<https://jobsearch.fgl-ts.co.jp/contents/Array/231#:~:text=%E3%80%8CJava%20Silver%E3%80%8D%E3%81%AF%E4%B8%AD%E7%B4%9A%E7%A8%8B%E5%BA%A6,%E7%9F%A5%E8%AD%98%E3%81%8C%E5%95%8F%E3%82%8F%E3%82%8C%E3%81%BE%E3%81%99%E3%80%82>

**Oracle認定試験とは？**

現在、Javaの権利はOracle社が有しており、そのOracle社公認で実施されている試験がOracle認定試験です。  
Oracle社は同社製品を扱っているエンジニアの技術認定資格の試験を行っており、数ある認定試験の中の1つとして、Java Silverが存在します。

**Oracle認定試験のレベル分け**

Oracle認定試験のうち、Javaに関する試験は、「Bronze」「Silver」「Gold」の3つのレベルが用意されています。  
ここからは、それぞれの試験の難易度に分けて、代表的な試験の内容や難易度などを紹介します。

**【Oracle認定試験のレベル】**  
・Oracle Certified Java Programmer, Bronze SE 認定資格  
・Oracle Certified Java Programmer, Silver SE 11 認定資格  
・Oracle Certified Java Programmer, Gold SE 11 認定資格

**Oracle Certified Java Programmer, Bronze SE 認定資格**

Oracle認定試験の中で、最も易しいレベルの試験が「Java Bronze」です。  
受験資格もなく、誰もが挑戦できるため、プログラミング初心者やJava学習者の登竜門と言えるでしょう。

**Oracle Certified Java Programmer, Silver SE 11 認定資格**

「Java Silver」は中級程度の難易度の試験です。  
受験資格はなく、Java開発の初心者を対象とした試験で、上級者の指導のもと開発ができるレベルのプログラマーであることが証明できます。  
試験内容は、簡単なJavaプログラムの作成や開発環境についての知識など、開発に関わるスキルや知識が問われます。

**Oracle Certified Java Programmer, Gold SE 11 認定資格**

「Java Gold」はOracle認定試験の中で、最も難易度が高い試験です。  
また、受験資格として「Java Silver」の資格を所持している必要があり、Javaについての深い知識と高いスキルが試される試験です。

**Java Silverの試験概要**

Java Silverの試験について、基本的な概要を紹介します。  
Java Silverの試験は、問題数80問に対して試験時間が180分のため、1問当たり平均2分ほどで解答していかなければなりません。  
試験時間自体が180分と長いため、ある程度の回答スピードや集中力が求められると言えるでしょう。

|  |  |
| --- | --- |
| **出題形式** | 選択問題 |
| **試験時間** | 180 分 |
| **問題数** | 80問 |
| **受験料** | 37,730 円（税込み） |
| **合格基準** | 63 % |

**Java Silverの2023年の試験日程**

Java Silverの試験は、CBT（Computer Based Testing）方式で受験可能です。  
CBT方式とは、全国にあるテストセンターのコンピューターを用いて受験する試験方式のことです。  
CBT方式は、自身の都合の良い期日を選択して受験できます。  
そのため、テストセンターが開いていれば基本的に毎日受験の機会があることが大きなメリットと言えます。

**Java Silverの試験範囲**

Java Silverの試験では、Java開発のスキルや開発環境についての知識が問われます。  
試験範囲は以下の12項目で、公式ホームページには12項目からさらに細かく分かれた出題範囲が記載されています。  
受験する際は、以下12項目の試験範囲について体系的に学習しておくと良いでしょう。

**【Java Silverの試験範囲】**  
・ Javaテクノロジと開発環境についての理解  
・ 簡単なJavaプログラムの作成  
・ Javaの基本データ型と文字列の操作  
・ 演算子と制御構造  
・ 配列の操作  
・ クラスの宣言とインスタンスの使用  
・ メソッドの作成と使用  
・ カプセル化の適用  
・ 継承による実装の再利用  
・ インターフェースによる抽象化  
・ 例外処理  
・ モジュール・システム