Universidade Federal do Espírito Santo - Departamento de Informática

Estruturas de Dados I INF09292

1º Trabalho Prático de 2017/02

Prof.ª Patrícia Dockhorn Costa, Email: pdcosta@inf.ufes.br

Data de Entrega: 07/11/2017 - Trabalho em dupla.

Este trabalho tem como objetivo praticar o uso de tipos abstratos de dados e estruturas de dados do tipo Lista.

Regras Importantes

- Não é tolerado plágio. Trabalhos copiados serão penalizados com zero.
- A data de entrega é inadiável. Para cada dia de atraso, é retirado um ponto da nota do trabalho.

Material a entregar

- Relatório: Documentação do trabalho (utilizando as normas ABNT UFES), que deve conter:
    - Capa do Projeto: deve conter os seguintes itens: Título, Autoria e Data.
    - Introdução: descrição do problema a ser resolvido e visão geral sobre o funcionamento do programa (em termos de módulos, arquivos, etc.).
    - Metodologia: descrição da implementação do programa. Devem ser detalhadas as estruturas de dados utilizadas (preferencialmente com **diagramas ilustrativos**), o funcionamento das principais funções utilizadas incluindo pré e pós condições, o formato de entrada e saída de dados, bem como decisões tomadas relativas aos casos e detalhes de especificação que porventura estejam omissos no enunciado. **Modularize o seu programa usando a técnica de tipos abstratos de dados**, como discutido em aula.
    - Conclusão: comentários gerais sobre o trabalho e as principais dificuldades encontradas em sua implementação.
    - Referências bibliográficas: bibliografia utilizada para o desenvolvimento do trabalho, incluindo sites da Internet se for o caso.
- Por email (**edufes20172@gmail.com**):
    - O assunto da mensagem deve ser ed20172:trab1:<aluno1><aluno2>
        - Por exemplo: ed20172:trab1:<joao><maria>
    - Documentação do trabalho (em formato PDF).

- Todos os arquivos .c e .h criados (exigido código muito bem documentado!).
- O makefile.

Simulador Aedes

O Aedes aegypti é um mosquito vetor de diversas doenças, entre elas Febre Amarela, Zika, Dengue e Chicungunya. É um problema atual no contexto das cidades com clima tropical devido ao alto poder de disseminar doenças e de produzir rapidamente descendentes do mosquito.

A transmissão de doenças é feita através da picada da fêmea do mosquito que possui um determinado vírus. A pessoa que é picada contrai a doença e, caso outro Aedes aegypti pique a mesma pessoa, o mosquito vira um novo vetor com o vírus da doença. Além disso, o mosquito passa o vírus entre as gerações, ou seja, cada descendente é um vetor com o vírus desde o nascimento. A fêmea do Aedes aegypti necessita do sangue humano para fazer a oviposição em água. Os ovos, então, eclodem e viram larvas que se alimentam da matéria orgânica presente na água. Em seguida, as larvas viram pupas, que por sua vez dão origem a novos descendentes.

O principal modo para o combate das doenças transmissíveis pelo Aedes aegypti é a prevenção do nascimento de mais mosquitos: não deixar água parada em utensílios e evitar o acúmulo de água da chuva. A prevenção é aliada aos agentes de saúde municipais, responsáveis por educar a população através de visitas domiciliares preventivas. Também, as prefeituras de diversas cidades brasileiras costumam utilizar o fumacê, para diminuir uma população local de mosquitos. As armadilhas para o Aedes aegypti são importantes indicadores da população de mosquitos em uma determinada região e uma outra importante forma de combate.

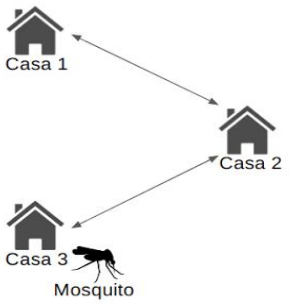
O Simulador Aedes, proposto por este trabalho, é um simulador de comportamento do Aedes aegypti no ambiente urbano. O objetivo do simulador é prever um possível cenário de epidemia em uma região, melhorando o entendimento do hábito de vida do mosquito e permitindo o planejamento de ações mais efetivas.

O simulador deste trabalho representa o ambiente de uma região. As casas são ligadas de forma a representar a possibilidade de vôo do mosquito de uma casa à outra. Os mosquitos inseridos podem mudar sua posição ao longo do tempo. Os agentes atuam de forma a escolher uma casa e acabar com os mosquitos. Por questões didáticas, a simulação foi simplificada. Em uma modelagem mais realista, outros elementos, tais como seres humanos, comportamentos reais do mosquito, vírus e clima, poderiam ser incluídos para torná-la mais acurada.

