29.5.2020

HW_7_SONUC_RAPORU

MÜH.TĞM. MUHAMMET YURTSEVEN

Anlaşılmayan : Soru9,Soru12,Soru18,Soru22,Soru24,Soru27,Soru31,Soru37

Soru 1- Cevap C

Soru 2 Cevap D: Bir method hem nihai hem de soyut olarak ilan edilemez. Tüm abstract arayüz yöntemleri dolaylı olarak herkese açıktır ve bu da D Seçeneğini düzeltici yapar.

https://medium.com/@hasandogn/abstract-soyut-s%C4%B1n%C4%B1flar-nedir-85a921536b94

Soru 3- Cevap C

Soru 4 - Cevap A: Kalıtım, nesnelerin yaygın olarak kullanılan özniteliklere ve yöntemlere erişmesine izin verir.

https://medium.com/gokhanyavas/java-oop-miras-alma-inheritance-6-95b0958f7dec

---

Soru 5 - Cevap A

Soru 6 – Cevap B : Java'nın bu özelliklerinden hangisinin henüz geliştirilmemiş olsa bile bir geliştiricinin uygulamalarını başka bir geliştiricinin kodu etrafında oluşturmasına izin verdiğini anlamaktır. Bu sorun için An interface arayüz en iyi seçimdir. İki ekip ortak bir arabirim üzerinde anlaşırsa, bir geliştirici arabirimi kullanan kod yazabilir, bir diğeri ise arabirimi uygulayan kod yazar. Hiçbir takımın arayüzü değiştirmediği varsayıldığında, her iki takım tamamlandığında kod kolayca entegre edilebilir. Bu dönemler için Seçenek B doğru cevaptır.

Soru 7 – Cevap B : https://gelecegiyazanlar.turkcell.com.tr/konu/c-sharp/egitim/c-101/cokbicimlilik-polymorphism

---

Soru 8 – Cevap D : Java bir sınıfın birden fazla sınıfı genişletmesine izin vermese de, bir sınıfın istenilen sayıda arabirimi uygulamasına izin verir. Bu nedenle, çoklu arayüzlere yalnızca arayüzler aracılığıyla izin verilir, bu da Seçenek D'yi doğru cevap yapar.

Soru 9- Cevap

Soru 10- Cevap C: Üst sınıfta tanımlanan onaylanmış özel durumu atmak için geçersiz kılınmış bir yöntem gerekmez.

Soru 11- Cevap C

Soru 12- Cevap

Soru 13- Cevap B: Seçenek B doğru yanıttır çünkü bir arabirim yöntemi statik olarak bildirilebilir.

Soru 14 Cevap C

Soru 15- Cevap B : Bir arayüz başka bir arayüz uygulayamaz ve Seçenek B'yi doğru cevap haline getirir.

Soru 16- Cevap D

Soru 17- Cevap D : An abstract hem soyut hem de somut yöntemler içerebilirken, interface sadece soyut yöntemler içerebilir.

Soru 18- Cevap

Soru 19- Cevap D : Saxophone'da hem Horn hem de Woodwind () yöntemi geçersiz kılınır.Bu nedenle dönüş tipi her ikisiyle de uyumlu olmalıdır. Ne yazık ki, kalıtsal yöntemler aynı zamanda birbirleriyle uyumludur.Kodun derlenmesine izin verecek olan boşluğu doldurmak için kullanılabilecek bir alt sınıf yoktur. Başka bir deyişle, Saxophone sınıfı, play () uygulamasından bağımsız olarak derlenemez ve Seçenek D'yi doğru cevap haline getirir.

Soru 20- Cevap C : implements, Sınıf bir arabirimi uygulayabilir, genişletemez. extends Alternatif olarak, bir sınıf soyut bir sınıfı genişletir.

Soru 21- Cevap A

Soru 22- Cevap

Soru 23- Cevap D: Protected, package-private ve public erişim değiştiricilerinin her biri soyut yöntemlere uygulanabilir.

Soru 24- Cevap

Soru 25- Cevap B : Bir sınıfa referans, belli bir rol olmadan bir üst sınıf referansına atanabilir.

Soru 26- Cevap B : Arayüz değişkenleri dolaylı olarak genel, statik ve final değerlerdir. Değişkenler ne arayüzlerde ne de sınıflarda soyut olarak ilan edilemez.

