**Trabalho 2**

Simulador de Tráfego em Malha Viária

# Objetivo do Trabalho

Desenvolver um simulador de tráfego. Neste simulador há veículos que se movem pelas vias de uma malha viária. A malha é informada através de arquivo. O trabalho deve ser realizado **em duplas.**

# Especificação dos Veículos

1. Cada veículo deve ser um *thread*.
2. O veículo se movimenta pela malha, uma posição por vez, respeitando o sentido de fluxo da pista.

O veículo só pode se mover caso a posição à frente esteja livre.

1. Ao se deparar com um cruzamento:
   1. Deve escolher, aleatoriamente, uma das vias de saída do cruzamento para seguir viagem. A escolha deve ser feita antes do veículo ingressar no cruzamento.
   2. 4 possibilidades
   3. A escolha deve ser feita ANTES de entrar no cruzamento. Não pode mais voltar atrás na decisão.
2. Só deve se mover pelo cruzamento se todas as posições por onde vai passar estiverem totalmente livres (exclusão mútua).
   1. O carro só entra no cruzamento quando todas as posições estão livres (não pode esperar no cruzamento).
   2. OBS: pode haver mais carros no cruzamento, mas não podem estar no caminho que o carro precisa seguir.
   3. Caso haja 4 veículos e cada um queira ir para a direita, nenhum vai interferir na trajetória do outro. (ponto de atenção no trabalho
3. Não deve bloquear o cruzamento de outros veículos (ficar parado no cruzamento).
4. Novos veículos são inseridos nos pontos de entrada da malha (ver especificação da malha)
5. Ao atingir um ponto de saída (ver especificação da malha), o veículo deve ser encerrado.
6. Veículos possuem velocidades diferentes (tempo de *sleep* da *thread* a cada passo).
   1. Não existe ultrapassagem, se um veículo mais rápido estiver atrás de um mais lento, ambos vão acabar indo “Na mesma velocidade”.
7. Veículos que seguem em vias ou direções diferentes nos cruzamentos devem se movimentar de maneira independente e simultânea (concorrência).
8. Não utilizar *ExecutorService* para rodar as *threads.*
   1. Tem limite de Threads e começa a reutilizar elas a partir de uma certa quantidade, o que quebra a condição A.

# Especificação da Malha Viária

1. Deve ser carregada de um arquivo texto.
2. Nas duas primeiras linhas estão a quantidade de linhas e colunas da malha, respectivamente.
3. As próximas linhas especificam o tipo de cada segmento (célula) da malha:

|  |  |
| --- | --- |
| **Valor (int)** | **Tipo de segmento**  **(movimentos permitidos)** |
| 0 | Nada (célula não usada pela malha) |
| 1 | Estrada Cima |
| 2 | Estrada Direita |
| 3 | Estrada Baixo |
| 4 | Estrada Esquerda |
| 5 | Cruzamento Cima |
| 6 | Cruzamento Direita |
| 7 | Cruzamento Baixo |
| 8 | Cruzamento Esquerda |
| 9 | Cruzamento Cima e Direita |
| 10 | Cruzamento Cima e Esquerda |
| 11 | Cruzamento Direita e Baixo |
| 12 | Cruzamento Baixo e Esquerda |

1. Características gerais das vias:
   1. São sempre horizontais ou verticais (não haverá vias em diagonal).
   2. São de mão dupla, ou seja, possuem duas pistas. iii) Nas bordas, só haverá vias perpendiculares.
   3. Entre vias paralelas, haverá sempre ao menos uma linha ou coluna em branco (ou seja, não haverá vias “grudadas” umas nas outras).
      1. A malha deve ser lida da perspectiva do Observador (de cima), não do carro!
2. Identificação de pontos de entrada e de saída de veículos:
3. **Entrada**: posição inicial da pista que está em uma das bordas da malha
4. **Saída**: posição final da pista que está em uma das bordas da malha.

Gráfico

O conteúdo gerado por IA pode estar incorreto.

Exemplo de malha e arquivo:

1. A malha e os veículos devem ser visualizados em uma interface gráfica.
   1. Tanto faz o estilo, apenas tem que ver as vias e os veículos andando.
2. O sistema **deve** possuir opções para:
3. limitar quantidade de veículos: usuário informar a quantidade máxima de veículos que poderão estar circulando **simultaneamente** na malha. O sistema fica inserindo veículos até atingir esta quantidade. Assim que veículos saírem da malha, novos veículos devem ser inseridos para manter a quantidade máxima circulando simultaneamente.
4. intervalo de inserção (em milissegundos): permite ao usuário informar um intervalo de inserção de veículos (de quanto em quanto tempo um novo veículo é inserido na malha).
   1. iniciar simulação: fica inserindo veículos, que se movimentam na malha respeitando a quantidade descrita anteriormente no item (i).
   2. encerrar inserção: para de inserir e aguarda os veículos existentes sairem da malha.
   3. encerrar simulação:para de inserir e *encerra imediatamente* todos os veículos.
5. O sistema deve suportar dois mecanismos de exclusão mútua (semáforos **e** monitores), devendo existir uma opção para o usuário escolher qual destes mecanismos será utilizado.
   1. Colocar um ComboBox ou um Radio Button para escolher qual tecnologia

# CRITÉRIOS DE AVALIAÇÃO

* Funcionalidades do sistema (deve atender todas as especificações acima).
* Ausência de erros. A movimentação dos veículos deve estar livre de impasses e sobreposições.

o Na apresentação, depuração com *breakpoints* deve ser utilizada para evidenciar que o comportamento dos veículos e a exclusão mútua garantem a ausência de impasses e sobreposições. Detalhes sobre a depuração e *breakpoints* serão fornecidos em aula.

* Adoção de técnicas de engenharia de software quando necessário (“ausência de gambiarras”).
* Elaboração e uso de slides organizados, claros e com linguagem correta.
* Exposição do projeto do sistema (no mínimo diagrama de classes), das técnicas/padrões de projeto utilizados, dificuldades encontradas e soluções adotadas.
* Linguagem verbal adequada, termos e expressões relacionados ao assunto.
* Aproveitamento do tempo (conforme tempo máximo estabelecido e divulgado previamente).
* Demonstração do software desenvolvido, com execução passo-a-passo (modo *debug*) evidenciando ausência de impasses e sobreposições de veículos.
* No dia da apresentação, outras malhas poderão ser usadas pelo professor para testar o sistema.
* Equipes que não apresentarem o trabalho terão nota zerada.

# ATENÇÃO

As notas de implementação e apresentação serão individualizadas por aluno, em função da clareza, coerência, domínio do assunto e do software, e respostas às perguntas.

# DATAS E ENTREGAS

## Até 20/9

* Criar repositório no Github e adicionar o professor como colaborador (usuário: [santos-fernando)](https://github.com/santos-fernando)
* O arquivo README.MD deve conter o nome de todos os integrantes da equipe.
* Perda de 3.0 pontos por semana de atraso na criação e compartilhamento do repositório.

**16/10 até 19:00h** *push* final para correção

* Nesta data/horário os repositórios serão clonados para correção final. Portanto a equipe deve ter feito *push* de todo o conteúdo (apresentação, documentos, códigos fonte, etc) no repositório até esta data/horário
* Perda de 1.0 ponto por dia de atraso na entrega do trabalho (conforme data do *Github*).

## 16/10 a 23/10 a partir das 19:00h

• Apresentações dos trabalhos. As apresentações serão por equipes voluntárias, ou sorteio caso não haja equipes voluntárias.