1.D

try blogu catch ya da fınally yaparak bitirilmediği için syntax hatası alınıyor.

2.B

Finally en sona konulur, catch'lere girilse de girilmese de finally bloguna girilecektir.

3.D

Javada tüm sınıflar Object sınıfından türer. Throwable sınıfı da bu sınıftan türemekte. Throwable sınıfı iki alt başlıkta toplanır. Bunlar Error ve Exception sınıflarıdır. Buna uygun secenek de D şıkkıdır.

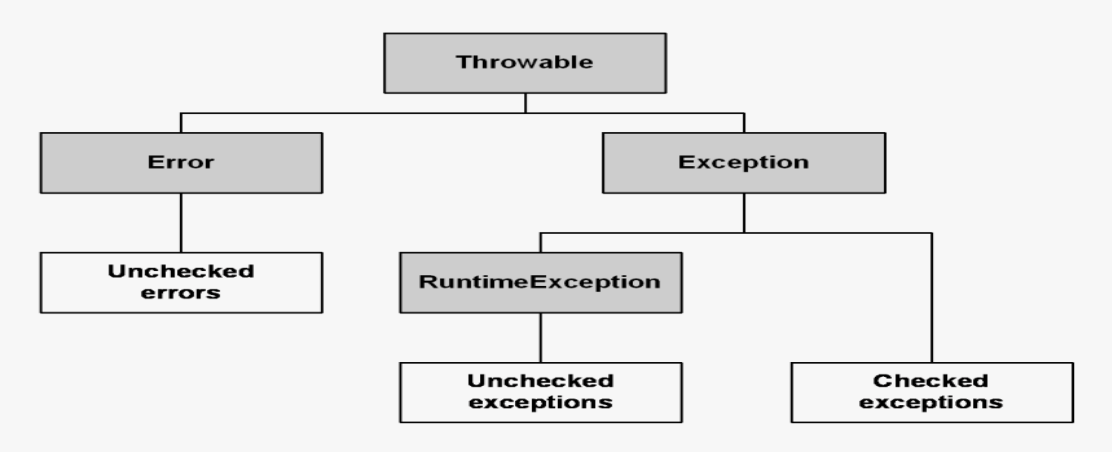
4.A

Exception istisna demektir ve bu durumda aslında beklenen uygulamanın devam edebileceği ama ara sıra yaşansa da sorun olmayacağı yönünde.Error yani hata ise bu durum yaşandığı zaman uygulamanın devam edemeyeceği burada kalması gerekiyor dediğimiz durumlarda kullanılmalıdır.

5.D

Score degıskenı try blogu içinde tanımlanmıştır diğer bloklarda kullanılamaz.

6.B



Unchecked exception İsminden de anlaşılacağı gibi derleyici tarafından kontrol edilemeyen exceptionlardır. Derleyici tarafından kontrol edilemedikleri için kodumuz başarıyla derlenmiş olsa bile çalışma zamanında bu hatalarla karşılaşabiliriz. Unchecked Exceptionlar RuntimeException'ın alt sınıflarıdır. Bunlardan en yaygınları ArithmeticException, NullPointerException, ClassCastException… gibi exceptionlardır.

Checked exception İsminden de anlaşılacağı üzere derleyici tarafından kontrol edilen exceptionlardır. Derleyici tarafından kontrol edildikleri için biz bu exceptionları kodumuzda belirtmek zorundayız. Aksi halde derleme işlemini başarıyla tamamlamak mümkün değildir.Checked Exceptionlar Exception sınıfının alt sınıflarıdır (RuntimeException hariç). En yaygınları ClassNotFoundException, IOException, SQLException… gibi exceptionlardır.

7.A

Java throw anahtar kelimesi, bir istisnayı açıkça atamak için kullanılır . Genel kullanım yapısı throw exception; şeklindedir.Örnek olarak ;

throw new ArithmeticException("/ by zero");

throw new IOException("aygıt hatası");

Yukarıdaki örneklerde new anahtar kelimesi hata nesnesi oluşturmak için kullanılır. Programın yürütme akışı, throw ifadesi yürütüldükten hemen sonra durur ve istisna tipine uyan bir catch ifadesi olup olmadığını görmek için en yakın ek try bloğu kontrol edilir. Eğer bir try cath ifadesi bulamaz ise programı durduracaktır.

