Programação Orientada a Objetos

Bacharelado em Ciência da Computação

Prof. Dr. Eduardo Takeo Ueda

2º Trabalho

Simulação de Aeroporto com Java

Data de entrega: 28 de abril de 2014

O paradigma de programação orientada a objetos é muito utilizado para simulação de sistemas que possuem um certo grau de complexidade. Dentro deste contexto o objetivo deste trabalho é implementar um sistema de gerenciamento das pistas de um aeroporto de grande movimento. As estruturas de dados deverão ser econômicas/adequadas e os algoritmos, eficientes.

O bom gerenciamento das pistas de um aeroporto, que são usadas para pousos e decolagens, é fundamental para o funcionamento do mesmo. Os aviões que solicitam pouso têm combustível limitado e não podem circular indefinidamente à espera de autorização para pouso. Similarmente, atrasos nas decolagens são indesejáveis e causam transtornos aos passageiros e grandes prejuízos às companhias aéreas.

O aeroporto que deve ser simulado tem três pistas, numeradas de 1 a 3. As duas primeiras pistas podem ser utilizadas para pousos ou decolagens. A pista 3 é usada apenas para decolagens, a menos que ocorra uma situação de emergência.

A simulação dependerá do valor de alguns parâmetros descritos a seguir. Em cada unidade de tempo, de 0 a K aviões comunicam à torre o desejo de decolar ou pousar. Os aviões – ou, mais precisamente, os voos – são identificados por uma sequência de três letras (identificação da companhia aérea) e quatro números. Além disso, os voos trazem um código de três letras correspondente ao aeroporto de/para onde vão/vêm. Ao contactar a torre, o piloto se identifica por meio do identificador do voo e aeroporto de destino/origem. Caso deseje pousar, ele comunica também quanto tempo tem de combustível (de 1 a C unidades de tempo). Caso deseje decolar, ele comunica qual é a duração aproximada do voo (de 1 a V unidades de tempo). Se um avião que está esperando a autorização de pouso chegar a ter tempo de combustível 0, deve pousar imediatamente.

De tempos em tempos (probabilidade menor que 15%), surge um voo de emergência (transporte de doentes, presidentes, sequestros, etc). Estes voos devem ter passagem livre para decolagem ou aterrissagem, assim que se comunicam com a torre. Cada pista pode manejar uma decolagem ou um pouso em cada unidade de tempo.

Os aviões que estão no ar para pousar permanecem circulando o aeroporto até

que a torre escolha e libere uma pista para o pouso. Estes aviões são atendidos em uma estratégia de "fila", ou seja, os aviões que se comunicaram primeiro com a torre devem ser atendidos antes. Há duas exceções a esta regra: as emergências e os aviões que ficam sem combustível. Nestes dois casos, o avião deve pousar imediatamente, independente dos demais aviões. Similarmente, os aviões que querem decolar são atendidos também na ordem em que se comunicaram com a torre, exceto pelas emergências e os casos de aviões que estejam esperando a liberação da pista por mais de 10% do seu tempo estimado de voo. Os dois últimos casos têm prioridade sobre os demais aviões que esperam para decolar. A cada unidade de tempo, os aviões se comunicam com a torre antes de acontecerem efetivamente decolagens ou pousos.

Você deve implementar uma estratégia que não permita que aviões caiam por falta de combustível. Seu programa deve detectar situações críticas (4 ou mais aviões ficarão sem combustível em um mesmo instante, por exemplo), e poderá escolher alguns desses aviões problemáticos para desviar suas rotas para outros aeroportos da região. Para isso, seu programa deverá manter uma tabela de aeroportos da região, que estejam a uma distância de até 50 unidades de tempo.

