UFPI – CCN – DEP. DE COMPUTAÇÃO
ESTRUTURA DE DADOS II Período: 2018.2
Prof. Raimundo Moura

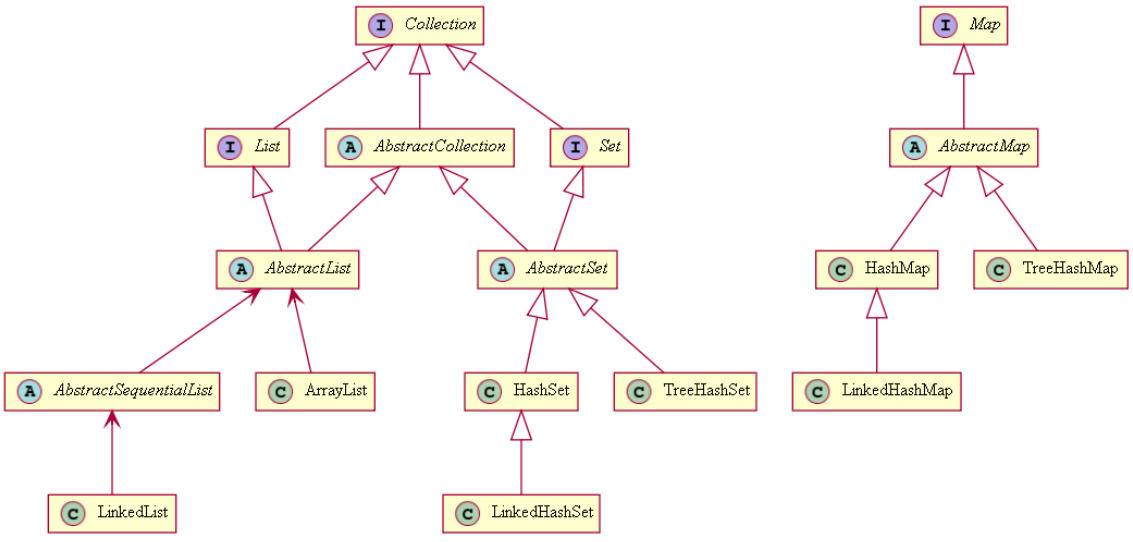
TRABALHO FINAL

INFORMAÇÕES GERAIS:

- Esta atividade versa sobre a *Java Collections API* e compreende 12 (doze) questões.
- O trabalho pode ser realizado em grupo (máximo: 2 pessoas).
- Algumas questões devem ser respondidas através de um texto argumentativo com a justificativa da resposta.
- Nas questões que exigem implementação, descrever os algoritmos utilizados e explicar como executar a aplicação.
- Escrever um relatório técnico (formato pdf) com a solução das questões abaixo.

CONSIDERAÇÕES:

Collections em Java são usadas para armazenar grupos de objetos. A *Collections API* fornece um número de interfaces (incluindo *Collection*, *List*, *Map* e *Set*) para definir uma maneira padrão de usar uma faixa concreta de estrutura de dados. As interfaces e classes da *Collections API* pertencem ao pacote *java.util* e são mostradas na figura abaixo:



OBS: Veja texto de apoio para mais informações sobre a *Collections API Java*. No Java SE 8, as classes *TreeHashSet* e *TreeHashMap* foram alteradas para *TreeSet* e *TreeMap*.

QUESTÕES:

1. Descrever as principais características da classe *LinkedList*, incluindo vantagens e desvantagens. Fazer uma aplicação Java para criar uma tabela de símbolos usando a classe *LinkedList* para armazenar as palavras (não repetidas) do arquivo "*leipzig100k.txt"* que possui cerca de 140.000 palavras. Qual o tempo total de execução desta aplicação?
2. Descrever as principais características da classe *ArrayList*, incluindo vantagens e desvantagens. Fazer uma aplicação Java para criar uma tabela de símbolos usando a classe *ArrayList* para armazenar as palavras (não repetidas) do arquivo "*leipzig100k.txt"* que possui cerca de 140.000 palavras. Qual o tempo total de execução desta aplicação?
3. Descrever as principais características da classe *HashSet*, incluindo vantagens e desvantagens. Fazer uma aplicação Java para criar uma tabela de símbolos usando a classe *HashSet* para armazenar as palavras (não repetidas) do arquivo "*leipzig100k.txt"* que possui cerca de 140.000 palavras. Qual o tempo total de execução desta aplicação?
4. Descrever as principais características da classe *LinkedHashSet*, incluindo vantagens e desvantagens. Fazer uma aplicação Java para criar uma tabela de símbolos usando a classe *LinkedHashSet* para armazenar as palavras (não repetidas) do arquivo "*leipzig100k.txt"* que possui cerca de 140.000 palavras. Qual o tempo total de execução desta aplicação?
5. Descrever as principais características da classe *TreeSet*, incluindo vantagens e desvantagens. Fazer uma aplicação Java para criar uma tabela de símbolos usando a classe *TreeSet* para armazenar as palavras (não repetidas) do arquivo "*leipzig100k.txt"* que possui cerca de 140.000 palavras. Qual o tempo total de execução desta aplicação?
6. Descrever as principais características da classe *HashMap*, incluindo vantagens e desvantagens. Fazer uma aplicação Java para criar uma tabela de símbolos usando a classe *HashMap* para armazenar as palavras (não repetidas) do arquivo "*leipzig100k.txt"* que possui cerca de 140.000 palavras. Qual o tempo total de execução desta aplicação? Os valores associados com as chaves do *Map* podem ser desconsiderados (i.e., valor *null*).
7. Descrever as principais características da classe *LinkedHashMap*, incluindo vantagens e desvantagens. Fazer uma aplicação Java para criar uma tabela de símbolos usando a classe *LinkedHashMap* para armazenar as palavras (não repetidas) do arquivo "*leipzig100k.txt"* que possui cerca de 140.000 palavras. Qual o tempo total de execução desta aplicação? Os valores associados com as chaves do *Map* podem ser desconsiderados (i.e., valor *null*).
8. Descrever as principais características da classe *TreeMap*, incluindo vantagens e desvantagens. Fazer uma aplicação Java para criar uma tabela de símbolos usando a classe *TreeMap* para armazenar as palavras (não repetidas) do arquivo "*leipzig100k.txt"* que possui cerca de 140.000 palavras. Qual o tempo total de execução desta aplicação? Os valores associados com as chaves do *Map* podem ser desconsiderados (i.e., valor *null*).
9. Fazer uma aplicação Java para criar uma tabela de símbolos usando a classe B*Tree* para armazenar as palavras (não repetidas) do arquivo "*leipzig100k.txt"* que

possui cerca de 140.000 palavras. Qual o tempo total de execução desta aplicação? Os valores associados com as chaves podem ser desconsiderados (i.e., valor *null*).

10. Fazer um gráfico com o tempo de execução das aplicações criadas nas questões anteriores, para avaliar a operação de inclusão das estruturas.
11. Repetir as questões de 1 a 9, e após criar a tabela de símbolos, incluir um trecho de código para consultar 10 palavras, a saber: Lisbon, NASA, Kyunghee, Konkuk, Sogang, momentarily, rubella, vaccinations, government, Authorities.
12. Fazer um gráfico com o tempo de execução das aplicações criadas na questão anterior, para avaliar a operação de busca de dados nas estruturas.

OBS: Nas avaliações temporais, procurar ser justo com as execuções, ou seja, usar sempre o mesmo computador e com os mesmos aplicativos e/ou serviços na memória principal.