1. Cevap D. Bir Java uygulamasına giriş noktası geri dönüş tipi void olan, public erişim belirleyicisine ve String[] tipinde bir argümana sahip olan bir main() metodudur. Argüman adı önemsizdir. İstenirse main metodu final olarak da implemente edilebilir. Bu özellikleri taşımayan A, B ve C seçenekleri yanlıştır.
   public static [final]void main(String[] arguments) şeklinde bir metod imzası olması beklenir.
2. Cevap A. Verilen diyagramda Metal sınfıı ve bu sınıfı extend eden Gold ve Silver sınıflarını göstermektedir. Bu nedenle B seçeneğinde verilen Java'da OOD (Object Oriented Design)'yi gösterir. Gold ve Silver sınıfları Metal sınıfını extend ettiği için Metal sınıfının weight ve color özelliklerini kalıtım yoluyla almaktadır. C seçeneği de doğrudur. Gold sınıfı, Silver sınıfının luster özelliğini kalıtım yoluyla alamaz. D seçeneği de doğrudur. Ancak A seçeneğinde ifade edilen Java'da platform bağımsızlıkla ilgili bir diyagram değildir.
3. Cevap C. Java'da derlenen byte kod dosyasının geçerli uzantısı ".class" tır. Kaynak kodu olan .java uzantılı dosyası terminal satırında derlenerek .class uzantılı byte kod dosyasının üretildiği görülmüştür.
4. Cevap B. Kod 4. Satırdaki Date sınıfının hangi pakete ait olduğu bilinmediğinden dolayı derlenmez. Date sınıfının 1 ve 2 numaralı satırlarda import edilen paketlerde bulunması kodun derlenmesi açısından problem oluşturmaz. A seçeneği yanlıştır. C seçeneğinde kullanılan Date sınıfı paket adıyla çağrıldığı için burada derleme hatası alınmayacağı için bu seçenek de yanlıştır. Kod derlenmeyeceği için D seçeneği doğrudan yanlış olacaktır.
5. Cevap A. B, C ve D seçeneklerinde verilen özellikler geleneksel nesne yönelimli programlamanın özellikleridir. Nesne yönelimli programlamada veriler ve eylemlerin tek bir nesne içinde gruplanması amaçlandığından A seçeneği yanlıştır.
6. Cevap D. Yerel değişkenler, en dar kapsama sahip olduğundan dolayı metot kapsamında limitlidirler. A seçeneğindeki interface değişkenlere, buradan implemente edilen sınıflardan erişilir. B seçeneğindeki sınıf değişkenlerine sınıf kapsamında erişilir. C seçeneğindeki instance değişkenlere tanımlandığı kapsamda erişilir.
7. Cevap B. Seçenekleri denemek için verilen paketlerde yer alan sınıflardan instance üretilirse örneğin B seçeneğinde yer alan java.lang paketinde yer alan Byte sınıfı koda eklendiğinde bir import işlemine gerek olmadığı görülecektir. A seçeneği için eklenen java.util.Date sınıfı import gerektirmektedir. C ve D seçenekleri verilen şekilde paketler Java'da yer alan paketler değildir. A, C ve D seçenekleri yanlıştır.
8. Cevap C. Java'da geçerli bir yorum satırı, tek bir satırı yorum haline getirmek için A seçeneğindeki haliyle, bir satırdan daha uzun yorumlar yapabilmek içinse B ve D seçeneklerinde haliyle yapılır. C seçeneğindeki gibi bir yorum şekli hatalıdır.
9. Cevap D. Bir .java dosyası birden fazla class implementasyonu içerebileceği için A seçeneği yanlıştır. Bir .java dosyasında public erişim belirleyicisi dosya ismini taşıyan sınıfa verilebilir. Aynı anda hem bir sınıf hem de bir interface public olamaz. B seçeneği yanlıştır. C seçeneği, her hangi bir sınıfa ya da interface'e public erişim belirleyici verme zorunluluğu olmadığı için yanlıştır. D seçeneği ise, en fazla bir public sınıf olabileceği için doğrudur.
10. Cevap B. Verilen main() metodu static olmadığı için Java uygulamasının giriş noktası değildir. P1 alanında Robot sınıfından üretilen bir instance ile Robot sınıfına ait sınıf değişkenlerine ve static değişkenlere erişilebilir. Lokal olarak tanımlanan retries değişkeni aynı blokta yer aldığı için buna da erişilebilir. Bu yüzden tüm değişkenlere erişilebilir.
11. Cevap B. Kullanılmayan import ifadeleri kodun derlenmesini etkilemez. A seçeneği yanlış ve dolayısıyla B seçeneği doğrudur. Aynı import ifadesinin birden fazla olması da derlemeyi etkilemeyeceği için C seçeneği de yanlıştır. Programda kullanılan bir sınıfın import edilmemesi kodun derlenmesine engel teşkil edeceği için D seçeneği de yanlıştır.

