Infraestrutura de Software
Implementação AV1

Atenção:

- **Respostas submetidas além do horário e dia limite estabelecido (8:00 am do dia 11.04) serão descartadas. Pessoas com laudo terão uma extensão do prazo, encerrando-se as 8:00 am do dia 15.04;**
- **O arquivo de implementação (o nome obrigatório é `real.c`) e `Makefile` deverão ser salvos dentro de um diretório cujo nome será formado pelas iniciais do seu e-mail em minúsculas. Exemplo: considerando que o seu e-mail é est@cesar.school, o diretório será `est` e qualquer diretório enviado fora desse padrão será automaticamente descartado. Se tal diretório apresentar qualquer outro arquivo além do `real.c` e `Makefile`, a submissão receberá a nota 0;**
- **O diretório será compactado em um arquivo `.tar`, e este deve ser disponibilizado para o professor via Form do Google Classroom. O nome do arquivo será formado pelas iniciais do seu e-mail em minúsculas. Exemplo: considerando que o seu e-mail é est@cesar.school, o arquivo a ser entregue será `est.tar` . Qualquer submissão enviada fora desse padrão será associada a nota 0;**
- **Serão apenas aceitos arquivos submetidos via Form do Google Classroom;**
- **Identificada a cópia de qualquer questão, seja com relação a outra/o aluna/o ou de alguma outra fonte, será associada a implementação a nota 0;**
- **Submissões que apresentarem erro na compilação feita por `Makefile` através do comando `make` via terminal como descrito a seguir, serão associadas a nota 0.**

Escalonamento em tempo real

Nesse projeto, dois algoritmos para simulação de sistemas de tempo real devem ser implementados, *rate-monotonic* (`rate`) e *earliest-deadline-first* (`edf`), em que o tempo total de simulação e as tarefas periódicas a serem escalonadas serão descritas a partir de um arquivo de entrada com a seguinte formatação:

```
[TOTAL TIME]
[TASK NAME 1] [PERIOD 1] [CPU BURST 1]
[TASK NAME 2] [PERIOD 2] [CPU BURST 2]
...
[TASK NAME n] [PERIOD n] [CPU BURST n]
```

, em que `[TOTAL TIME]` é o tempo total de simulação,`[TASK NAME i]` é o nome da i-ésima tarefa, `[PERIOD i]` é o período da i-ésima tarefa e `[CPU BURST i]` é o tempo que a i-ésima tarefa precisa para ser executada.

O arquivo de saída deverá ser nomeado por `algorithm.out`, em que `algorithm` o método de escalonamento utilizado (`rate` ou `edf`), e deve seguir rigorosamente o modelo disponibilizado na atividade do Classroom. Atenção nos espaços em branco e na ausência de uma linha extra no final do arquivo.

Algumas especificações

1. O programa deve ser implementado em C e ser executável exclusivamente em sistemas Linux, Unix ou macOS, com a compilação feita por `Makefile`. O comando `make` irá receber como parâmetro o algoritmo a ser compilado (`rate` ou `edf`), com nome do arquivo executável de acordo com tal algoritmo. Por exemplo:

   ```
   make rate
   ```

   irá compilar para o *rate-monotonic* e deve produzir o arquivo executável `rate`. Enquanto que:

   ```
   make edf
   ```

   irá compilar para o *earliest-deadline-first* e deve produzir o arquivo executável `edf`.

2. A execução deve ser realizada utilizando comando `./rate [namefile]` ou `./edf [namefile]`, em que o `[namefile]` corresponde ao nome do arquivo com o tempo total de simulação e a lista das tarefas periódicas.

3. O arquivo com a lista de tarefas não terá uma linha em branco ao final, e ao final de cada linha não haverá um ou mais espaços em branco, entre `[TASK NAME i]` e `[PERIOD i]` haverá um único espaço em branco, assim como entre `[PERIOD i]` e `[CPU BURST i]` (ver exemplo disponibilizado no *Classroom* da atividade).

4. Considere que as tarefas descritas no arquivo de entrada chegaram simultaneamente ao sistema no tempo 0 (zero) para serem executadas.

5. Para validação da implementação, será utilizado o compilador `gcc 13.2.0`, com o arquivo de saída seguindo rigorosamente a formatação e texto apresentados.