Seu programa deverá ser o mais eficiente possível, e sua estratégia para designação e liberação de pistas deverá fazer com que o tempo de espera seja o mínimo possível e o número de aviões desviados seja o menor possível. **Não deverá haver duplicação de informações!** Um avião deve ser representado por apenas um objeto. As filas deverão ter "referências" para o objeto avião. A escolha do tipo de fila a ser utilizado e da implementação de cada fila deve ser feita para que as operações envolvidas na simulação sejam tão eficientes quanto possível. Isso será levado em conta na avaliação do seu trabalho.

Os aviões devem ser de pelo menos 5 companhias aéreas diferentes, e os aeroportos de origem/destino pelo menos 30. Lembrem que certas companhias aéreas operam somente em alguns aeroportos.

A simulação deverá ocorrer durante T unidades de tempo. A saída de seu programa deve indicar com clareza o que está acontecendo a cada momento. Mostre na tela, a cada unidade de tempo:

(a) os aviões que estão nas filas de decolagem e pouso, na ordem em que aparecem na respectiva fila;

(b) o tempo médio de espera para decolagem;

(c) o tempo médio de espera para pouso;

(d) a quantidade média de combustível dos aviões esperando para pousar;

(e) a quantidade média de combustível disponível nos aviões que pousaram;

(f) a quantidade de aviões pousando/decolando em condições de emergência;

(g) e o número de aviões desviados desde o início da simulação.

A saída deve ser fácil de compreender e auto-explicativa. Isso também será levado em consideração na correção do seu trabalho.

A entrada da simulação depende dos parâmetros K, C, V e T, já descritos. Ela deve ser criada usando um gerador de números pseudo-aleatórios para decidir dentre as várias possibilidades: quantos aviões, entre 0 e K, se comunicam com a torre em uma dada unidade de tempo, qual é o seu identificador, quantos destes querem pousar, quantos querem decolar, quanto combustível tem cada avião que comunica que quer pousar, quanto é o tempo estimado de voo de um avião que comunica que quer decolar, etc. Lembre-se de fixar a semente do gerador uma única vez no seu programa, de maneira que seja fácil reproduzir uma mesma entrada novamente durante a fase de depuração do seu programa, e mesmo na correção deste.

O desenvolvimento de interface gráfica **não é obrigatório**, mas para aqueles que desejavem fazer valerá um ***bônus*** de **até** 1 **ponto** na nota do trabalho, dependendo da qualidade do *layout* produzido.

Recomendações importantes

1. O trabalho pode ser feito individualmente ou em grupo de dois alunos, mas lembre-se que **não é permitido** o mesmo grupo do 1º trabalho.

2. O grupo que for composto por membros que já fizeram o 1º trabalho juntos será punido com nota 0 (**ZERO**).

3. Produza um pequeno **relatório** descrevendo os testes que foram feitos e indicando como o professor da disciplina poderá reproduzi-los para conseguir os mesmos resultados.

4. Você deve submeter no BlackBoard um arquivo .rar (ou .zip), contendo o código-fonte da simulação (implementado no ambiente Windows com NetBeans), o relatório (item 3) e um arquivo README (com instruções de como executar seu código e simular o aeroporto); e identificado com a concatenação do seu primeiro nome com seu sobrenome. Por exemplo, meu nome é Eduardo Takeo Ueda, então devo submeter um arquivo chamado EduardoUeda.rar (ou .zip).

5. Se você por acaso julgar que é(são) necessário(s) outro(s) arquivo(s) além dos exigidos neste enunciado, poderá adicionar ao arquivo .rar, desde que justifique no arquivo README porque foi preciso fazer isso.

6. Não esqueça de deixar comentários explicando muito bem seu código-fonte, pois isso será levado em conta na correção do trabalho.

7. Códigos-fonte que forem confirmados como **plágio** receberão nota 0 (**ZERO**) no trabalho.

8. Não serão aceitos trabalhos entregues por e-mail, apenas pelo sistema BlackBoard.

9. Não serão aceitos trabalhos atrasados, ou seja, depois da data limite de entrega.

10. Não deixe para fazer o trabalho na última hora, como na véspera da data de entrega.

Bom trabalho!