12. Cevap A. Kod derlenmez, çünkü main() metodu static olduğundan dolayı static değişkenlere erişilir. Kod derlenmeyeceği için diğer seçenekler doğrudan yanlıştır.
13. Cevap D. Java komutu .class uzantılı bytecode'u çalıştırır. I. numaralı ifade yanlıştır. Java nesne yönelimli bir programlama dili olduğu için II. numaralı ifade de yanlıştır. III. numaralı ifade ise, javac komutu ile .java uzantılı kaynak kod dosyası derlendiğinde makine koduna değil bytecode'a derler. Bu nedenle verilen ifadelerin hepsi yanlıştır.
14. Cevap D. A seçeneğindeki bir .java dosyası default package içinde ise import ifadesi ilk satır olabilir. C seçeneğinde görüleceği gibi .java dosyası default package içinde değilse, ilk satır package tanımlaması olacaktır. B seçeneğindeki yorum satırı her zaman için bir .java dosyasında ilk satır olabilir. Ancak D seçeneğindeki değişken tanımlama ifadesi kesinlikle bir .java dosya dosyasında ilk satır olamaz.
15. Cevap C. A seçeneğinde ifade edilen her sınıfın bir paket deklarasyonu içermesi bilgisi yanlıştır. Sınıf default package içindeyse paket deklarasyonu içermez. A seçeneği yanlıştır. B seçeneği, bir paket oluşturmak için package.init dosyasının dizine eklenmesine gerek duyulmacağı için yanlıştır. D seçeneğinde ifade edilen "Bir package içindeki nesne ve metotlara kısıtlı erişim imkansızdır." bilgisi, package-private erişim belirleyicisi ile sağlanarak kısıtlı erişim oluşturabileceği için yanlıştır. C seçeneği, paketlerin var oluş sebebini açıklayan nedendir. C seçeneği doğrudur.
16. Cevap B. Bir java kodunu derlemek için "javac [dosya_ismi].java" komutu, derlenen kodu çalıştırmak için "java [derlenen_dosya_adi]" komutu verilir.
17. Cevap D. Java sınıfında yalnızca sınıf içindeki yöntemlerin örnek değişkenlerine erişebileceği şekilde yapılanması tanımı kapsülleme(encapsulation)'dir.
18. Cevap D. Terminale basılmak istenen toplama ifadesindeki height değişkeni lokal bir değişken olduğu için tanımlı olduğu blok dışında erişilemez. Bu değişkeni if bloğu dışında deklare edip ilk değer ataması yapılsa idi, sonuç 56 olurdu.
19. Cevap A. Bytecode, Uyumlu bir JVM ile herhangi bir bilgisayarda run edilebilir. A seçeneği doğrudur. B seçeneğindeki yaratıldığı bilgisayarla aynı tipteki bir yerde çalışır bilgisi kesinlikle yanlıştır. Bu ifade Java'nın platform bağımsızlığı fikrine muhalefet eder, yanlıştır. Java kodunun derlenmesi sonucu elde edilen bytecode hex byte'lardan oluştuğu için standart metin editörleri ile kolayca okunup değiştirilemezler. C seçeneği de bu nedenle yanlıştır. D seçeneği, bir bytecode'un çalışabilmesi için kaynak kodun bytecode ile beraber taşınması fikrini ifade etmektedir. Kaynak olmadan çalışabileceği için D seçeneği de yanlıştır.
20. Cevap D. Java'da bir ifade noktalı virgül ile sonlandırılır.
21. Cevap C. today değişkeni 20, tolls.tomorrow değişkeni 10 ve tolls.yesterday değişkeni 1 değerine sahiptir ve toplamları olan 31 değeri terminale basılır.
22. Cevap C. A seçeneğindeki deklarasyonda class ifadesi olmadığı için syntax hatası vardır. B seçeneğinde hem double hem int veri tipi ile değişken tanımlanmaya çalışılması nedeniyle yine syntax hatası vardır. D seçeneğindeki ifadede değişkene void geri dönüş tipi ve verilmeye kalkılmış ve static ifadesi erişim belirleyicisinden önce yazılmıştır. C seçeneğindeki ifadede parametresiz bir metot yazılmıştır ve bu satırda hata yoktur.
23. Cevap D. Java'nın geniş bir bilgisayar ve cihaz üzerinde çalışabilmesi özelliği platform bağımsızlığıdır. Bu tanımın, kapsülleme, nesne yönelimi ve kalıtım özellikleri ile ilgisi olmadığı için A, B ve C seçenekleri doğru cevap değildir.
24. Cevap A. JVM (Java Virtual Machine – Java Sanal İşlemcisi) C++ dilinde yazılmış bir programdır. Bir Java programı javac ile derlendikten sonra byte code ismi verilen bir ara sürüm oluşur. Byte code, ana işlem biriminin (CPU – Cental Processing Unit) anlayacağı cinsten komutlar ihtiva etmez, yani klasik Assembler değildir. Java byte code sadece JVM bünyesinde çalışır. JVM, derlenen Java programı için ana işlemci birimi olma görevini üstlenir.

