

kod szkolenia: OCPJP8 / PL AA 4d

Przygotowanie do certyfikatu Oracle Java SE 8 Programmer II (1Z0-809)



Przeznaczenie szkolenia

Szkolenie skierowane do osób, którym zależy na przygotowaniu do certyfikatu Oracle Java SE 8 Programmer II (1Z0-809)



Korzyści wynikające z ukończenia szkolenia

Posiadanie kompletnej wiedzy, pozwalającej na zdanie egzaminu Oracle 1Z0-809.



Oczekiwane przygotowanie słuchaczy

Znajomość języka Java 8.



Język szkolenia

- Szkolenie: polski
- Materiały: polski



Czas trwania

4 dni / 28 godzin

Agenda szkolenia

1. Budowa klas w Javie

- Enkapsulacja
- Dziedziczenie z uwzględnieniem modyfikatorów widoczności oraz kompozycji
- Polimorfizm
- Nadpisywanie metod z klasy Object: toString, equals, hashCode
- Tworzenie i użycie klas typu singleton oraz klas niezmiennych (immutable)
- Użycie modyfikatora static do zmiennych, metod, klas oraz bloków inicjalizacyjnych

2. Zaawansowane aspekty budowy klas w Javie

- Użycie klas i metod abstrakcyjnych
- Użycie słowa kluczowego final
- Tworzenie wszystkich rodzajów klas wewnętrznych
- Użycie typów enum
- Deklaracja, implementacja, dziedziczenie interfejsów oraz użycie adnotacji @Override
- Tworzenie i użycie wyrażeń lambda

3. Kolekcje i typy generyczne

- Tworzenie i użycie klas generycznych
- Tworzenie i użycie typów ArrayList, TreeSet, TreeMap, ArrayDeque
- Użycie interfejsów java.util.Comparator oraz java.lang.Comparable
- Przetwarzanie kolekcji za pomocą strumieni i filtrów
- Użycie metod forEach do iterowania po strumieniach i listach
- Interfejs Stream oraz potoki strumieni
- Filtrowanie kolekcji z użyciem wyrażeń lambda
- Zastosowanie referencji do metod ze strumieniami

4. Interfejsy funkcyjne wyrażeń lambda

- Zastosowanie interfejsów z pakietu java.util.function : Predicate, Consumer, Function, Supplier
- Zastosowanie interfejsów funkcyjnych w wersjach do typów prostych
- Użycie dwuargumentowych wersji interfejsów funkcyjnych
- Użycie interfejsu UnaryOperator

5. Java Stream API

- Tworzenie kodu pobierającego dane ze strumieni przy pomocy metod peek() oraz map()
- Wyszukiwanie danych w strumieniach przy pomocy metod findFirst(), findAny(), anyMatch(), allMatch(), noneMatch() klas typu Stream
- Wykorzystanie klasy Optional
- Użycie metod wyliczających i wyszukujących dane z interfejsu Stream
- Sortowanie kolekcji przy pomocy Stream API
- Zapis wyników przetwarzania z użyciem metody collect() oraz podział / grupowanie danych przy pomocy klasy Collectors
- Użycie metody flatMap()

6. Wyjątki i asercje

- Użycie konstrukcji try-catch oraz throw
- Użycie catch, multi-catch oraz finally
- Zastosowanie konstrukcji try-with-resources
- Tworzenie własnych wyjątków oraz zasobów typu AutoCloseable
- Sprawdzanie niezmienników z wykorzystaniem asercji

7. API dot. daty / czasu w Javie SE 8

- Zastosowanie klas LocalDate, LocalTime, LocalDateTime, Instant, Period, Duration
- Obsługa stref czasowych
- Tworzenie i praca z obiektami reprezentującymi odcinki czasu, z wykorzystaniem klas Period, Duration, TemporalUnit

8. Podstawy Java IO

- Odczyt i zapis danych z/do konsoli
- Użycie klas z pakietu java.io: BufferedReader, BufferedWriter, File, FileReader, FileWriter, FileInputStream, FileOutputStream, ObjectOutputStream, ObjectInputStream, oraz PrintWriter

9. Plikowe I/O w Javie (NIO.2)

- Użycie interfejsu Path do operacji na ścieżkach do plików i folderów
- Użycie klasy Files do operacji takich jak odczyt, usunięcie, kopiowanie pliku bądź folderu
- Użycie Stream API z NIO.2

10. Wielowątkowość

- Tworzenie wątków z użyciem Runnable, Callable oraz równoległe wykonywanie zadań z użyciem ExecutorService
- Identyfikacja problemów z wątkami: deadlock, starvation, livelock, race conditions
- Użycie konstrukcji synchronized oraz klas z pakietu java.util.concurrent.atomic do kontroli kolejności wykonywania wątków
- Zastosowanie kolekcji i klas z pakietu java.util.concurrent, z uwzględnieniem CyclicBarrier oraz CopyOnWriteArrayList
- Użycie frameworka Fork/Join
- Zastosowanie strumieni równoległych (redukcja, dekompozycja, łączenie procesów, potoki, wydajność)

11. Tworzenie aplikacji bazodanych z wykorzystaniem JDBC

- Podstawowe interfejsy JDBC API: Driver, Connection, Statement, ResultSet
- Użycie elementów JDBC API niezbędnych do nawiązania połączenia z bazą danych z użyciem klasy DriverManager z uwzględnieniem URL bazy danych
- Wysyłanie zapytań do bazy oraz odczyt wyników, iteracja po wynikach oraz zamykanie obiektów JDBC API

12. Lokalizacja

- Odczyt i ustawienie lokalizacji z użyciem klasy Locale
- Utworzenie i odczyt pliku Properties
- Stworzenie zasobów lokalizacyjnych dla każdej lokalizacji i ich odczyt w aplikacji