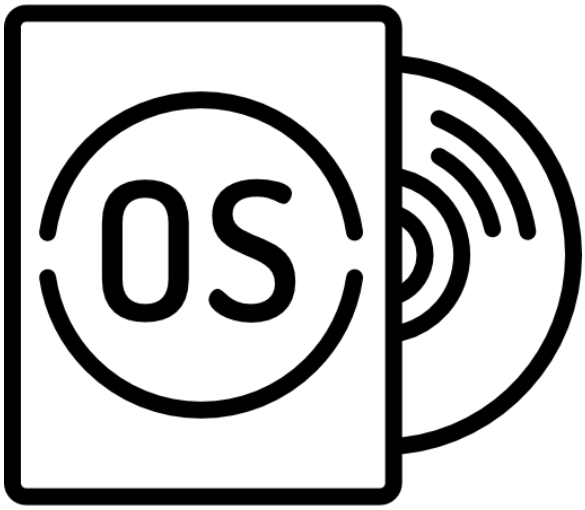
UNIVERSIDADE DE ÉVORA

Relatório do Trabalho Prático

Simulador de Sistema Operativo

Modelo de 5 estados



Curso: Engenharia Informática

Disciplina: Sistemas Operativos

Docentes: Luís Rato

Entregue Abril 2022

Alunos:

Joana Carrasqueira nº48566

João Condeço nº48976

João Oliveira nº48979

Descrição da estrutura do programa

De modo, a simular um sistema operativo foi criada uma struct (geral) que continha a informação necessária para gerir devidamente os processos em causa. Para tal, têm-se os estados dos processos num apontador de inteiros states, bem como a posição do processo no array do input (int *pos), o número de processos envolvidos (int nprocesses), o processo que se encontra no estado RUN (int running), a fila do estado BLOCKED (struct QueueRecord blocked), a fila do estado READY (struct QueueRecord ready), o número de instantes em que o processo se repete (int comp), e por fim, o valor do Quantum (int Quantum). Foi também utilizada uma estrutura auxiliar (struct work) onde são guardadas as informações individuais de cada processo e o instante em que entrou para o estado RUN.

Com esta estrutura desenvolvemos a função processHandler, onde começamos por verificar se se trata da abordagem Round Robin (com Quantum 3) ou FCFS. Se for FCFS igualamos o valor do Quantum à variável comp caso contrário fica como está.

De seguida tratamos primeiro processo que se encontra no estado RUN de modo a assegurar que quando os processos no estado READY verificam se o RUN está livre este já esteja processado. Para tal começamos por verificar se este já terminou, se for o caso removemo-lo do estado RUN e colocamos o mesmo no próximo estado com base nas suas necessidades, caso contrário, se exceder o valor do Quantum, passamos o mesmo para a fila do READY.

Uma vez assegurado o estado READY passamos à análise dos restantes processos. Primeiramente verificamos se o processo ainda não começou, se não tiver começado e o instante atual corresponder ao tempo de chegada do processo este passa para o estado NEW.

Se o processo já tiver sido iniciado verificamos se se encontra no estado READY. Se for o caso e estiver na frente da fila e o estado RUN estiver vazio o processo transita para este ultimo estado.

Caso esteja no estado BLOCK e já tenha estado os ciclos exigidos no mesmo irá averiguar se as condições para passar diretamente para o RUN estão presentes, caso contrário passa para a fila do READY.

Na hipótese de se encontrar no estado NEW o processo simplesmente passa para a fila do estado READY. Caso esta fila esteja vazia e o estado RUN estiver vazio este pode passar diretamente para o estado RUN.

Por fim, quando estiver no estado EXIT, este ocorre apenas um instante finalizando assim o processo.