Bu özelliginden dolayı Java programlarını değişik platformlar üzerinde çalıştırmak mümkündür. Her platform için bir JVM sürümü Java programlarını koşturmak için yeterli olacaktır. Bu sebepten dolayı Java "write once, run anywhere – bir kere yaz, her yerde koştur" ünvanına sahiptir.

JVM bünyesinde tüm hafıza otomatik olarak JVM ve Garbage Collector tarafından yönetilir. Programcının bu konuda yapması gereken fazla birşey yoktur. Java Heap dışında işlem yapmak zorunda kalındığında durum farklıdır. Native hafıza alanını JVM bünyesindeki Garbage Collector yönetmez. Programcının C/C++ dillerinde olduğu gibi native hafıza alanını kendisi yönetmesi gerekir.

Verilen seçeneklerden B, C ve D JVM özelliğidir. JVM, A seçeneğinde ifade edilen Java bytecode un kolaylıkla decode/decompile edilmesini önlemez. Bytecode kolayca decompile edilebilir.

25. Cevap B. A seçeneğinde ifade edilen türde bir değişken türü yoktur. Class(static) değişkenlere tüm program kapsamında erişilebilir. Instance değişkenlere, kendi türünden yaratılan bir nesne aracılığı ile erişilir. Yerel değişkenlere ise sadece tanımlı olduğu blokta erişilir.
26. Cevap C. A seçeneğinde, television.actor paketi altındaki sınıflara erişim amaçlı import ifadesi eklenmesine rağmen daha alt bir paket içindeki sınıf eklenmeye çalışılmaktadır. "*" ifadesi, özyineli şekilde sınıfları eklemeyeceği için bu seçenek yanlıştır. B seçeneğinde, movie.director paketi altındaki tüm sınıfların eklenmesi amacıyla bir import ifadesine eklenmesine rağmen John sınıfı movie.directors paketi altında olduğu için bu seçenek de yanlıştır. D seçeneğindeki NewRelease sınıfının eklenmesi için tanımlanması gereken import ifadesi "movie.*" şeklinde olmalıydı. Bu nedenle, verilen import ifadeleri ile eklenebilecek sınıf C seçeneğindeki television.actor.Package sınıfıdır.
27. Cevap D. Bir Java sınıf dosyasındaki ifade sıralaması varsa önce paket imzası, daha sonra import ifadeleri ve sonrasında sınıf deklarasyonudur. Bu nedenle doğru sıralama D seçeneğindeki gibidir.
28. Cevap D. Soruda varsayıldığı şekilde stars paketi ve Blackhole sınıfı oluşturulmuştur. Verilen import ifadelerinden ilk 3 tanesi "import java.lang.*; import java.lang.Object;" olmasa da kod derlenebilir. Blackhole sınıfı stars paketinde yer aldığı için "import stars.*;" ve "import stars.Blackhole;" ifadelerinden birisi yeterlidir. Bu nedenle 3 tane import ifadesi silinebilir.
