Aluno: Mateus Coutinho Ferreira

Simulador Aeroporto

O objetivo deste trabalho foi simular os padrões de aterrissagem e decolagem num aeroporto que recebe até seis aviões por ciclo e tem apenas três pistas, necessitando de uma organização para que as pistas sejam usadas de forma suficientemente eficiente, de forma que poucos aviões caiam e as filas não cresçam excessivamente.

A cada ciclo, de zero a três aviões são adicionados às menores filas de decolagem.
A classe *Queue* é a implementação de uma fila estática circular, uma estrutura de dados que usa um único vetor de tamanho fixo como se estivesse conectado de ponta a ponta. Os objetos dessa classe possuem os campos *items* para o vetor, *start* para o índice do primeiro item, e *size* para a quantidade de itens na fila. Para encontrar a menor fila, usa-se a função *Queue.shortest*, que recebe o vetor de filas de decolagem em *Airport.takeoffs* e retorna a *Queue* com o menor *size*. Para inserir um item na fila, usa-se a função *Queue.enqueue*, que o armazena em *items[(start + size) % items.length*] e incrementa *size*. O operador *modulus* é usado para que, ao alcançar o último índice, o próximo índice seja novamente o primeiro, assim conectando as duas pontas do vetor. O avião a ser inserido é gerado pela função *Plane.createTakeoffPlane*, e tem um campo *id* sendo um número par sequencial (calculado a partir de *takeoffPlaneCount*), *initialFuel* sendo 20 (combustível máximo), e *arrivedAt* sendo o ciclo atual (obtido da função *Airport.getTime*).

De zero a três aviões são adicionados às menores filas de aterrissagem.
Segue uma lógica semelhante à acima, porém o avião gerado pela função *Plane.createLandingPlane* tem o campo *id* sendo um número ímpar sequencial (calculado a partir de landingPlaneCount) e *initialFuel* entre 1 e 20. A menor fila de aterrissagem é obtida passando *Airport.landings* para a função *Queue.shortest*.

Tenta-se aterrissar aviões das maiores filas de aterrissagem nas duas primeiras pistas.
Para encontrar a maior fila, usa-se a função *Queue.longest*, que recebe o vetor de filas de decolagem em *Airport.landings* e retorna a *Queue* com o maior *size*. Para retirar um avião da fila é usada a função *Queue.dequeue*, que retorna o elemento *items[start]* e posteriormente incrementa *start* (usando do operador *modulus* para evitar *overflow*) e decrementa *size*. Caso o avião não tenha caído (combustível maior que 0), registra a ocupação de uma pista por esse avião. Repete o processo até que as duas primeiras pistas estejam ocupadas ou as filas de aterrissagem se esvaziem.

Aterrissa nas pistas restantes os aviões sem reserva de combustível (igual a 1).
Para ver o item no início da fila, usa-se a função *Queue.peek*, que retorna o primeiro item sem o desenfileirar. Para saber o combustível atual do avião, usa-se a função *Plane.getFuel*, que checa pelo *id* se o avião está no ar ou não, e, se sim, retorna o tempo decorrido (*Airport.getTime() - arrivedAt*) subtraído de *initialFuel*. Caso o avião esteja sem reserva de combustível, o desenfileira e registra a ocupação de uma pista por esse avião.

Se ainda há pistas desocupadas, decola os aviões das maiores filas de decolagem.
Segue uma lógica semelhante à aterrissagem não-prioritária. Porém a maior fila de decolagem é obtida passando *Airport.takeoffs* para a função *Queue.longest*.

A cada 5 ciclos imprime estatísticas mais detalhadas.
Mostra o tempo médio de espera para decolagem e aterrissagem, a quantidade de aviões que aterrissaram com prioridade, e o conteúdo das filas.

Para avançar para o próximo ciclo, é necessário apertar a tecla ENTER.