Universidade Federal de Campina Grande
Centro de Engenharia Eletrica e Informática
Departamento de Sistemas e Computação
Graduação em Ciência da Computação

Exercício sobre heap binária

Objetivo: Praticar a implementação de heap binária**.**

O endereço do sistema de submissão é o **https://les.dsc.ufcg.edu.br:8443/EasyLabCorrection**.

Relembre o conceito de heap binária visto em sala de aula.

Atividades necessárias antes de iniciar o exercício:

1. Crie um projeto no Eplipse chamado LEDA, por exemplo (pode ser qualquer outro nome que lhe convier);
2. Descompacte o arquivo baixado (exceto o PDF) na pasta dos fontes (normalmente **src**) do seu projeto LEDA criado no seu workspace. O arquivo baixado tem a seguinte estrutura:
   - adt
   -- heap
   --- GenericHeap.java (INTERFACE DE UMA HEAP GENÉRICA)
   --- MaxHeap.java (INTERFACE DE UMA MAX HEAP)
   --- MaxHeapImpl.java (IMPLEMENTACAO PARCIAL DE UMA MAX HEAP)
   --- Util.java (CLASSE UTILITÁRIA)
3. No Eclipse, selecione a pasta dos fontes no projeto LEDA e faça um refresh (apertar F5). Note que deve aparecer um pacote adt.heap contendo os arquivos mencionados acima.

Agora voce está pronto para começar a trabalhar nas seguintes atividades:

1. Observe a interface GenericHeap.java. Ela descreve os serviços de uma Heap genérica.
2. Observe a interface MaxHeap.java. Ela representa uma Max Heap.
3. Observe também a existência implementação incompleta MaxHeapImpl. Voce precisa implementar os métodos incompletos. Note que sua implementacao parcial de heap ja contem os atributos: INITIAL_SIZE (o temanho inicial da heap) e INCREASING_FACTOR (fator de crescimento). Sua heap comeca com o tamanho inicial. Quando ela atinge o tamanho maximo entao ela aumenta de tamanho somando-se INCREASING_FACTOR ao tamanho atual da heap. Isso acontece toda vez que a heap enche. Nas remoções não precisa se preocupar com a diminuição do espaço alocado.
4. Concentre-se em implementar conforme descrito na interface e pense em cenários para testar suas implementações. Alguns cenários interessantes são: testar insercoes e verificar se o tamanho da heap muda e se o elemento inserido esta na posicao correta da heap. O memso com a remocao. Nos testes do heapsort, voce pode inclusive adaptar os testes usados pelos algoritmos de ordenacao para testar o heapsort. Procure testar todos os metodos da heap. Voce pode até testar sua heap comparando ela com uma implementação pronta de Java.

Instruções para o envio

Ao terminar o exercício, faça os seguintes passos:

1. Compacte a pasta **adt** que existe nos fontes de seu projeto LEDA (**src**) e retire suas classes de teste desse arquivo compactado. A compactação DEVE ser feita a partir do diretório raiz de seus fontes de forma a preservar a estrutura de pastas que refletem a estrutura dos pacotes (package) Java. Por exemplo, voce deve ter um arquivo compactado NOME_COMPLETO_DO_ALUNO.ZIP com a seguinte estrutura:
   - adt
   -- heap
   --- MaxHeapImpl.java
   Envie esse arquivo com sua solução para o sistema de submissão e verifique que o contador de submissões será alterado.

Observações finais:

- **A interpretação do exercício faz parte da atividade.**
- **A atividade é individual. A conversa entre alunos é proibida.**
- **É proibido coletar códigos prontos e adaptar. Implemente as questões. Isso é para seu aprendizado.**
- **Caso voce observe qualquer problema no sistema de submissão, contacte o professor imediatamente.**
- **Se voce nao compactar o arquivo seguindo a estrutura de diretórios a compilação não terá sucesso e o sistema mostrará isso. Erro de compactação serão de responsabilidade do aluno. O professor não ajudará o aluno nesse item. É só seguir as instruções deste arquivo.**