# Engarrafamento

Trabalho


## 1 Objetivos

Construir uma simulação de tráfego em dois cruzamentos por um período de 24 horas virtuais (relógio da simulação).

Vocês não terão que realizar uma simulação gráfica, mas teremos alguns parâmetros a mais que vocês terão de incluir na simulação, como o tempo

de chegada de novos carros de forma a existir a possibilidade de ocorrer "retrobloqueio"de um cruzamento em função de acúmulo de carros no trecho após o sinal. Você não precisa fazer uma interface gráfica, no entanto se construir a visualização poderá receber pontos de bonificação (0-3).

# 2 Requisitos

## 2.1 Pistas

1. Representar cada pista por uma fila
2. O sistema possui um vetor de suas filas
3. Cada fila contem um vetor das filas/pistas de saída e uma velocidade
4. Cada fila/pista possui uma velocidade
5. Carros saem de uma fila para ir para outra com a velocidade da fila destino
6. Cada fila/pista tem um tamanho fixo em metros e comporta um número limitado de carros.
7. Quando uma pista enche, a entrada daquela pista é bloqueada e o semáforo é marcado.
8. Se o próximo carro de uma pista estiver programado para entrar em uma pista cheia, ele não entra e bloqueia a pista.
9. Cada pista tem uma variável randômica com distribuição uniforme dividida em faixas de valores que modela para qual de suas pistas eferentes um carro vai ir.
10. Algumas filas são sumidouros e carros que nela entram, são eliminados após a percorrerem.
11. Algumas filas são fontes e “recebem” carros a intervalos randômicos dentro de uma faixa de tempos com média e faixa de valores definida pelo professor

## 2.2 Semáforo

1. O sistema possui um vetor dos semáforos
2. Cada semáforo possui um vetor das filas que fazem parte dele, dividindo-as em eferentes (saída) e aferentes (entrada).
3. Cada semáforo associa ao vetor de "pistas eferentes"(as suas filas de saída) um vetor de probabilidades de um carro dobrar em cada uma dessas pistas eferentes.

## 2.3 Veículos

1. Cada veículo possui um tamanho
2. O tamanho do veículo é dado pelo seu tamanho mais 1 metro à frente e 2 metros atrás.

## 2.4 Relógio

1. O sistema possui uma fila de eventos que representa o "relógio do sistema".
2. São eventos:
   - chegada de um novo carro
   - mudança de estado do semáforo
   - chegada de carro ao ao semáforo
   - troca de pista
3. O relógio é uma fila ordenada por hora de ocorrência do evento.

# 3 Geração de Valores Aleatórios

Este é um trabalho de aula e por isso devemos fazer alguns compromissos para que o tamanho do trabalho fique dentro de limites factíveis. Podemos imaginar que uma distribuição realista para o intervalo de tempo de chegada de carros é uma variável aleatória com distribuição normal.

Então não vamos complicar onde não há necessidade. O importante é aprender a programar uma simulação, e não obter dados absolutamente realistas. Para facilitar vamos então utilizar variáveis com distribuição uniforme.

### 3.1 Toques de programação para geração de valores aleatórios em um intervalo:

- Gerar valores aleatórios com distribuição uniforme no intervalo 0 a 1, utilize as funções *rand* e *srand*.
- Lembre-se de inicializar o sempre gerador de números aleatórios, antes de usar, para garantir de que sejam usados valores diferentes em cada simulação.
- Para gerar um número entre 0 e 1, você precisa dividir o valor gerado por "RAND_MAX", definido em "stdlib.h".
- Para gerar um número aleatório com distribuição uniforme em um intervalo, pegue este resultado, multiplique pelo tamanho do intervalo e adicione a valor do limite inferior do intervalo.

  **Por exemplo:** para gerar um valor aleatório de tempos de chegada entre 8 e 12 segundo (10 +/- 2), você pega o tamanho do intervalo, que é de 8 a 12 inclusive, logo 5 valores, e multiplica o seu numero aleatório de 0 a 1 por 5. A seguir adiciona o limite inferior do intervalo, 8, ao resultado. Para que você possa usar este numero ainda falta truncar, pegando só a parte inteira. Para isto basta fazer um *typecasting*: inteiro = (int) real;

## 4 Resultado final do programa

Ao finalizar a execução da simulação, o programa deve apresentar os seguintes dados:

1. Número total de carros que entraram no sistema;
2. Número total de carros que saíram do sistema; e
3. Número de carro atualmente nas pistas e filas.

## 5 O que deve ser entregue

- Código fonte do programa em C (bem identado, comentado e organizado (diretórios específicos e múltiplos arquivos)), de acordo com a abordagem de implementação de TDA visto em aula.
- Makefile para a construção do projeto.

- E um arquivo de LEIAME explicando o projeto.
- Deve seguir as **diretivas gerais de entrega** de trabalho da disciplina.

# 6 Entrega

A entrega deve ser via **moodle** no item de entrega relacionado ao trabalho de fila. Não será aceito trabalho entregue atrasado.

Os seguintes tópicos também serão levados em consideração na avaliação:

1. Uso de variável global: -5 descontados da nota;
2. Vazamento de memória: 1 vazamento = -20% e >1 vazamento = -40%;
3. Não compila: Nota 0;
4. Expirou o tempo de 60 segundos: Nota 0; e
5. Erro de execução (fim de programa anormal): 0.

# 7 Comentários Gerais

- Clareza, identação e comentários no programa também vão ser considerados na avaliação;
- O trabalho é individual (grupo de UM aluno);
- Trabalhos copiados (e FONTE) terão nota zero;
- Trabalhos entregue em atraso serão aceitos, todavia a nota atribuída ao trabalho será zero;
- Evite discussões inócuas com o professor em tentar postergar a data de entrega do referido trabalho.