6. Em caso de empate pelo critério de prioridade, a tarefa que será executada é definida pelo algoritmo FCFS.

7. Em caso de uma tarefa A acabar no exato tempo em que uma tarefa B de maior prioridade chegar, a tarefa A é considerada como finalizada.

8. No caso de um processo A chegar no exato momento do tempo final se encerrar, deve-se contar esse processo como KILLED e não precisa indicar que ele executou 0 (zero) no tempo final.
9. O Burst e Período serão sempre números inteiros.

10. No caso de uma task é perdida no tempo T, porém nesse tempo T ela não executa nenhuma unidade, não é preciso mostrar que ela foi perdida utilizando 0 unidades.

11. O deadline para fazer o cálculo de prioridade das tarefas do `rate` e `edf` são o período dado.

Segue um exemplo.
Dado que um arquivo de entrada hipotético de nome `input.txt` apresente o seguinte conteúdo:

```
165
T1 50 25
T2 80 35
```

, indicando que a execução terá `165` unidades de tempo (u.t.) e duas tarefas, que chegam simultaneamente ao sistema para serem executadas: a primeira de nome `T1`, a ser chamada a cada `50 u.t.` e com o tempo de execução de `25 u.t.`, e uma segunda com nome `T2`, a ser chamada a cada `80 u.t.` e com o tempo de execução de `35 u.t.`. O *deadline* para cada tarefa requer que tal tarefa seja completada antes de uma nova instância da mesma. A execução

```
./rate input.txt
```

deverá criar o arquivo `rate.out` com o seguinte conteúdo:

```
EXECUTION BY RATE
[T1] for 25 units - F
[T2] for 25 units - H
[T1] for 25 units - F
```

```
[T2] for 5 units - L
[T2] for 20 units - H
[T1] for 25 units - F
[T2] for 15 units - F
idle for 10 units
[T1] for 15 units - K

LOST DEADLINES
[T1] 0
[T2] 1

COMPLETE EXECUTION
[T1] 3
[T2] 1

KILLED
[T1] 1
[T2] 1
```

Segue a descrição de cada linha:

```
EXECUTION BY RATE       - o algoritmo utilizado
[T1] for 25 units - F   - T1 executou por 25 u.t. e finalizou (F)
[T2] for 25 units - H   - T2 executou por 25 u.t. e ficou em espera (H)
[T1] for 25 units - F   - T1 executou por 25 u.t. e finalizou (F)
[T2] for 5 units - L    - T2 executou por 5 u.t. mas atingiu o deadline e foi perdido
[T2] for 20 units - H   - T2 executou por 20 u.t. e ficou em espera (H)
[T1] for 25 units - F   - T1 executou por 25 u.t. e finalizou (F)
[T2] for 15 units - F   - T2 executou por 15 u.t. que restava e finalizou (F)
idle for 10 units       - não há nenhuma processo a ser executado
[T1] for 15 units - K   - T1 executou por 15 u.t. e foi removido (K) pois o tempo expirou
                        - linha em branco
LOST DEADLINES          - indicar para cada tarefa, quantas atingiram o deadline
[T1] 0                  - nenhuma tarefa T1 atingiu o deadline
[T2] 1                  - uma tarefa T2 atingiu o deadline
                        - linha em branco
COMPLETE EXECUTION      - quantas tarefas de cada tipo finalizaram durante o tempo total
[T1] 3                  - três tarefas T1 finalizaram
[T2] 1                  - uma tarefa T2 finalizou
                        - linha em branco
KILLED                  - quantos processos foram removidos pois o tempo expirou
[T1] 1                  - uma tarefa T1 foi removida
[T2] 1                  - uma tarefa T2 foi removida
```

, com o arquivo de saída seguindo rigorosamente a formatação e texto apresentados.

ATENÇÃO: NÃO DEVE EXISTIR ESPAÇO EM BRANCO AO FINAL DE CADA LINHA, HÁ UM ÚNICO ESPAÇO EM BRANCO SEPARANDO OS ELEMENTOS DE CADA LINHA, NÃO HÁ ESPAÇO EM BRANCO NAS LINHAS EM BRANCO, E NÃO HÁ UMA LINHA EM BRANCO AO FINAL DO ARQUIVO.

Caso a execução fosse

```
./edf input.txt
```

a saída cria o arquivo `edf.out` e a primeira linha dever ser:
`EXECUTION BY EDF`

Pontuação
Teste 1 - 20%
Teste 2 - 20%
Teste 3 - 30%
Teste 4 - 30%