Throws Kavramı , bir metot yönetemediği bir istisnaya neden olursa, metot çağırıcılarının kendilerini bu istisnaya karşı koruyabilmeleri için, metodun bu davranışı belirtmesi gerekir. Bunu, metodun deklarasyonuna bir throws cümleciği ekleyerek yapabilirsiniz.

Genel Throws Kavramının Kullanımı:

tip metot_adi (parametreler) throws istisna_listesi {//İçerik}

Özet olarak, Throw try cath blokları içinde hata fırlatmak için kullanılır throws ise metod içinde oluşabilecek hataları barındırır.

8.B

IOException, Exception sınıfının altsınıfı olduğu için önce IOException un önce görülmesi gerekiyor,

9.D

t değişkeni tanımlanmadığı için kod derlenmeyecektir.

10.C

```
package castles;

public class Fortress {
	public void openDrawbridge() throws Exception { // p1
		try {
			throw new Exception("Circle");
		} catch (Exception e) {
			System.out.print("Opening!");
		} finally {
			System.out.print("Walls"); // p2
		}
	}

	public static void main(String[] moat) {
		new Fortress().openDrawbridge(); // p3
		// Unhandled exception type Exception
	}
}
```

11.B

Diğerleri exception sınıfının subclasslarıdır.

12.A

```
packagegame;
publicclassBasketBall {
	publicstaticvoid main(String[] dribble) {
		try {
			System.out.print(1);
			thrownewClassCastException();
		} catch (ArrayIndexOutOfBoundsExceptionex) {
			System.out.print(2);
		} catch (Throwableex) {
			System.out.print(3);
		} finally {
			System.out.print(4);
		}
		System.out.print(5);
	}
}
```

13.A

```
try
{
//hesaplanmak istenen ifade
}
catch
{
//Bir hata türü tespit edilince verilmesi gereken mesaj
}
catch
{
//başka Bir hata türü tespit edilince verilmesi gereken mesaj
}
finally
{
//her durumda çalıştırılacak olan kod parçası
}
```

14.C

```
packageoffice;
import java.io.*;
publicclass Printer {
	publicvoidprint() {
		try {
			thrownewFileNotFoundException();
		} catch (IOExceptionexception) {
			System.out.print("Z");
		} catch (FileNotFoundExceptionenfe) {
	//Unreachablecatchblockfor FileNotFoundException. It is
already handled by the catch block for IOException
			System.out.print("X");
		} finally {
			System.out.print("Y");
		}
	}

	publicstaticvoid main(String... ink) {
		new Printer().print();
	}
}
```

15.C

16.B

Exception fırlatıldıktan sonra uygulama sonlandırılmaz.

17.D

```
packageharbor;
publicclassBoat {
	publicinttravel() throwsException {
		return 4;
	};// j1
	publicstaticvoid main(String... distance) throwsException{
		try {
			System.out.println(newBoat().travel());
		}
		catch(Exception e)(
				System.out.print(8);
				)
		//() parantezkullanıldığıiçinderlenmeyecektir.
	}
}
```

18.??

19.D

Hiçbir importstatement kullanılmadan da Throwable içeren uygulama derlenebilmektedir.

20.C

```
package Q20;

publicclassOxygenextends Element {
	publicintgetSymbol() {
		return 8;
	} // g2

	publicvoidprintData() {
		try {
			System.out.print(getSymbol());
	}catch
	{ // g3   catchyazılmısamafınallygibikullanılmış
		System.out.print("Unabletoread data");
	}
}
}
```

21.B

Checkedexceptionfirlatan bir metodu cagirdigimizdahandle-daclerekuralina uymak zorundayiz. Bu durumda bu kurala uygun sekilde ya throws anahtar kelimesini kullanmali ya da try-catchblogundayakalamamiz gereklidir.

