Animador de Algoritmos
Proj2

Informações

- Trabalho em *duplas*;
- Entrega pelo Moodle em <http://trab.dc.unifil.br/moodle/>;

1 Introdução

Neste trabalho, vamos construir um animador de algoritmos para ilustrar a execução das técnicas que aprendemos em sala de aula. Assim como no trabalho anterior, "Desenhista de Arranjos", vamos representar graficamente com barras um arranjo qualquer, entrado pelo usuário. Porém, neste trabalho, vamos utilizar cores para representar estágios de processamento do arranjo, como por exemplo uma comparação entre dois elementos ou troca de posições.

O trabalho é ambicioso mas muitas das funcionalidades já estão pré-implementadas no projeto "Animador de Algoritmos", disponível no Moodle. Por isso, recomendo que utilize-o como ponto de partida. Nele, a interface gráfica já está implementada, bem como a funcionalidade de gravar e reproduzir o processo algorítmico.

De maneira geral, caberá ao aluno implementar:

- O desenhista de arranjos com variações de cores para cada elemento;
- Os algoritmos de busca e ordenação, com algumas anotações para gravação;
- Estender algumas funcionalidades do gravador, para possibilitar e facilitar o correto processo de gravação para cada algoritmo.

2 Roteiro de Trabalho

Para facilitar o seu sucesso neste trabalho, siga o roteiro de atividades seguinte, de preferência na ordem enumerada.

1. No método `paintComponent` da classe `Tocador`, há duas funcionalidades desejadas, mas ainda não implementadas e descritas em comentários, para exibir certas informações na tela. Comece o trabalho implementando-as.

2. Na classe `ArranjoGravado`, implemente o método `pintar`. Ele é muito similar ao que você fez no trabalho "Desenhista de Arranjos", por isso utilize o seu código como base. `ArranjoGravado` possui como atributos um arranjo de valores e outro de cores.

   O arranjo de valores deverá ser desenhado como retângulos coloridos e com borda preta, sendo que a cor de preenchimento de cada retângulo é indicado pela cor de mesma posição no arranjo

de cores. Ou seja, o retângulo que representa o elemento `arranjo[i]` deve ser preenchido com a cor `corIdxs[i]`. Se a cor do elemento do arranjo for `null`, utilize como padrão a cor `Color.BLUE`.

Após implementar corretamente esta funcionalidade, o aplicativo será capaz de exibir corretamente o processo de busca sequencial, portanto utilize-a como teste para saber se está desenhando corretamente cada quadro.

3. Na classe `AlgoritmosAnimados`, escreva um comentário em código para cada linha do método `buscaSequencial`, dando atenção especial às que envolvem uso do objeto `Gravador anim`. O objetivo é que você compreenda como utilizar o gravador para registrar o funcionamento dos algoritmos que você vai implementar a seguir.

4. Na classe `AlgoritmosAnimados`, implemente a ordenação pelo algoritmo da bolha no método homônimo. Após implementá-lo, utilize o `Gravador` para marcar os pontos de gravação do método, que são:

   - Disposição inicial do arranjo;
   - Cada vez que dois elementos são comparados, registre a operação, com o método adequado do `Gravador`;
   - Cada vez que ocorrer uma troca de posições entre elementos, registre-a;
   - Disposição final do arranjo;

5. Na classe `Gravador`, faça a documentação no formato *javadocs* para todos os seus métodos. O objetivo é que você entenda o seu funcionamento, para ser capaz de escrever novos métodos de gravação quando for necessário.

6. Na classe `AlgoritmosAnimados`, crie e implemente o método `ordenarPorSelecao`, com as devidas marcações com o objeto de `Gravação`.

   Após terminar a implementação, é necessário ligá-la na interface gráfica. Para tal, crie uma entrada para ela dentro do `switch` do método `onBtnCarregaPressionado` na classe `Animador`. Para todas as outras implementações subsequentes, será necessário fazer o mesmo.

7. Implemente, em `AlgoritmosAnimados`, o método `buscaBinária`. Os pontos de gravação a serem marcados são:

   - Disposição inicial;
   - Cada vez que o elemento for buscado em uma posição. Neste caso, deverá marcar também, com cores distintas, a posição inicial e final de busca binária do arranjo.
   - Disposição final, com elemento encontrado destacado, caso houver;

   Para poder fazer as marcações de gravação da busca binária, será necessário implementar um novo método de gravação na classe `Gravador`, que receba três índices, e os marque com cores distintas.

8. Implemente o restante dos algoritmos animados:

(a) Ordenação por Inserção: deverá marcar as comparações, os deslocamentos à direita e as inserções.

(b) Ordenação Mergesort: deverá marcar as subdivisões, comparações e intercalação.

(c) Ordenação Quicksort: deverá marcar as subdivisões, comparações e trocas de posição. Marcar distintamente (utilizando outra cor) quando a troca for com o pivô.