29. Cevap C. main() metoduna gönderilen parametre deerParams olmasına karşın bir alt satırda terminale basılmak istenen theInput değişkeni herhangi bir yerde tanımlı değildir. Varsayalım ki, kullanıcı deerParams parametresinin 2. İndisindeki değeri basmak istesin. Bu durumda A seçeneği ile "While-tailed deer", B seçeneği ile "3", C seçeneği ile "White-tailed" ve D seçeneği ile boş sonuç üretilirdi. Yani syntax hatası bu şekilde çözülseydi sonuç C seçeneği olacaktı.
30. Cevap B. javac komutu ile bir .java dosyası derlenerek .class uzantılı bytecode üretilir. java komutu ile derlenen bytecode çalıştırılır. Bu bilgiye göre A seçeneğindeki java komutunun .java dosyasını derleyerek .class dosyası üretmesi bilgisi yanlıştır. C ve D seçeneklerinde ise .class dosyasının derlenerek .java dosyası üretilmesinden bahsedilmektedir. Böyle bir yöntem olmadığı için C ve D seçenekleri de yanlıştır.
31. Cevap B. Prosedürel programlama, işlemleri, prosedür veya rutin olarak adlandırılan, alt işlem gruplarına bölerek çözümünün kolaylaştırılması yöntemidir. Prosedürel programlama yöntemi, aslında mantıksal program yürüyüşünü içermese de modüler programlama sistemi bu ilerlemeye yol açmıştır. Tipik prosedürel program dilleri, Algol, Pascal, Modula-2, Ada, serisidir. Javascript gibi birçok programlama dili, prosedürel olarak kullanıma olanak vermektedir. Ancak, Java nesneye yönelik bir programlama dilidir ve A seçeneği yanlıştır.

Nesneye yönelik programlamanın bir yaklaşımı olan çok biçimliliğin (polymorphism) gereği bir metodun/işlevin farklı şekillerde yapılabilmesi özelliği metot aşırı yükleme (method overloading)'dir. Bu nedenle Java için B seçeneği doğrudur.

Operatör aşırı yükleme (operator overloading), "+, -, *, /, %, …" herhangi bir operatörün farklı amaçlar için ve farklı veri tipleri kullanılabilmesini mümkün kılan özelliktir. Bu özellik Java'da bulunmadığı için C seçeneği yanlıştır.

JVM, bellekteki objelerin adresleri ile çalışmaktadır. Doğrudan erişime izin verilmediği için D seçeneği yanlıştır.