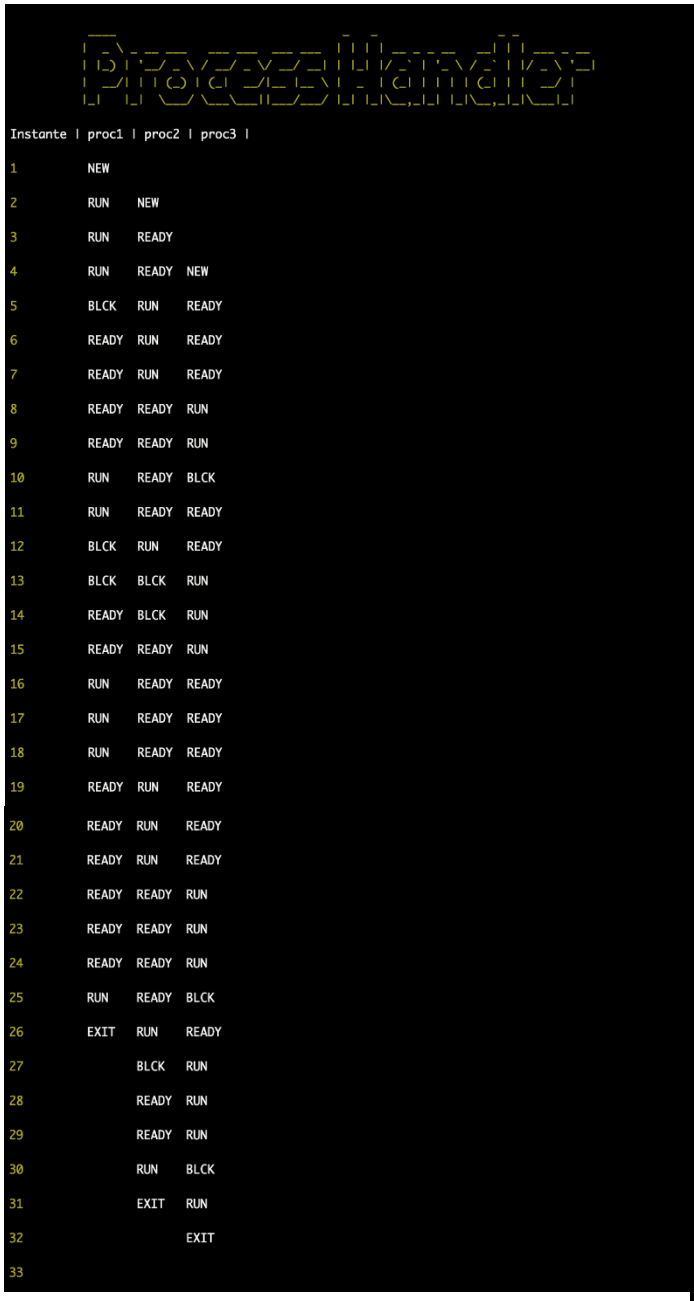
Este processo de tratamento dos dados acontece através da execução da função processHandler (descrita anteriormente) instante a instante na função main onde são recolhidas/definidas previamente as informações necessárias para a execução da mesma.

De forma a gerir as duas filas do programa (referentes ao estado BLOCK e READY) foi implementado o tipo abstrato de dados QUEUE para permitir uma melhor gestão dos dados.

