2021 için En Çok Java OOPs Mülakat Soruları

tr.tutorialcup.com/interview-questions/java-oops-interview-questions.htm

January 25, 2021

Bu eğitim, en önemli Java OOP mülakat sorularını kapsar.

1. OOP nedir?

OOP'ler, verileri ve davranışı tanımlamak için nesneleri kullanan bir programlama tekniği olan Nesne yönelimli bir programlama sistemidir. Tüm işlevlere ve özelliklere nesneler kullanılarak erişilir. Bu kavram, kodu daha esnek hale getirir ve okunabilirliği artırır.

2. OOP'lerin avantajları nelerdir?

- Kod fazlalığını azaltır
- Kod okunabilirliğini iyileştirir
- Düşük geliştirme maliyeti
- Geliştirilmiş yazılım kalitesi
- Daha hızlı ürün geliştirme
- Kodun yeniden kullanılabilirliği
- Kolay bakım
- Genişletilebilir, bu da mevcut olana kolayca yeni özellikler ekleyebileceğimiz anlamına gelir.

3. Prosedürel programlama ve OOP arasındaki fark nedir?

Prosedürel programlama	Nesne yönelimli programlama
Fonksiyonlara göre	Nesnelere göre
Yürütülen işlevlerin sırasına daha fazla önem	Nesnenin durumları ve davranışları üzerindeki önemi
Kapsüllemeyi desteklemiyor	Kapsüllemeyi destekler
Yukarıdan aşağıya yaklaşımı takip eder	Aşağıdan yukarıya yaklaşımı takip eder
Daha az kod yeniden kullanılabilirliği	Daha iyi kod yeniden kullanılabilirliği
Kodu genişletmek ve değiştirmek daha karmaşıktır	Daha az karmaşık ve kodu genişletmek ve değiştirmek kolay

4. OOP'nin özellikleri nelerdir?

- Soyutlama
- Encapsulation

- miras
- Dernek
- Kompozisyon :
- toplama
- Polimorfizm

5. Soyutlama nedir?

Soyutlama Java'daki temel OOP kavramlarından biridir. Esas olarak ne tür verilerin görünür olması gerektiğini ve neyin gizlenmesi gerektiğini tanımlar. Başka bir deyişle, dahili işlevselliği değil, yalnızca yöntemleri sağlar. Örneğin, bir cep telefonu kullandığımızda, sadece davranışını biliyoruz, ancak nasıl çalıştığına dair iç mekanizmayı bilmiyoruz. Uygulamayı daha güvenli hale getirmek için soyutlamayı kullanabiliriz. Örneğin, net bankacılığa giriş yaptığımızda ve bir kullanıcı adı ve şifre verdiğimizde, şifre aslında gizlidir.

Soyutlamayı bir sınıf, veri veya yöntem için kullanabiliriz. Soyutlamayı uygulamak için kullanıyoruz **arayüzler** ve **soyut sınıflar**.

6. Kapsülleme nedir?

Encapsulation Java'daki verileri ve yöntemini harici müdahaleden bağlayan başka bir OOP kavramıdır. Bu, diğer sınıfların bu verilere veya yöntemlere doğrudan erişmesine veya bunları güncellemesine izin vermediği anlamına gelir. İyi bir örnek, Java sınıfının kendisidir. bağlandığı veriler ve sınıf içindeki işlevleri. Bunu başarmak için değişkenleri veya verileri şu şekilde açıklıyoruz: **özel değişkenler** bu, onlara sınıfın dışında erişemeyeceğimiz anlamına gelir. Bu verilere erişmek ve güncellemek için kullanmalıyız **halka açık olsun ve ayarla** yöntemler. Bu nedenle kapsülleme, hassas verileri kullanıcıdan gizlemeye yardımcı olur. Aynı zamanda **veri gizleme.**

Enkapsülasyonun en iyi bir örneği, **kapsül** (adından da anlaşılacağı gibi) içinde bizim için görünmeyen ilacımız var.

7. Polimorfizm nedir?

Adından da anlaşılacağı gibi polimorfizm, **"birden fazla form".** Bu başka bir önemli Java'da OOP kavramı Aynı yöntemin farklı uygulamalarına sahip olabileceğimiz birçok yazılım uygulamasının geliştirilmesinde yaygın olarak kullanılır.