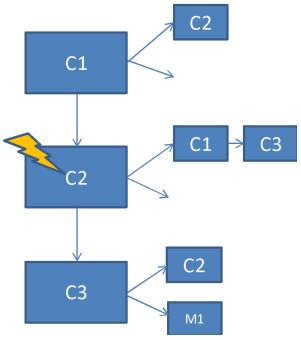
1ª Parte: Implementação das Estruturas de Dados

Para a simulação do cenário, é necessária a implementação das estruturas de dados conforme a "entrada", indicando as entidades que farão parte do processo. Faça a leitura do arquivo texto de entrada (o nome do arquivo será “entrada.txt”, e estará na mesma pasta da execução do arquivo) que contém os comandos e transforme os elementos do cenário em uma lista de listas encadeadas.

Considere o cenário demonstrado abaixo, no qual são ilustradas 3 casas: casas Casa1(C1) e Casa2(C2) estão “conectadas”, bem como as casas Casa2(C2) e Casa3(C3). O mosquito (M1) está localizado na Casa 3 (C3). Considere que o agente esteja atuando na casa C2.



Este cenário poderia, por exemplo, gerar uma lista de listas, como a da figura a seguir:



Note que, de acordo com esta figura, cada casa pode ter uma lista de vizinhança, bem como uma lista de mosquitos. Verifica-se que: C1 é vizinha de C2 (e vice-versa); C2 é vizinha de C1 e C3; C3 é vizinha de C2; C1 e C2 não possuem mosquitos (listas vazias); C3 possui 1 mosquito (M1). O raio está representando a atuação o agente naquela casa.

Ainda considerando este exemplo, o arquivo de entrada a seguir, ilustra a sequência de comandos que foram usados para gerar as estruturas de dados correspondentes (note que os comandos podem estar em qualquer ordem, com exceção do **iniciasimulacao** e **FIM** que estão sempre no final do arquivo, e haverá apenas um AGENTE_ATUA e um MOSQUITO_BOTA por arquivo de entrada).

Exemplo de arquivo de entrada:

```
AGENTE_ATUA 2
```

```
MOSQUITO_BOTA 5

inserecasa C1

inserecasa C2

inserecasa C3

ligacasas C1 C2

ligacasas C2 C3

inseremosquito C3

insereagente C2

iniciasimulacao 50

FIM
```

Especificação dos possíveis comandos do arquivo de entrada:

AGENTE_ATUA <movimentos> : Especifica o número de movimentações dos mosquitos (ao todo) antes de cada chamada da função agente_atua(). O número de movimentos nunca pode ser 0.

MOSQUITO_BOTA <movimento> : Especifica o número de movimentação do mosquito (cada mosquito) antes de cada chamada da função mosquito_bota(). O número de movimentos nunca pode ser 0.

inserecasa <nome_casa> : Insere uma casa de um determinado nome na simulação (ao final da lista de casas).

ligacasas <casa_1> <casa_2> : Insere uma ligação entre uma casa e outra que permite o vôo do mosquito. A ligação é sempre bidirecional (inserções ao final da lista).

inseremosquito <nome_casa> : insere o mosquito em uma determinada casa (ao final da lista). O nome do mosquito deve ser M<numero_do_mosquito>. O número do mosquito é dado pela ordem de inserção, começando em um, por exemplo: o primeiro mosquito a ser inserido tem o nome “M1”, o segundo “M2”, o décimo “M10” e assim por diante (sem as aspas).

insereagente <nome_casa>: insere um agente de combate ao mosquito em determinada casa. Deve haver apenas um agente por simulação.

retiracasa <nome_casa>: retira uma casa da lista de casas, exclui toda a lista de vizinhança, bem como a possível lista de mosquitos (liberando toda a memória corretamente). Não pode ser retirada uma casa que tenha um agente! Deve-se fazer esta verificação.

iniciasimulacao <movimentos> : inicia a simulação limitando os movimentos dos mosquitos (ao todo) a um determinado número.

FIM : finaliza a simulação, gerando o arquivo de log e limpando a memória.

2ª Parte: Implementação da Simulação

Na 1ª parte, o cenário da simulação foi implementado em termos de estruturas de dados estáticas. Nesta nova parte, o cenário deve ser implementado de forma a ganhar vida. Para tanto, três funções (no mínimo) devem ser implementadas: **mosquito_move()**, **agente_atua()** e **mosquito_bota()**.

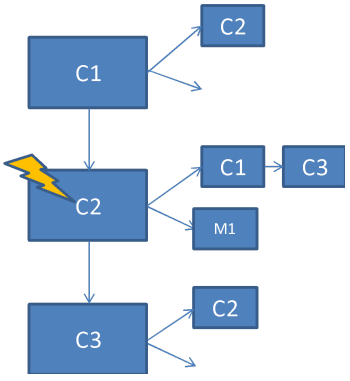
Estas funções devem ser chamadas durante a execução do comando **iniciasimulacao <movimentos>**. O número de movimentos indicados neste comando limita a movimentação ao todo dos mosquitos. Observe que esta é a soma da movimentação de todos os mosquitos da simulação.