32. Cevap D. İlk satırda package imzası olması gerektiği için boşluğa package gelecektir. İkinci satırda sınıf deklarasyonu yapılmaktadır. Banker sınıfı tanımlanması için ikinci boşluğa class ifadesi gelecektir. Üçüncü satırda ise metodun dönüş tipi eksik bırakılmıştır. Buraya null yazılamaz ve return ifadesi yer aldığı için dönüş tipi void olamaz. Bu bilgilere göre D seçeneğindeki ifadelerle eksikler tamamlanacaktır.
33. Cevap A. Verilen kod örneği derlenip sorunsuz çalışacaktır. main() metodunda AirPlane sınıfı kurucu metoduna 10 değeri gönderilir. Bu değeri tutan x değişkenine hemen ilk satırda 4 değeri atanır. Hemen peşinden çalıştırılan fly() metoduna gönderilen 5 değeri distance değişkeninde tutulur. Bu metot içinde ilk satırda 4-2 işlemi uygulanır ve bir boşluk eklenir. Daha sonra distance değişkenindeki 5 değeri basılır. Sonuç olarak A seçeneğinde verilen "2 5" ifadesi terminale basılacaktır.
34. Cevap D. Kalıtımda bir sınıfın başka bir sınıfı extend edebilmesi için aynı pakette yer almasına gerek olmadığı için A seçeneği yanlıştır. B seçeneği, kalıtımda katmanlı yapı olması ve katmanlar arasındaki işlemler dolasıyla zaman ve kaynak tüketimi olabilir. Bu ifade, kalıtımın bir amacı değil sonucudur. Bu nedenle, B seçeneği doğru olarak kabul edilecek seçenek değildir. C seçeneğinde, parent classta yer alan metot imzalarında yapılan değişikliklerin sub-classlardaki overlaoded metotları etkilemesi durumundan bahsedilmektedir. Bu seçenek de, B seçeneği gibi kalıtımın bir amacı değil, sonucudur. C seçeneği de doğru olarak kabul edilmeyecektir. D seçeneği, kalıtımın asıl amacını manifest etmektedir. Kalıtım sayesinde aynı kodlar birden fazla implemente edilmeyerek mükerrer kodların yazılımda bulunmasının önüne geçmektedir. Kalıtımın en önemli amacı budur. Cevap D, seçeneğidir.
35. Cevap A. Java'da yorum satırları "//" karakterleri ile tek satır olarak ya da "/* … */" ifadesi çoklu satırlar halinde yazılabilir. Bu yazım şekline uyan tek seçenek A seçeneğidir. B ve C seçeneklerindeki gibi bir yorum yazım şekli Java'da yer almaz.
36. Cevap B. Java'da bir uygulamaya giriş noktası olan main() metodu public erişim belirleyicisine sahip static bir metottur ve geri dönüş tipi void olmalıdır. Ayrıca içerisine bir String dizi parametresi almalıdır. Bu string dizinin adının ne olduğu önemsizdir ve diziyi belirtmek için kullanılan köşeli parantezler "[]" yerine 3 noktalı gösterim "…" de kullanılabilir. İstenirse final olarak da deklare edilebilir. Bu bilgilere uymayan seçenek B seçeneğindeki gösterimdir. A, C ve D seçenekleri geçerli birer main() metodudur.
37. Cevap B. Line 1 sınıf tanımı dışında yer almaktadır ve burada değişken tanımı yapılamayacağı için burada değişken tanımlanması halinde kod derlenmez. Line 2 ve Line 4'te instance değişkeni olarak tanımlanabilir. Ancak Line 3'te tanımlanacak değişkenin bir erişim belirleyicisi olamaz. Burada sadece String color olarak bir değişken tanımlanabilir.
38. Cevap A. Bir java dosyasında olmazsa olmaz tanımlama, sınıf tanımlamasıdır. Eğer bir java dosyası default package içinde konumlandırılmışsa package tanımı olmayacaktır. Package tanımı gerekli olmadığı için B seçeneği yanlıştır. Java, herhangi bir import ifadesine de gerek duymaz. Tanımlanan sınıf için zaten java.lang paketindeki sınıfları default olarak import etmektedir. Dolayısıyla C seçeneği de yanlıştır. Ayrıca public erişim belirleyicisi gerekli olmadığı için D seçeneği de yanlıştır.

