Tecnologia e Análise em Desenvolvimento de Sistemas

Disciplina de Estrutura de Dados
Prof. Demétrios Coutinho - Data de entrega: 09/01/2014

- Implementação Lista de Prioridade Máximos e Mínimos (*Heap* Min-Max)-

1 Introdução

O objetivo dessa tarefa é implementar um *TADs (Tipo Abastratos de Dados)* - `MinMaxHeap` - **genérico**, ou seja, ser capaz de associar diferentes tipos de dados.

Em ciência da computação, uma ***heap* min-max** é uma fila de prioridade com duas extremidades, implementado como uma versão modificada de um *heap* binário. Como um *heap* binário, um *heap* min-max é representado como uma árvore binária completa. Ao contrário de um *heap* binário, no entanto, os nós desta árvore não obedecem a propriedade min-*heap*, mas eles obedecem a propriedade min-max *heap*. Cada nó no nível impar são valores de mínimo e os nós de nível par são valores de máximos.

2 A tarefa

Segue a descrição da tarefa. A primeira parte consiste em implementar um *Heap* binário de mínimos e máximos. A segunda é fazer um programa com interface gráfica que mostre o passo a passo das modificações do *Heap*.

2.1 Parte 1

A tarefa consiste em criar uma classe **MinMaxHeap** que implementa um *heap* Min-Max genérica capaz de empilhar objetos arbitrários. A tabela 1 descreve os métodos públicos que deverão ser implementados.

Método	Descrição
MinMaxHeap(int size)	Construtor, podendo receber como argumento o tamanho inicial do *Hea*
findMin()	Retorna a chave mínima sem remover do *Heap*.
findMax()	Retorna a chave máxima sem remover do *Heap*.
isEmpty()	Verifica se o *Heap* está vazia.
size()	Retorna o tamanho do *Heap*.
insert(Elemento Genérico x)	Insere uma chave genérica `x` no *Heap*.
deleteMin()	Remove a chave mínima do *Heap*.
deleteMax()	Remove a chave máxima do *Heap*.
Makeclear()	Torna o *heap* vazio.

Tabela 1: Listagem dos métodos da classe **HeapMinMax**. O nome e a descrição de todos os métodos devem ser fielmente seguidos.

A classe **HeapMinMax** deve ser implementada utilizando-se um arranjo unidimensional, com capacidade *default* de 15 elementos. O cliente pode informar um tamanho alternativo passando este valor por parâmetro para o construtor.

Por utilizar um arranjo unidimensional com capacidade de armazenamento pré-determinada, a classe **HeapMinMax** deve ser capaz de **lançar dois tipos de exceções**: *StackOverflow*, gerada quando uma inserção excede a capacidade de armazenamento do *Heap*, e; *StackUnderflow*, gerada quando uma remoção ou consulta é solicitada sobre um *Heap* vazio.

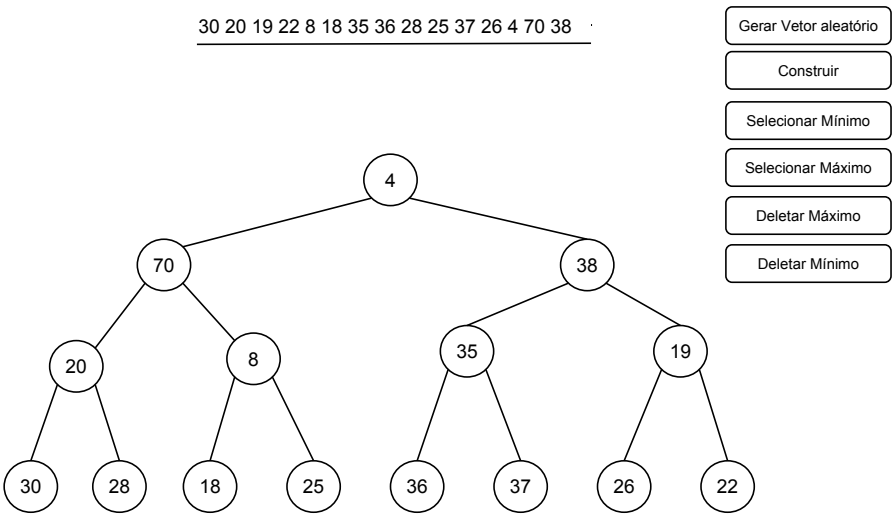
Não esquecer de implementar os métodos privados de subida (`percolateUp(int)`) e de descida (`percolateDown(int)`). Ambos recebem como parâmetro um inteiro referente ao nó que deseja-se subir e descer, respectivamente.

2.2 Parte 2

Fazer um programa que faça a animação do funcionamento de um *Heap* Min-Max. O programa deve receber um vetor com as chaves de inteiros ou gerar um vetor com as chaves aleatórias. A dimensão do vetor deve ser definida pelo usuário. O programa deve conter seguinte ações: Construir o *Heap*, Deletar o mínimo, deletar o máximo, encontrar o mínimo, encontrar o máximo. Para cada ação deve-se aparecer uma animação destacando os nós trocados, inseridos ou removidos. O programa deve mostrar os nós em estrutura de árvores.

A figura 1 é um exemplo de uma possível tela contendo os requisitos solicitados. Perceba que o *Heap* Min-Max esta corretamente construída, pois após fornecer o vetor com dados aleatórios, o usuário selecionou a ação `construir`. Ao selecionar essa ação deve-se aparecer a árvore contendo as chaves na mesma ordem fornecida. E logo em seguida, uma animação destacando os nós selecionados sendo trocados até que o *Heap* Min-Max esteja corretamente construída. Para cada ação deve existir sua respectiva animação.

Figura 1: Exemplo de um programa com animação.



3 Entrega do Trabalho

Independentemente da linguagem utilizada, o código deve ser documentado em alguma ferramenta de documentação, como por exemplo, o *doxygen* para o C++ ou o *javadoc* para Java. A descrição da classe, dos métodos devem ser explicados na documentação em formato HTML. O código, também, deve esta bem comentado. Os métodos e nome de classes deve seguir fielmente esse documento. Nome de variáveis, métodos privados, classes extras privadas, como também a própria documentação deve ser em **inglês**.

Deve ser gerado uma biblioteca, o qual qualquer programador da linguagem desenvolvida possa usar a implementação do *Heap* Min-Max.

A parte 2 também deve ser encontrado na *home page*, sendo um próprio aplicativo web ou um link para download do programa. Lembre que caso seja feito um programa desktop, informe quais os pré-requisitos para que o programa possa seja executado na máquina do usuário.

O trabalho deve ser feito no máximo em duplas, acompanhado de uma *home page*, contendo, no mínimo, como conteúdo:

- A descrição da tarefa.
- Uma breve explicação de uma pilha e fila.
- Disponibilizar a documentação do seu código.
- Disponibilizar o próprio código.
- Disponibilizar a biblioteca, contendo explicações de como instalá-la em qualquer projeto.
- Disponibilizar um arquivo exemplificando o uso da sua biblioteca.

Enviar pelo Edmodo até o dia 09/01/2014 o link para o *home page*. Cada parte concluída constará como 50% da nota. Assim, realizando as duas partes, os componentes terão 100% da nota desse trabalho.