22.B

```
packagecastles;

publicclassCitadel {
	publicvoidopenDrawbridge() throwsRuntimeException { // q1
		try {
			thrownewKnightAttackingException();
		} catch (Exceptione) {
			thrownewClassCastException();
		} finally{
			thrownewCastleUnderSiegeException(); // q2  bu exceptionthrows edilmediği için tanınmıyor
		}
	}

	publicstaticvoid main(String[] moat) {
```

```
                new Citadel().openDrawbridge(); // q3
        }
}
```

23.A

İlk olarak hangi catchbulogu ile eşleşirse o çalıştırılır.

24.C

```
package system;
public class Computer {
        public void compute() throws Exception {
                throw new RuntimeException("Error processing request");
        }

        public static void main(String[] bits) {
                try {
                        new Computer().compute();
                        // Unhandled exception type Exception
                        System.out.print("Ping");
                } catch (NullPointerException e) {
                        System.out.print("Pong");
                        throw e;
                }
        }
```

25.

26.D

StackOverflowException, genellikle çok derin veya sınırsız özyineleme durumunda yürütme yığını taşma hataları için oluşturulur.

27.C ??

Uygulamanın sonlanmayacağını garanti edemez.

28.D

finally bloğu catch'den önce kullanılmış.

29.A

try ifadesi tek başına kullanılamaz , finally ifadesi de birden fazla kullanılamaz. Finally varsa catch kullanılmayabilir.

30.D

31.C

catchbuloğundaki ifade yapılacaktır. En sonunda finally ifadesine gidileceği için buradaki exception da fırlatılacaktır.

33.C ??

Runtimeexception tanımlandığı için o fırlatılacaktır.