39. Cevap D. Java'da bir kaynak kod dosyası .java uzantılıdır. Bu kodun derlenmesi sonucu elde edilen bytecode ise .class uzantılıdır. .class dosyası kaynak kod dosyası olmadığı için B seçeneği yanlıştır. A seçeneği, .jav uzantılı bir dosya biçimi olmadığından dolayı yanlıştır. C seçeneği de .source uzantılı bir dosya biçimi olmadığından dolayı yanlıştır.
40. Cevap C. Kod 6.satırdaki Math sınıfının java.lang paketinden mi, pocket.complex paketinden mi kullanılacağına karar veremeyeceği için derlenmez. 2 ve 3.satırlardaki import ifadeleri eğer geçerli paketler varsa kodun derlenmesine problem oluşturmaz. Bu nedenle A ve B seçenekleri yanlıştır. Kod derlenmeyeceği içinse D seçeneği yanlıştır.
41. Cevap A. B seçeneğinde verilen Georgette sınıfı dog.puppy paketinde yer aldığı için bu sınıfa erişebilir. C seçeneğinde verilen Webby sınıfı dog paketinde yer aldığı için bu sınıfa erişilebilir. D seçeneğindeki Object sınıfı java.lang paketinde yer alan tüm sınıfların default olarak import edilmesi nedeniyle erişilebilir. Ancak A seçeneğinde yer alan KC sınıfına erişilebilmesi için "import dog.puppy.female.*;" şeklinde bir import ifadesi olmalıydı. "*" sembolojisi sadece mevcut paketin içerisindeki tüm sınıfları import eder, alt paketlerdeki sınıfları import etmez.
42. Cevap B. Nesneye yönelim, birimler üzerinde aksiyonlar ve niteliklerin bir birim olarak ele alan yapısal programlama tekniğidir.

    Kapsülleme: Kapsülleme/Sarmalama (Encapsulation), nesneye yönelik programlamanın temel kavramlarından biridir. Genel tanımıyla kullanıcı tarafından verilerin, sınıfların ve metotların ne kadarının görüntülenebileceği ve değiştirilebileceğinin sınırlarının konulmasını sağlar. Public (herkese açık), private (özel) ve protected (koruma altında) olmak üzere üç adet erişim belirleyicisinden bahsedilebilir.

    Kalıtım: Kalıtım (Inheritance), bir sınıftan başka bir sınıf türetirken aralarında bir ata-çocuk ilişkisi oluşturmayı ve bu sınıflar üzerinde ortak metotlar ve özellikler kullanılmasını sağlayan bir mekanizmadır. Nesneye yönelik programlamanın temel kavramlarından biridir. Hali hazırda var olan sınıfların üzerine başka sınıfların inşa edilmesini sağlar.

    Çok Biçimlilik: Çok biçimlilik (Polymorphism) nesneye yönelik programlamada, programlama dilinin farklı tip verileri ve sınıfları farklı şekilde işleme yeteneğini belirten özelliğidir. Daha belirgin olarak, metotları ve türetilmiş sınıfları yeniden tanımlama yeteneğidir. Örnek olarak şekil diye bir sınıf olsun; çok biçimlilik tekniği sayesinde programcı farklı şekillerin alanlarını farklı metotlar ile belirleyebilir. Şeklin ne olduğu fark etmeksizin program kullanıcıya doğru alanı verecektir.