Java'da nesne yönelimli programlamanın bu özelliğini bir örnekle anlayalım. Örneğin, her bisiklet türünün farklı bir hız kapasitesi olabilir. TVS 50 hız kapasitesi gibi yavaş hızlı bisikletler, Honda Access gibi orta hızlı bisikletlere kıyasla yavaştır ve Pulsar gibi yüksek hızlı bisikletlerle karşılaştırıldığında bu yavaş olabilir. Bu nedenle 3 bisiklet türünün tamamı hıza sahip olsa da uygulama birbirinden farklıdır.

8. Kalıtımı tanımlayın

Kalıtım, alt sınıfın veya alt sınıfın, ana sınıfın veya üst sınıfın tüm özelliklerini ve davranışlarını doğrudan miras alabildiği bir tekniktir.

9. Çoklu miras nedir? Java çoklu kalıtımı destekliyor mu?

Çoklu miras, alt sınıfın birden çok üst sınıfın özelliklerini ve davranışlarını miras aldığı süreçtir. Hayır, Java çoklu devralmayı desteklemez.

10. Java neden çoklu mirası desteklemez?

çoklu miras Java'da bir elmas sorunuyla sonuçlanır. Bu, bir alt sınıfın 2 ebeveyn sınıfını miras alması durumunda, her iki sınıfta da aynı yöntem bulunduğunda belirsizlik olacağı anlamına gelir. Bu nedenle, alt sınıf hangi üst sınıf yönteminin çağrılacağını bilemeyecektir.

11. Statik bağlama ve dinamik bağlama nedir?

Statik bağlama, derleme sırasında çözülür; bu, derleyicinin derleme sırasında hangi yöntemi çağıracağını bildiği anlamına gelir. Örnek: Yöntem aşırı yükleme.

Dinamik bağlama, çalışma zamanı sırasında çözülür; bu, yalnızca yürütme sırasında oluşturulan nesneye göre hangi yöntemin çağrılacağını bildiği anlamına gelir. Örnek: Yöntemi geçersiz kılma.

12. Dernek nedir?

Java'daki ilişki, iki sınıf arasındaki ilişkiyi açıklar. Nesneleri aracılığıyla ilişkiler kurar. Bir ilişki, bire bir, bire çok, çoka bir veya çoktan çoğa ilişkileri temsil edebilir. Sağlar **HAS-A** sınıflar arası ilişki.

Örnekler:

- Bir kişinin sadece 1 ehliyeti olabilir. Bu bire bir ilişkiyi gösteriyor.
- Bir kişinin birden fazla banka hesabı olabilir. Bu bire çok bir ilişkidir.
- Birçok çalışanın tek bir raporlama yöneticisi olabilir. Bu, çoka bir ilişkiyi gösterir.
- Birçok öğrencinin birçok öğretmeni olabilir. Bu çoktan çoğa bir ilişkiyi gösterir.

13. Toplama nedir?

Toplama, aynı zamanda, bir **HAS-A** ilişki ve sadece tek yönlü bir ilişkiyi destekler. Bu, bir sınıfın başka bir sınıfa referansı olduğu anlamına gelir. Örneğin bir kişinin adresi var diyebiliriz. Bunun tersi, yani adresin bir kişi olması anlamlı değildir. Bu nedenle, başka bir ilişki biçimi olan tek yönlü bir ilişki olduğunu söylüyoruz.

14. Kompozisyonu tanımlayın

Kompozisyon, bir **ait veya parçası** ilişkilendirme türü. Başka bir deyişle, bir nesnenin başka bir nesneyi içerdiğini veya ondan oluştuğunu söyleyebiliriz. Bu, güçlü bir ilişkilendirme türü uygular; bu, ana dış nesne yok edilirse iç nesne var olamazsa anlamına gelir.

Örneğin öğrencisiz bir okulumuz olamaz. Bu, içinde öğrenci olmadan bir okulun var olamayacağı anlamına gelir. Böylece okul ve öğrenciler arasında bir kompozisyon oluşturur.

15. Bir sınıf ve nesne tanımlayın

Sınıf, farklı yöntemleri ve özellikleri tek bir birimde gruplayan bir şablondur.

Nesne, bir sınıf içindeki çeşitli durumları ve davranışları temsil eden bir sınıf örneğidir. Tek bir sınıfın birden çok nesnesi olabilir.