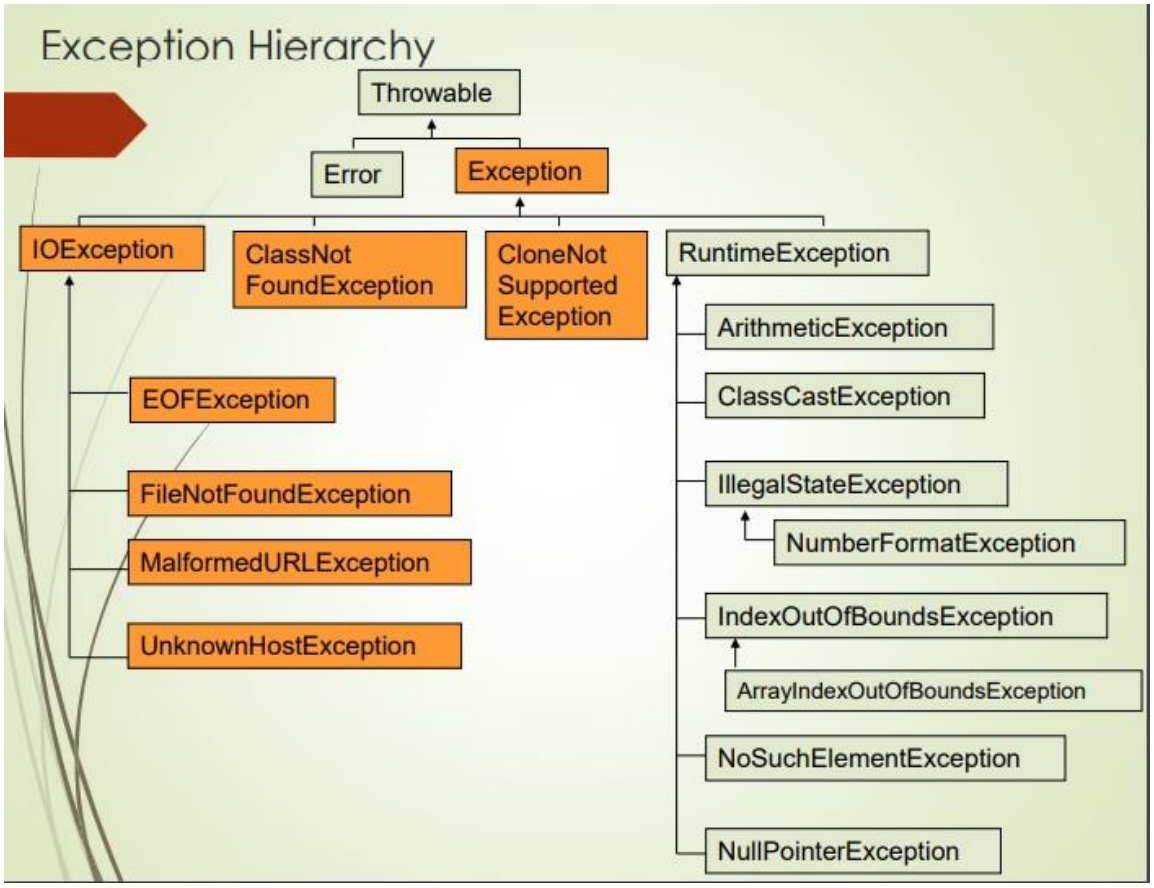
34.C

Exception tablosu düşünülürse error ve exception, Throwablenin altsınıflarıdır.

35.?? Değerler denenecek

36.A

classCastExceptionRuntimeException un alt sınıfı oldupu için önce o yazılmalıdır.



https://m.firat.edu.tr/upload/user_449/903469dda765f135ba3dd165b3baf7a0d8d3c8a2_dosya_449.pdf

37.D

exception kullanmanın ana amacı uygulama sonlanmasının önüne geçmektir.

38.C

```
package body;
publicclassHeartextends Organ {
		publicvoidoperate() throwsException {
			System.out.print("beat");
		}
		//Organ sınıfındanextendedilmişve organ
sınıfırunTimeExceptiondanthrownedilmişken
		//thrownexceptiondenilereküstsınıfıthrownedilmeyecalılşılıyor
		publicstaticvoid main(String... cholesterol) throwsException {
			try {
				newHeart().operate();
			} finally {
			}
		}
}
```

39.B ?

Throw komutu ile ilgili exception fırlatılır.

40.D

```
packageclothing;

publicclassCoat {
	publicLongzipper() throwsException {
		try {
			StringcheckZipper = (String) new Object();
		} catch (Exceptione) {
			throwRuntimeException("Broken!");
			//newkeyword ü eklenemli
		}
		returnnull;
	}

	publicstaticvoid main(String... warmth) {
		try {
			newCoat().zipper();
			System.out.print("Finished!");
		} catch (Throwablet) {
		}
	}
}
```

41. ?? 3 tanesini yazdırıyor test et

42.?

43.D

ErrorThrowable'in alt sınıfıdır, IllegalArgumentException ise Exception'ın alt sınıfıdır.

44.D

```
packagecastles;
classDragonExceptionextendsException {
}

publicclassLair {
	publicvoidopenDrawbridge() throwsException { // r1
		try {
			thrownewException("ThisException");
		} catch (RuntimeExceptione) {
			throwDragonException(); // r2
		} finally{
			thrownewRuntimeException("Ormaybethisone");
		}
	}

	publicstaticvoid main(String[] moat) throwsException {
		newLair().openDrawbridge(); // r3
	}
}
```

45.C

İkisideruntimeException un altsınıfları olduğu için öncelik sıralamaları yoktur.

46.D

```
class Problem implementsRuntimeException  satırında RunTimeExceptionimplements değil extends
edilmeliydi.
classProblemextendsRuntimeException {
```

47.D

```
packagelighting;
publicclassLightBulbimplements Source {
	publicvoidflipSwitch() {
		try {
			throwsnewRuntimeException("Circuit Break!");
			// throwsdeğilthrowkeyword ü kullanılmasıgerekiyor
			} finally {
				System.out.print("Flipped!");
				} }

	publicstaticvoid main(String... electricity) throwsThrowable {
		final Source bulb = newLightBulb();
		bulb.flipSwitch();
	}
}
```

48.B

İnternet bağlantısı bulunmuyorsa error verilebilir.

49.C

“e” ifadesi hem FileNotFoundException için hem de Exception için kullnaılmıiş.

50.B

X1 satırında Exception ifadesi handle edilmeden kullanılmış.