As funções são explicadas a seguir:

mosquito_move()

Quando chamada, cada mosquito do cenário deve mudar a sua localização para uma casa que possui ligação com sua atual casa. A escolha da nova casa deve ser baseada na seguinte estratégia: o mosquito deve escolher mover-se para a casa vizinha com o menor número de mosquitos; caso houver empate, escolher a casa com o "menor" nome (utilize a função "strcmp" para definição de "menor"). As movimentações dos mosquitos devem começar pela primeira casa da lista de casas (no exemplo: os mosquitos da casa C1 serão movimentados, seguidos pelos mosquitos da casa C2 e depois da C3). Mosquitos recém movimentados poderão se movimentar novamente na casa para qual se mudaram.

Exemplo: Mosquito escolheu nova casa. A figura seguinte representa este comportamento em comparação ao cenário inicial. Na figura em questão, o mosquito está na casa 2.



Atenção! Cada movimentação de mosquito deve estar documentada no Log, seguindo o formato:

```
Mosquito M1 C3 -> C2
```

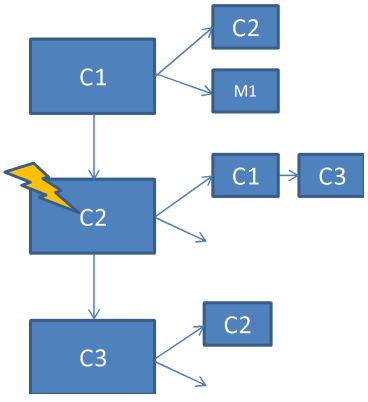
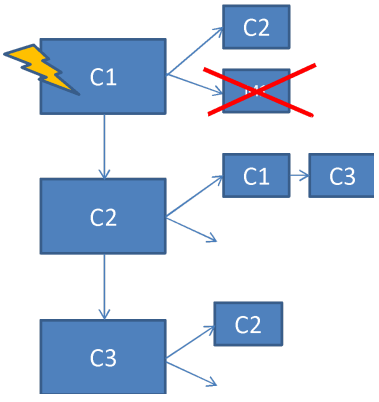
agente_atua()

Quando chamada, o agente vai para uma nova casa e há o combate ao mosquito. Se houver mosquitos na casa onde o agente está atuando, eles devem ser eliminados (memória liberada). Note que o agente é inserido em uma casa inicial por meio do comando **insereagente** no arquivo de entrada.

A função agente_atua() deve ser chamada após um determinado número de movimentações totais dos mosquitos na simulação (este valor é definido no arquivo de entrada pelo comando AGENTE_ATUA). Portanto, no exemplo da entrada acima (AGENTE_ATUA 2), o combate do agente ocorrerá após 2 movimentações de mosquito.

Após o combate, a escolha da casa para a próxima movimentação do agente deve seguir a seguinte estratégia: o agente deve dirigir-se para a casa vizinha com o maior número de mosquitos; caso houver empate, escolher a casa com a “menor” descrição (utilize a função “strcmp” para definição de “menor”).

Exemplo: no caso da entrada acima, que especifica 2 chamadas, se houver 1 mosquito, 2 movimentações deste mosquito devem ser executadas antes da chamada da função agente_atua(). No caso do exemplo, o mosquito estará na casa C1 após 2 movimentações (veja figura a seguir).

Mosquito M1 está na casa C1 após 2 movimentações de mosquito	O agente irá para casa C1 (segundo a estratégica de casa vizinha com maior número de mosquito), e o combate ao mosquito é realizado.
	

Atenção! Cada ação do agente deve estar documentada no Log, seguindo o formato:

```
Agente C2 -> C1

Agente eliminou o mosquito M1!
```

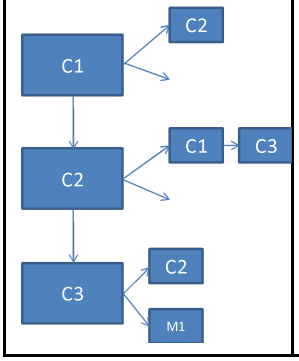
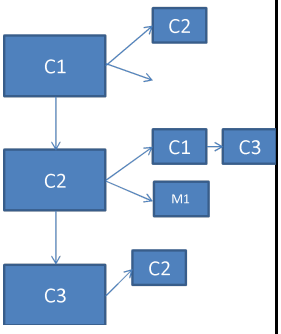
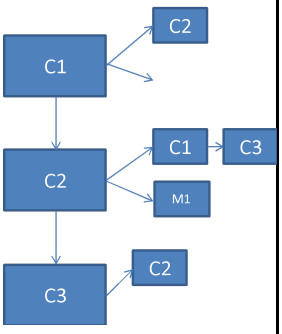
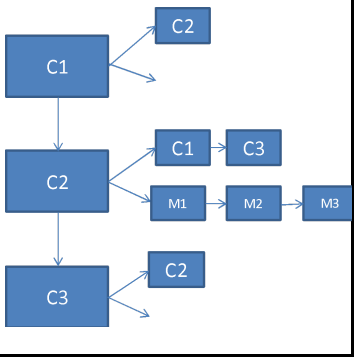
Mosquito_bota()