16. Bir sınıf örneği oluşturmadan bir temel sınıf yöntemini çağırmak mümkün müdür?

Evet, statik bir yöntem kullanarak veya temel sınıf başka bir alt sınıf tarafından miras alındığında mümkündür

17. Farklı kalıtım türleri nelerdir?

- Tek miras
- Çok düzeyli kalıtım
- Hiyerarşik miras
- Hibrit miras
- Çoklu miras - Java desteklemez

18. Hibrit kalıtım nedir?

Karma kalıtım, çok düzeyli ve hiyerarşik gibi farklı kalıtım türlerinin birleşimidir.

19. Üst sınıf ve alt sınıfı tanımlayın.

Üst sınıf, diğer sınıfların özellikleri ve yöntemleri miras alabileceği bir ana sınıf olarak da bilinir. Örneğin: Shape, karenin temel sınıfıdır.

Alt sınıf, özellikleri ve yöntemleri ana sınıftan devralan bir alt sınıf olarak da bilinir. Örneğin: Bisiklet, Araç'ın bir alt sınıfıdır.

20. Aşırı yükleme ve geçersiz kılma arasında ayrım yapın

Yöntem aşırı yükleme	Yöntem geçersiz kılma
Bu, statik polimorfizmi uygular	Bu dinamik polimorfizmi uygular

Yöntem aşırı yükleme	Yöntem geçersiz kılma
Bu, derleme sırasında olur	Bu, çalışma zamanı sırasında olur
Aynı sınıfta aynı yöntemler mevcuttur	Farklı sınıflarda aynı yöntemler mevcuttur

21. Arayüz nedir?

Bir arabirim, herhangi bir uygulama olmadan yalnızca bir işlevi bildirmeye izin veren bir OOP konseptidir. Arabirim yönteminin işlevselliğini uygulamak için arabirimi uygulayan sınıfın sorumluluğundadır. Bu, farklı sınıfların gereksinime bağlı olarak aynı arayüz yöntemi için farklı uygulamalar sağlamasına izin verir.

22. Yapıcı nedir?

Yapıcı, sınıf adıyla aynı ada sahip özel bir yöntemdir. Bir sınıf, farklı sayıda parametreye sahip birden çok kurucuya sahip olabilir. Bir kurucunun ana kullanımı, nesneleri ve değerleri başlatmaktır.

23. Java'daki farklı kurucu türleri nelerdir?

- Varsayılan kurucu
- Parametreli yapıcı
- Argüman içermeyen yapıcı

24. Finalize etmenin faydası nedir?

Finalize, tüm kaynakları serbest bırakmak ve gerçekleştirmek için kullanılan özel bir nesne yöntemidir. bellek yönetimi.

25. Soyut bir sınıf ile bir arayüz arasındaki fark nedir?

Soyut sınıf	arayüzey
Soyut sınıf hem soyut hem de soyut olmayan yöntemlere sahip olabilir	Arayüz sadece soyut yöntemlere sahip olabilir. Java 8'den varsayılan yöntemleri destekler
Çoklu mirası desteklemiyor	Çoklu mirası destekler
Soyut anahtar kelime kullanır	Arayüz anahtar kelimesini kullanır
Soyut sınıfı miras almak için extends anahtar sözcüğünü kullanır	Arayüzü uygulamak için anahtar kelimeyi uygular
Başka bir Java sınıfını genişletebilir ve ayrıca arabirim uygulayabilir	Yalnızca başka bir arayüzü genişletebilir
Üyeler, özel, korumalı vb. Erişim değiştiricilere sahip olabilir.	Üyeler yalnızca herkese açık olabilir

Soyut sınıf	arayüzey
Statik, statik olmayan, nihai veya nihai olmayan değişkenlere sahip olabilir.	Yalnızca statik ve son değişkenlere sahip olabilir

26. Try-catch bloğu ve son olarak bloğu tanımlayın.

Bir dene-yakala ve en sonunda bloğu uygulamak için kullanılır istisna işleme. Fırlatma istisnasına sahip olan kod, try bloğu içinde tanımlanır ve istisnalar, ilgili catch bloğunda yakalanır.

Son olarak blok, bu blok içindeki ifadelerin her zaman bir istisna olup olmadığına bakılmaksızın yürütülmesini sağlar.

