa termina e é retornado 0.

Comando para executar o código: gcc trabalho.c queue.c -o teste

Diferenças entre o primeiro trabalho e este

A maior diferença é essencialmente entre os dois trabalhos foi a diferença entre os inputs.

O primeiro enunciado descreve a implementação de um simulador de Sistema Operativo utilizando a linguagem C. O simulador é baseado em um modelo de 5 estados: NEW, READY, RUNNING, BLOCKED e EXIT. Os processos são representados por conjuntos de pseudo-instruções-máquina, compostas por números que alternam entre o tempo que o processo fica no estado RUNNING e o tempo que gasta no estado BLOCKED. Os processos são organizados em filas de espera do tipo FIFO (first in first out) nos estados READY e BLOCKED. O simulador deve seguir algumas regras, como a mudança instantânea entre estados, o tempo de permanência de um processo em cada estado, a transição de NEW para READY, a transição de BLOCKED para READY, a transição de RUNNING para EXIT, entre outras. O escalonamento utilizado é o algoritmo FCFS (first-in first-out). O simulador deve fornecer como output o estado de cada processo em cada instante de tempo.

O segundo enunciado faz referência a um trabalho prático relacionado ao primeiro enunciado. A parte 1 consiste em desenvolver um simulador baseado em um modelo de 5 estados (NEW, READY, RUN, BLOCKED, EXIT) e implementar a comunicação entre processos através da instrução UNBLOCK. A instrução UNBLOCK permite que um processo bloqueado seja desbloqueado por outro processo. A parte 2 do trabalho consiste em ler o input do simulador a partir de um arquivo, onde o formato do arquivo é descrito, mas não está presente na pergunta.

Instruções de Uso

Neste código, é necessário haver um ficheiro, chamado "proc.txt"com os dados do programa a ler. Também necessita de ter todos o programas a terminar com os últimos dois valores a 0.

Para além disso, o ficheiro tem de ter o formato correto. A primeira linha contem o número de programas e o número máximo de mudanças. As restantes linhas têm o valor respetivo de cada linha da matriz.

No ultimo exemplo, dado pelo professor na segunda versão do enunciado, o ultimo valor do ultimo programa não faz sentido (estaria a tentar desbloquear um programa "0", sendo que a contagem dos programas na matriz começa no "1 "), pelo que os testes foram realizados com esse exemplo, modificado para ter a sequência de dois zeros no fim e para não ter aquele zero que não faz sentido. Esse exemplo será o que vamos usar para demonstrar o funcionamento do nosso programa aquando da apresentação.