Quando chamada, cada mosquito cria mais dois descendentes diretos. A função deve ser chamada após um certo número de movimentos de cada mosquito, que é definido no comando MOSQUITO_BOTA, no arquivo de entrada. No caso da entrada indicada, cada mosquito terá descendentes após 5 movimentações (MOSQUITO_BOTA 5).

Os nomes dos mosquitos descendentes devem seguir a mesma regra de nomenclatura do comando inseremosquito, ou seja, se já existem 10 mosquitos na simulação, os próximos mosquitos descendentes de um determinado mosquito serão chamados de “M11” e “M12”.

Exemplo: utilizando a entrada que especifica 5 movimentações, caso um mesmo mosquito possa se movimentar 5 vezes na simulação (e não seja “morto” por nenhum agente), dois outros mosquitos são criados e inseridos na lista de mosquitos do mosquito original (que os criou). A inserção desses mosquitos descendentes é sempre feita ao final da lista.

As figuras a seguir apresentam um curso de 5 movimentações de mosquito (assumindo-se que o agente não está atuando, por questões didáticas).

Cenário Inicial	mosquito_move() vai para C2	mosquito_move() é chamada mais 4 vezes (variando as posições do mosquito entre C1 e C2) – ficando em C2 depois da quinta movimentação	Após a quinta movimentação, há a chamada para mosquito_bota():
			

3ª Parte: Implementação do Log

Ao final da execução da simulação, o programa deve ter gravado informações a respeito da simulação em um arquivo log.txt:

Número de mosquitos iniciais:

Número de mosquitos finais:

Número de movimentos totais dos mosquitos:

Número de erros dos agentes:

Número de acertos dos agentes:

No log.txt devem ser gravadas, além do histórico de movimentações dos mosquitos e ação dos agentes, as situações iniciais e finais das estruturas de dados, como no formato do exemplo a seguir:

```
Inicial:

C1(vizinhos) -> C2
C1(mosquitos)-> null
C2(vizinhos) -> C1 -> C3
C2(mosquitos)-> null
C3(vizinhos) -> C2
C3(mosquitos) -> M1
Agente (C2)

Mosquito M1 C3 -> C2
Mosquito M1 C2 -> C1
Agente C2 -> C1
Agente eliminou M1!

Final:

C1(vizinhos) -> C2
C1(mosquitos)->null
C2(vizinhos) -> C1 -> C3
C2(mosquitos)->null
C3(vizinhos) -> C2
C3(mosquitos) -> null
Agente (C1)
Número de mosquitos iniciais: 1
Número de mosquitos finais: 0
```

```
Número de movimentos totais dos mosquitos: 2
Número de erros dos agentes: 0
Número de acertos dos agentes: 1
```

Observações importantes:

- Caso o mosquito seja combatido antes do número de movimentações, isso é, se não houver mais nenhum mosquito na memória, a execução deve ser terminada e deve haver a escrita do log.txt.
- Observe que o número de movimentações do mosquito necessárias para a execução das funções agente_atua() e mosquito_bota() é contabilizado de forma diferente:
    - A atuação do agente depende da movimentação de todos os mosquitos, sendo especificada pelo comando AGENTE_ATUA, no arquivo de entrada.
    - A procriação do mosquito depende da movimentação daquele mosquito, sendo especificada pelo comando MOSQUITO_BOTA, no arquivo de entrada.
- Em caso de ocorrência das funções mosquito_bota() e agente_atua() ao “mesmo tempo” em uma casa (quando uma movimentação de mosquito chama ambas as funções), a função mosquito_bota() deve ser executada antes de agente_atua().
- Podem existir vários mosquitos, tome cuidado com esse caso.
- Sempre haverá uma ligação de uma casa a outra (não há casas isoladas no sistema). A ligação é de via dupla, isso é, se C1 vai para C2, C2 vai para C1.
- Cada casa contém um nome diferente, que permite identificá-la.
- Os mosquitos contêm um inteiro para o controle da movimentação. **Evite ao máximo usar variáveis globais**!
- As funções propostas são as principais funções, não são as únicas a serem implementadas!

Bom trabalho!!!