    Platform Bağımsızlık: Java'nın platform bağımsız olması şu anlama geliyor. Bir kere yaz her yerde çalıştır mantığıdır. Çünkü C,C# gibi diller makinenin işletim sisteminde derlendiği için platform bağımlıdır. Fakat Java, JVM(Java Virtual Machine) üzerinden çalışır. Bu yüzden aynı program nerde derlenirse derlensin, aynı bytecode'u üretir bu yüzden bir kere yazıldıktan sonra istenilen ortamlarda çalıştırmak mümkündür. Buradaki tek sorun hız sorunudur. Çünkü işletim sistemiyle birlikte JVM de bilgisayarın kaynaklarını kullandığı için daha yavaş çalışmaktadır.
43. Cevap A. Soruda verilen şekilde paketler ve sınıflar implemente edildiğinde "import food.vegetables.*;" ya da "import food.vegetables.Broccoli;" ifadesi, Broccoli sınıfından bir nesne üretebilmemiz için gereklidir. Benzer şekilde, Apple sınıfından nesne üretebilmek için de "import food.fruit.*;" ya da "import food.fruit.Apple;" ifadesi eklenmesi gerekir. Date sınıfından bir nesne üretebilmek için de "import java.util.Date;" ya da "import java.util.*;" ifadesi eklenmelidir.
44. Cevap C. Verilen kod parçasında main() metodunda çağrılan değişkenlerden numLock değişkeni static olmadığı için, main() metodunun static olmasından dolayı derlenmez. numLock değişkeni static olarak değiştirip kodu derler ve çalıştırırsak sonuç A seçeneğindeki gibi olacaktır.

45. Cevap D. main() metodu içinde terminale basma komutuna gönderilen değerlerden feet değeri 4, main bloğunda yer alan tracks değeri 15 ve RollerSkates sınıfından üretilen s nesnesinin wheels değeri 1'dir. Bu değerlerin toplamı olan 20 sonucu terminale basılacaktır. Burada karıştırılmaması gereken s nesnesinin tracks değeri çağrılmamaktadır. Bunu çağırmak için s.tracks ifadesi olmalıdır.
46. Cevap B. main() metodundaki kullanım her nerede kullanılırsa kullanılsın, printColor() metoduna gönderilen değer, bu metot içinde tutulmamakta ve metoda girildiğinde Bicycle sınıfından üretilen nesnenin color özelliği purple olarak set edilmekte ve bunu terminale basmaktadır.
47. Cevap C. java komutu paketleri ayırt etmek için nokta (.) karakteri kullanır, kesme(/) karakteri kullanmaz. Bu bilgiye göre, I doğru ve III yanlış olur. javac komutu ile .java uzantılı derlenerek .class bytecode üretilir. Dolayısıyla, java dosyası girdi, class dosyası çıktı olarak düşünülebilir. Bu yüzden doğru cevap I ve II'dir.
48. Cevap D. Verilen kod parçasında herhangi bir söz dizimi hatası olmadığı için derleme hatası olmayacaktır. Ancak main() metodu, parametresiz olduğu için bu java dosyasının derlenmesi sonucu elde edilen .class dosyası bir runnable olarak kullanılamayacağı için derlenen kod rumtime'da hata fırlatacaktır. Bu neden cevap D seçeneğidir.
49. Cevap C. Verilen diyagram Book isimli bir sınıfa aittir. Bu sınıfın public erişim belirleyicisine sahip bir özelliği ve public erişim belirleyicisine sahip bir metodu vardır. UML standartlarına göre, bu metot ve özelliğin public olduğu "+" işaretinden anlaşılmaktadır.
50. Cevap C. Çöp toplayıcı uygulamanın çalışması sırasında herhangi bir anda işleme girebilir. Bu nedenle A seçeneği yanlıştır. B seçeneği, JVM'in uygulamanın herhangi bir nedenle veya iş mantığı ile sonsuz döngüye sokulması durumlarında uygulamanın sonlanacağını garanti edemeyeceği için yanlıştır. D seçeneğinde uyumlu bir JVM olması gerektiğinden bahsedilmediği için eksik bilgi dolayısıyla yanlıştır. C seçeneği doğrudur. Çünkü, JVM bir uygulamaya girebilmek için belirli bir giriş noktası arayacaktır.