27. Metot aşırı yükleme nedir?

Aynı ada sahip ancak farklı uygulamalara sahip birden fazla yöntem olduğunda, buna yöntem aşırı yüklemesi diyoruz. Yöntem aşırı yüklemesini iki farklı şekilde uygulayabiliriz:

- Farklı sayıda parametre
- Farklı tipte parametreler.

28. Geçersiz kılan bir yöntemin argüman listesini değiştirebilir miyiz?

Hayır, geçersiz kılınan yöntemin geçersiz kılan yöntemle aynı argüman listesine sahip olması gerektiğinden, yapamayız.

29. Bir alt sınıfta geçersiz kılma yönteminin üst sınıf sürümünü nasıl çağırabilirim?

Biz kullanabilirsiniz süper anahtar kelime bir alt sınıfta geçersiz kılma yönteminin üst sınıf sürümünü çağırmak için.

30. Java'da son bir yöntemi geçersiz kılabilir miyiz?

Hayır, son bir yöntemi geçersiz kılamayız çünkü bir yöntem nihai olarak bildirilirse mantıksal uygulamaları değiştiremeyiz. Nihai yöntemin ana fikri, değişikliği önlemektir.

31. Bir arayüz içinde soyut olmayan yöntemler tanımlayabilir miyiz?

Evet, Java 8'den soyut olmayan yöntemleri, durağan ve varsayılan yöntemleri bir arabirim içinde bildirebiliriz. Java 8'den önce, soyut olmayan yöntemlere izin vermez.

32. Ana yöntemi aşırı yükleyebilir miyiz veya geçersiz kılabilir miyiz?

Hayır, statik olduğu için ana yöntemi geçersiz kılamayız. Ancak, onu aşırı yükleyebiliriz.

33. Java'da çoklu kalıtım nasıl elde edilir?

Java, belirsizlik ve birden çok sınıfı genişletmenin karmaşıklığı nedeniyle çoklu kalıtımı doğrudan desteklemez. Ancak, tek bir sınıf aynı anda birden fazla arabirimi uygulayabildiğinden, bunu arabirimler kullanarak başarabiliriz. Arayüz yöntemlerinin uygulanmasını sağlamak uygulama sınıfının sorumluluğunda olduğundan bu durum belirsizliği çözer.

34. Erişim değiştiriciler nelerdir?

Erişim değiştiriciler, değişkenlerin kapsamı veya erişilebilirliği veya bir sınıfın yöntemleri. Aşağıda farklı erişim değiştirici türleri bulunmaktadır:

- özel
- korumalı
- halka açık
- varsayılan

35. Statik ve dinamik bağlama arasındaki farklar nelerdir?

Statik Bağlama	Dinamik Bağlama
Erken bağlama veya derleme zamanı bağlama olarak da adlandırılır	Geç bağlama veya çalışma zamanı bağlama olarak da adlandırılır
Yöntem aşırı yüklemesi statik bağlamadır	Yöntemi geçersiz kılma dinamik bağlamadır
Bağlamayı çözmek için sınıf türünü kullanır	Bağlamayı çözmek için nesne türünü kullanır
Daha hızlı uygulama	Yavaş uygulama

36. Hangi OOP kavramı yalnızca gerekli bilgiyi çağıran işleve açıklar?

Kapsülleme - yalnızca gerekli bilgileri çağıran işleve maruz bırakarak veri gizlemeyi uygular.

37. Java, Operatör aşırı yüklemesini destekliyor mu?

Hayır, Java, operatörün aşırı yüklenmesini desteklemez.

38. Veri soyutlamasına nasıl ulaşabiliriz?

Veri soyutlamasını aşağıdakileri kullanarak başarabiliriz:

- Soyut sınıf
- Soyut yöntem

39. Olağanüstü muameleyi tanımlayın.

Hariç tutma, yürütmeyi aniden kesintiye uğratmaması için istisnaların tespit edilmesine ve ele alınmasına izin veren bir mekanizmadır.

40. Nesne yönelimli bir yaklaşım için 5 tasarım ilkesi nedir?

- S - Tek Sorumluluk ilkesi
- O - Açık-kapalı tasarım prensibi
- L - Liskov ikame ilkesi
- I - Arayüz Ayrıştırma prensibi
- D - Bağımlılık tersine çevirme ilkesi