Projeto Prático 2
Sistemas Operacionais

1 Instruções

- Esta atividade prática pode ser realizada em grupo de até 3 alunos.
- A entrega deve ser feita através do **moodle** até o dia **22/05/2023**.
- A apresentação acontecerá no dia **22/05/2023**.
- Atividades iguais ficarão com **zero**.
- O trabalho deve ser realizando considerando as informações descritas neste documento.
  - **Não serão fornecidas outras entradas ou saídas de teste**.
- Os itens que serão avaliados são:
  - **(1,0)** O atendimento as especificações e instruções aqui descritas para o funcionamento correto da versão **sequencial** do programa;
  - **(5,5)** O atendimento as especificações e instruções aqui descritas para o funcionamento correto da versão **paralela** do programa;
  - **(3,5)** Tabela de desempenho, analisando e apresentando os resultados obtidos durante a apresentação.
- A apresentação tem caráter eliminatório, ou seja, **se não souber responder** as perguntas da apresentação seu trabalho inteiro poderá ser **zerado**.

2 Descrição

Leia atentamente as instruções do que deve ser feito.

2.1 Jogo da vida

John Horton Conway inventou o jogo da vida em 1970, quando ainda era um jovem matemático na Universidade de Cambridge, Inglaterra. Através de Martin Gardner e de sua coluna na revista Scientific American (outubro/1970 e janeiro/1971), o jogo foi popularizado entre os programadores do mundo inteiro e passou a ser um dos maiores consumidores de ciclos de CPU nos anos 70.

2.2 Do tabuleiro e das regras

Este jogo simula a evolução de uma sociedade de organismos vivos, onde o objetivo é dada a configuração inicial de uma população, simular o movimento desta população ao longo de várias gerações, de acordo com certas regras para nascimento, sobrevivência e morte de cada indivíduo.

Neste universo bi-dimensional e discreto, o tempo flui em passos discretos. A situação do universo em um dado instante é chamada de geração. Uma geração define como será a próxima e assim por diante, de acordo com algumas regras, numa evolução determinística. As regras do jogo da vida que determinam a evolução de uma geração para a sua sucessora são simples e locais, isto é, a situação seguinte de uma célula só depende dela e de suas oito células vizinhas.

A evolução das figuras no universo do jogo da vida lembra o crescimento de uma colônia de bactérias. Isto pode ser utilizado como uma analogia para facilitar a memorização das regras do jogo:

- Se a célula está viva e tem menos de dois vizinhos, ela morre de solidão. Se ela tem mais de três vizinhos, ela morre por problemas devidos à superpopulação.
- Uma célula morta rodeada por três células vivas resultará em uma célula viva na próxima geração.
- Uma célula viva, adjacente a duas ou três células vivas, permanece viva.

Um aspecto importante a considerar é que todas as células vão de geração a geração ao mesmo tempo.

2.3 O papel do jogador

Enquanto nos outros jogos o "jogador" tem participação importante no desenrolar do jogo, isto não acontece no jogo da vida. Aqui o jogador simplesmente define uma configuração inicial e daí, mecanicamente, o jogo se desenvolve sozinho.

2.4 Objetivo

Deverá ser criado um algoritmo Java ou Python para simular a evolução de uma população para uma certa quantidade de gerações. O algoritmo deve explorar a capacidade de processamento paralelo que os computadores possuem. Ou seja, o algoritmo precisa ser paralelo e utilizar threads para o seu processamento.

2.5 Entrada

A entrada será através de um arquivo, composto pelos seguintes itens:

- Quantidade de linhas da matriz M $\times$ M, em que M é a sua quantidade de linhas (matriz quadrada).
- M linhas com M caracteres em cada linha, onde cada caractere pode ser 0 ou 1 (a própria matriz).

A Figura 1 ilustra um exemplo de um arquivo de entrada válido. A primeira linha mostra a quantidade de linhas, e as demais linhas são os valores da matriz de entrada (população inicial). Conforme o exemplo da Figura 1, a matriz possui 10 linhas (informação presente na primeira linha do arquivo).

```
10
1111000000
1100010010
0110111100
1011001000
0000001001
1110111110
0101011101
1001110000
1010111010
0111101001
```

Figura 1: Exemplo de arquivo de entrada

2.6 Funcionamento

Seu algoritmo deve calcular as gerações seguintes, até atingir a geração solicitada através da linha de comando. Para a sua execução devem ser informados o arquivo de entrada e a quantidade de gerações para a população.

Para utilizar processamento paralelo, o programa deve executar com 1 (execução sequencial), 2 e 4 threads. Além disso, deve-se medir o tempo de processamento de cada execução.

A Tabela 1 ilustra todos os testes que deverão ser executados.

Tabela 1: Execuções a serem medidas

Tamanho da matriz	**Número de Gerações**	**Quantidade de Threads**	**Tempo (seg)**
1000×1000	1000	sequencial	
		2	
		4	
1000×1000	2000	sequencial	
		2	
		4	
1000×1000	3000	sequencial	
		2	
		4	
2000×2000	1000	sequencial	
		2	
		4	
2000×2000	2000	sequencial	
		2	
		4	
2000×2000	3000	sequencial	
		2	
		4	
3000×3000	1000	sequencial	
		2	
		4	
3000×3000	2000	sequencial	
		2	
		4	
3000×3000	3000	sequencial	
		2	
		4	

2.7 Saída

A saída do algoritmo é um arquivo contendo a configuração da população após todas as gerações terem ocorrido. **Não imprima nada na saída padrão**. O arquivo deverá conter somente a geração final. A Figura 2 mostra um exemplo de arquivo de saída (com M igual a 10), sendo que ele contém M linhas e M colunas. Cada célula contém 0 ou 1, de acordo com o valor obtido na última geração.

Para cada execução, deve-se armazenar a matriz de saída em um arquivo. A saída será comparada com a saída esperada para validar o algoritmo.

```
0000000000
0000111000
0001111000
0011000100
0100000000
1100000111
1000000000
0100000000
0000011000
0000011000
```

Figura 2: Exemplo de Arquivo de Saída

Seu programa pode gerar um arquivo csv contendo o resultado das análises de comparação de tempo ou pode ser entregue um arquivo pdf contendo as tabelas com a análise de resultados.