Dificuldades encontradas

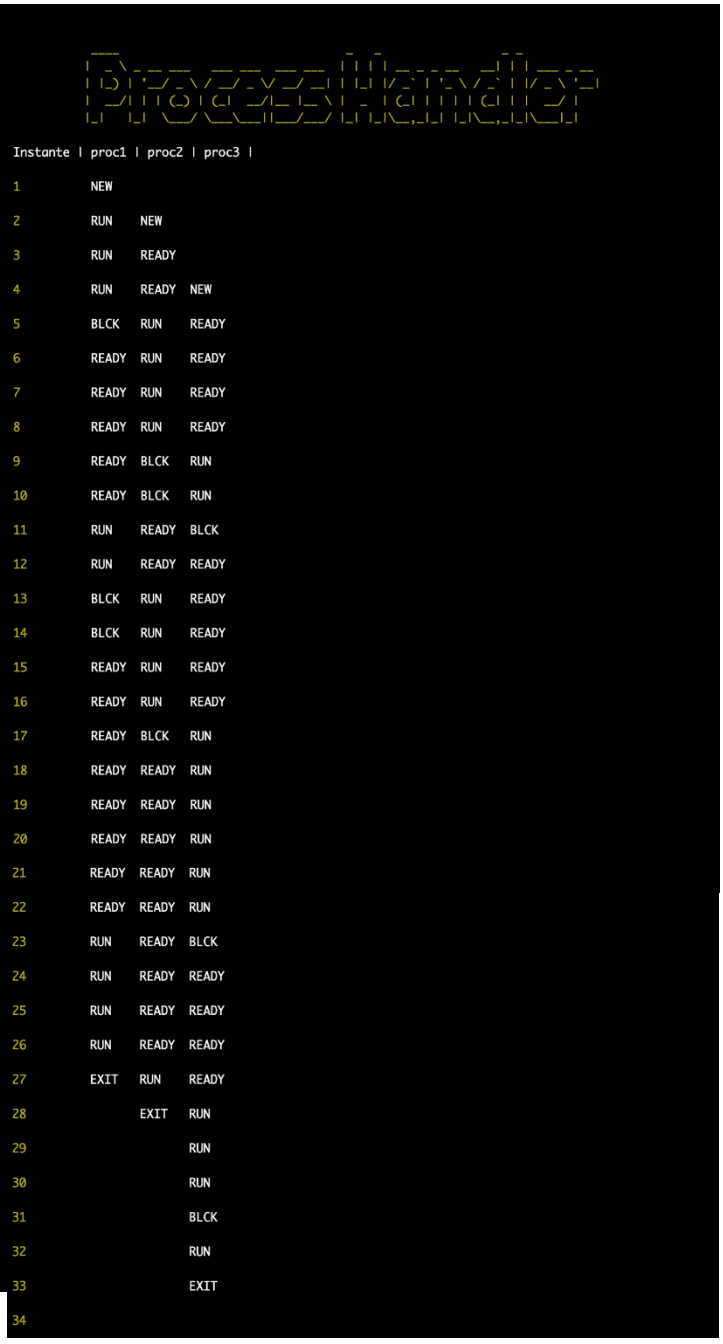
Durante a elaboração do trabalho foi encontrada uma dificuldade ao nível da implementação da estrutura de modo a gerir o espaço na memória devidamente.

Outputs



Instante	proc1	proc2	proc3
1	NEW		
2	RUN	NEW	
3	RUN	READY	
4	RUN	READY	NEW
5	BLCK	RUN	READY
6	READY	RUN	READY
7	READY	RUN	READY
8	READY	READY	RUN
9	READY	READY	RUN
10	RUN	READY	BLCK
11	RUN	READY	READY
12	BLCK	RUN	READY
13	BLCK	BLCK	RUN
14	READY	BLCK	RUN
15	READY	READY	RUN
16	RUN	READY	READY
17	RUN	READY	READY
18	RUN	READY	READY
19	READY	RUN	READY
20	READY	RUN	READY
21	READY	RUN	READY
22	READY	READY	RUN
23	READY	READY	RUN
24	READY	READY	RUN
25	RUN	READY	BLCK
26	EXIT	RUN	READY
27		BLCK	RUN
28		READY	RUN
29		READY	RUN
30		RUN	BLCK
31		EXIT	RUN
32			EXIT
33			

Fig 1 – Round Robin (RR)



Instante	proc1	proc2	proc3
1	NEW		
2	RUN	NEW	
3	RUN	READY	
4	RUN	READY	NEW
5	BLCK	RUN	READY
6	READY	RUN	READY
7	READY	RUN	READY
8	READY	RUN	READY
9	READY	BLCK	RUN
10	READY	BLCK	RUN
11	RUN	READY	BLCK
12	RUN	READY	READY
13	BLCK	RUN	READY
14	BLCK	RUN	READY
15	READY	RUN	READY
16	READY	RUN	READY
17	READY	BLCK	RUN
18	READY	READY	RUN
19	READY	READY	RUN
20	READY	READY	RUN
21	READY	READY	RUN
22	READY	READY	RUN
23	RUN	READY	BLCK
24	RUN	READY	READY
25	RUN	READY	READY
26	RUN	READY	READY
27	EXIT	RUN	READY
28		EXIT	RUN
29			RUN
30			RUN
31			BLCK
32			RUN
33			EXIT
34			

Fig 2 – First Come First Served (FCFS)