Soru 27- Cevap

Soru 28- Cevap C : Overloaded olmuş yöntemler aynı adı, ancak farklı bir parametre listesini ve isteğe bağlı olarak farklı bir dönüş türünü paylaşırken, geçersiz kılınmış yöntemler tam olarak aynı adı, parametre listesini ve dönüş türünü paylaşır. Her ikisi için de ortak yönlerden biri, aynı yöntem adını paylaşmaları ve Seçenek C'yi doğru cevap haline getirmeleridir.

Soru 29- Cevap A

Soru 30- Cevap C : Üst sınıftaki bir değişkenle aynı ada sahip bir örnek değişkeni tanımlayan bir sınıf, bir değişkene "hidding" olarak adlandırılırken, statik yöntemle aynı imzayla statik bir yöntemi tanımlayan bir sınıfa bir üst sınıfa "hidding" yöntemi denir.

Soru 31- Cevap

Soru 32- Cevap C

Soru 33- Cevap B: Bir sınıf önemsiz bir şekilde bir superclass referans değişkenine atanabilir, ancak bir alt sınıf referans değişkenine belli bir döküman atanmasını gerektirir. Bu nedenlerle Seçenek B doğrudur.

Soru 34- Cevap C :concrete class, miras alınan tüm soyut yöntemlerin uygulanması için gereken ilk soyut olmayan alt sınıftır.

Soru 35- Cevap D

Soru 36- Cevap B : Her ikisi de varsayılan yöntemler içerebilir.

Soru 37- Cevap

Soru 38- Cevap A: Java, sadece non-static, non-final, and non-private olmayan yöntemler sanal olarak kabul edilir.

Soru 39- Cevap B : Bir interface yalnızca başka bir arabirimi genişletebilirken, sınıf yalnızca diğer sınıfı genişletebilir. Seçenek B doğru cevaptır. Extend,extend

---

Soru 40- Cevap A

Soru 41- Cevap D : `public final void dance()` İşaretli kural dışı durumun kaldırılmasına, daha geniş bir erişim değiştiricisinin uygulanmasına ve son özniteliğin eklenmesine, geçersiz kılınan yöntemler için izin verilir.

Soru 42- Cevap C

Soru 43- Cevap A : bir interface yöntemi final seçenek olamaz ve bu da Seçenek A'yı doğru yanıt haline getirir.

Soru 44- Cevap A

Soru 45- Cevap D : Overridden yöntemlerinde, üst sınıftaki türle tam olarak aynı olmayabilir, kovaryant dönüş türleri olmalıdır. Bu nedenle, Seçenek D doğru cevaptır.

Soru 46- Cevap B : Bir üst sınıf bağımsız değişken yapıcı içermiyorsa, alt sınıf yine de açıkça bir sınıf bildirebilir; Seçenek A'yı doğru yapan super () ile uygun bir üst yapıcı çağırmalıdır. Bir üst sınıf argüman yapıcı içermiyorsa, derleyici varsayılan B bağımsız değişkeni yapıcısını ekleyemeyeceğinden, Seçenek B'yi doğru yapan alt sınıf açıkça bir kurucu bildirmelidir. Üst sınıf, argüman yapıcısına sahip olmadığından, alt sınıfları bulunmadığından C seçeneği yanlıştır. Seçenek C doğru olsaydı, tüm sınıflar bağımsız değişken yapıcısına sahip olmadıkları için bağımsız değişken yapıcılara sahip olmak zorunda kalacaklardır. D seçeneği de yanlıştır. Varsayılan bağımsız değişken yapıcısı, bağımsız değişken yapıcısı olmayan bir sınıfı doğrudan genişleten herhangi bir sınıfa eklenebilir. Bu nedenle, alt sınıftaki hiçbir kurucu gerekli değildir.

Soru 47- Cevap D : object type(Nesne türü), bellekte var olan nesnenin nitelikleriyle ilgilidir; reference type(referans türü), nesnenin arayan tarafından nasıl kullanılabileceğini belirler. Bu dönemler için D Seçeneği doğrudur.

Soru 48- Cevap hiçbirinde kod derlenmedi.

Soru 49- Cevap B: Java'ya varsayılan arayüz yöntemleri eklemek için birincil motivasyon, geriye dönük uyumluluk içindi. Bu yöntemler, geliştiricilerin mevcut sınıflardaki işlevselliği bozmadan bir arabirimin daha yeni bir sürümüyle eski sınıfları güncellemelerine olanak vererek Seçenek B'yi doğru yanıt haline getirir. Seçenek A, saçmalıktır ve düzeltici değildir. C ve D seçenekleri akla yatkındır, ancak her ikisi de sadece statik arayüz yöntemleri ile gerçekleştirilebilir.

Soru 50- Cevap C