Trilha Certificação OCP Java SE 8 Programmer II

Java Class Design:

1. Implementar encapsulamento
2. Implementar herança incluindo modificadores de visibilidade e composição
3. Implementar polimorfismo
4. Substitua os métodos hashCode, equals e toString da classe Object
5. Criar e usar classes singleton e classes imutáveis
6. Desenvolver código que use palavras-chave estáticas para inicializar blocos, variáveis, métodos e classes

Advanced Java Class Design:

1. Desenvolver código que use classes e métodos abstratos
2. Desenvolver código que use a palavra-chave “final”
3. Criar classes internas, incluindo classe interna estática, classe local, classe aninhada e classe interna anônima
4. Use tipos enumerados, incluindo métodos e construtores em um tipo de enum
5. Desenvolver código que declare, implemente e / ou estenda interfaces e use a anotação @Override.
6. Criar e usar expressões Lambda

Generics e collections:

1. Crie e use uma classe genérica
2. Criar e usar objetos ArrayList, TreeSet, TreeMap e ArrayDeque
3. Use as interfaces java.util.Comparator e java.lang.Comparable
4. Collections Streams e Filters
5. Iterar usando métodos forEach de Streams e List
6. Descrever a interface do Stream e o pipeline do Stream
7. Filtre uma collection usando expressões lambda
8. Use referências de método com Streams

Lambda Built-in Functional Interfaces:

1. Use as interfaces integradas incluídas no pacote java.util.function, como Predicate, Consumer, Function e Supplier
2. Desenvolver código que use versões primitivas de interfaces funcionais
3. Desenvolver código que use versões binárias de interfaces funcionais
4. Desenvolver código que usa a interface UnaryOperator

Java Stream API:

1. Desenvolva código para extrair dados de um objeto usando os métodos peek () e map (), incluindo versões primitivas do método map ()
2. Pesquise dados usando métodos de pesquisa das classes Stream, incluindo findFirst, findAny, anyMatch, allMatch, noneMatch
3. Desenvolver código que use a classe Optional
4. Desenvolver código que usa métodos de dados de Stream e métodos de cálculo
5. Sort uma Collection usando Stream API
6. Salvar resultados em uma Collection usando o método collect e agrupar / particionar dados usando a classe Collectors
7. Use métodos flatMap () na API Stream

Trilha Certificação OCP Java SE 8 Programmer II

Exceptions and Assertions:

1. Use try-catch e throw
2. Use cláusulas catch, multi-catch e finally
3. Use recursos Autoclose com uma declaração try-with-resources
4. Crie exceções personalizadas e recursos que podem ser fechados automaticamente
5. Teste invariantes usando assertions

Use Java SE 8 Date/Time API:

1. Crie e gerencie eventos baseados em data e hora, incluindo uma combinação de data e hora em um único objeto usando LocalDate, LocalTime, LocalDateTime, Instant, Period e Duration
2. Trabalhe com datas e horas em todos os fusos horários e gerencie as alterações resultantes do horário de verão, incluindo formato de data e valores
3. Definir, criar e gerenciar eventos baseados em data e hora usando Instant, Period, Duration e TemporalUnit

Java I/O Fundamentals:

1. Ler e gravar dados do console
2. Use BufferedReader, BufferedWriter, File, FileReader, FileWriter, FileInputStream, FileOutputStream, ObjectOutputStream, ObjectInputStream e PrintWriter no pacote java.io.

Java File I/O (NIO.2):

1. Use a interface Path para operar em caminhos de arquivos e diretórios
2. Use a classe Files para verificar, ler, excluir, copiar, mover, gerenciar metadados de um arquivo ou diretório
3. Use Stream API com NIO.2

Java Concurrency:

1. Crie threads de trabalho usando Runnable, Callable e use um ExecutorService para executar tarefas simultaneamente
2. Identificar problemas potenciais de threading entre deadlock, starvation, livelock e condições de corrida
3. Use a palavra-chave synchronized e o pacote java.util.concurrent.atomic para controlar a ordem de execução do thread
4. Use coleções e classes java.util.concurrent incluindo CyclicBarrier e CopyOnWriteArrayList
5. Use o Framework de Fork / Join paralela
6. Use Streams paralelos incluindo reduction, decomposition, merging processes, pipelines e performance.
7. .
8. Building Database Applications with JDBC
9. Descrever as interfaces que constituem o núcleo da API JDBC, incluindo as interfaces de Driver, Connection, Statement e ResultSet e sua relação com as implementações do provedor
10. Identifique os componentes necessários para se conectar a um banco de dados usando a classe DriverManager, incluindo o URL JDBC
11. Envie consultas e leia os resultados do banco de dados, incluindo a criação de declarações, o retorno de conjuntos de resultados, a iteração dos resultados e o fechamento adequado de conjuntos de resultados, declarações e conexões

Trilha Certificação OCP Java SE 8 Programmer II

Localization:

1. Leia e defina a localidade usando o objeto Locale
2. Criar e ler um arquivo de Propriedades
3. Construir um pacote de recursos para cada local e carregar um pacote de recursos em um aplicativo