1.C

String sınıfları immutable StringBuffer/StringBuilder ise mutable olduğunu yukarda dedik.(String = Değiştirilemez, StringBuffer||StringBuilder=Değiştirilebilir)

String objenin içine saklanan değer değiştirilemez. Peki nasıl oluyor da ben istediğim zaman string bir değişkeni değiştirebiliyorum diye sorabilirsiniz. Sizin görmüş olduğunuz nesene aynı nesne değildir. String objesi değişmek için arka planda yeni bir String nesnesi oluşturur. Her değişikilte yeni bir String class'ı oluşuyor. Bu da zamanla performansı kötü yönde etkiliyor.

StringBuffer/StringBuilder objelerini kullanırsanız performans açısından daha iyi sonuçlar elde edersiniz. Çünkü Objenin içinde saklanan değer değiştirilebilir bir değerdir.

StringBuffer ile StringBuilder arasında ki tek fark ise "senkronizasyon"dur. StringBuffer "synchronized" iken StringBuilder "synchronized" değildir.

Thread kullancaksanız; StringBuffer, kullanmayacaksanız StringBuilder kullanmanız daha verimli olcaktır.

2.D

Stringler immutabledir yani değiştirilemezler

3.D

4.B

```
publicclass Q4 {
	publicstaticvoid main(String[] args) {
		StringBuilder teams = new StringBuilder("333");
		teams.append(" 806");
		teams.append(" 1601");
		System.out.print(teams);
	}
}
```

5.B

Sadece ArrayList'i kabul ediyor.

6.C

```
import java.util.ArrayList;
import java.util.List;

publicclass Q6 {
	publicstaticvoid main(String[] args) {
		List<String>tools = new ArrayList<>();
		tools.add("hammer");
		tools.add("nail");
		tools.add("hex key");
		System.out.println(tools.get(1));
	}
}
```

7.C

```
publicclass Q7 {
	publicstaticvoid main(String[] args) {
		StringBuilder sb = new StringBuilder("radical").insert(sb.length(), "robots");
		//The local variable sb may not have been initialized
		System.out.println(sb);
```

```
        }
}
```

8.??

Kod derleniyor ama bir çıktı vermiyor.

9.C

```
publicclass Q9 {
        publicstaticvoid main(String[] args) {
                StringBuilder b = new StringBuilder("12");
                b = b.append("3");
                b.reverse();
                System.out.println(b.toString());
        }
}
```

10.D

Programlama dili çerçevesinde Lambda, anonim tekil görevler olarak değerlendirilebilir. Lambda deyimleri (Lambda fonksiyonları da denebilir), referans verilebilir ve tekrar tekrar kullanılabilirdir. Genel kullanım açısından Lambda fonksiyonları, diğer bir fonksiyona argüman olarak iletilebilirdir. Böylece, bir tarafta tanımlanan iş birimi, diğer bir iş biriminde koşturulabilir olmaktadır. Burada dikkat çekilecek unsur, bir iş biriminin diğer bir uygulama birimine referans olarak eriştirilebilirliğidir.

11.D

```
publicclass Q10 {
        publicstaticvoid main(String[] args) {
                StringBuilder line = new StringBuilder("-");
                StringBuilder anotherLine = line.append("-");
                System.out.print(line == anotherLine);
                System.out.print(" ");
                System.out.print(line.length());
        }
}
```

12.A ???

```
import java.util.ArrayList;

publicclass Q12 {
        publicstaticvoid secret(ArrayListmystery) {
                mystery.add("metal");
                 String str = (String) mystery.get(0);
                intnum = ((CharSequence) mystery).length();
                }
}
```

13.?? Hatalı gibi

14.A

```
import java.util.ArrayList;
import java.util.List;

publicclass Q14 {
        publicstaticvoid main(String[] args) {
                List<Character>chars = new ArrayList<>();
                chars.add('a');
                chars.add('b');
```

```
                chars.set(1, 'c');
                chars.remove(0);
                System.out.print(chars.size() + " " + chars.contains('b'));
        }
}
```

15.D

```
publicclass Q15 {
        publicstaticvoid main(String[] args) {
                String b = "12";
                b += "3";
                b.reverse();
                //reverse metotutanımlanmadıgıiçinderlenmez
                System.out.println(b.toString());

        }
}
```

16.C

```
import java.util.function.Predicate;

publicclass Q16 {
publicstaticvoid main(String[] args) {
        //Predicate<String> pred1 = s -> false;
        //Predicate<String> pred2 = (s) -> false;
        Predicate<String> pred3 = String s ->false;
        //Multiple markers at this line
        //Predicate<String> pred4 = (String s) -> false;
}
}
```

17.A

```
publicclass Shoot {
        interface Target {
                boolean needToAim(doubleangle);
        }

        staticvoid prepare(doubleangle, Target t) {
                booleanready = t.needToAim(angle); // k1
                System.out.println(ready);
        }

        publicstaticvoid main(String[] args) {
                prepare(45, d ->d> 5 || d< -5); // k2
        }
}
```

18.A

```
publicclass Q18 {
        publicstaticvoid main(String[] args) {
                String teams = new String("694");
                teams.concat(" 1155");
                teams.concat(" 2265");
                teams.concat(" 2869");
                System.out.println(teams);
        }
}
```

19.A

20.C

Diğer seçenekler “radicalrobots” cıkrısı verirken C seçeneği “radicalrobots” çıktısı vermektedir.

21.A

```
import java.util.Arrays;
import java.util.List;

publicclass Q21 {
	publicstaticvoid main(String[] args) {
		String[] array = { "Natural History", "Science" };
		List<String>museums = Arrays.asList(array);
		museums.set(0, "Art");
		System.out.println(museums.contains("Art"));
	}
}
```

22.??

23.D

Çünkü runtime exception fırlatıyor.

24.B

```
publicclass Q24 {
	publicstaticvoid secret( Stringmystery) {
		mystery = mystery.replace("1", "8");
		mystery.startsWith("paper");
		Strings = mystery.toString();
		}
}
```

NOT: çalıştırıldığında ezception fırlatıyor

25.??

26.D

27.A

```
public class Q27 {
	public static void main(String[] args) {
		String line = new String("-");
		String anotherLine = line.concat("-");
		System.out.print(line == anotherLine);
		System.out.print(" ");
		System.out.print(line.length());
	}
}
```

28.C

```
import java.util.function.Predicate;

public class Q28 {
	public static void main(String[] args) {
		Predicate dash = c -> c.startsWith("-");
		// The method startsWith(String) is undefined for the type Object
		System.out.println(dash.test("-"));
	}
}
```

29.B

LocalDate sadece tarih içermektedir, LocalDateTime ve LocalTime saat,dakika ve sn içermektedir.

30.D

StringIndexOutOfBoundsException alınacaktır.

31.?

32.B

```
import java.time.LocalDate;

public class Q32 {
	public static void main(String[] args) {
		LocalDate xmas = LocalDate.of(2016, 12, 25);
		xmas.plusDays(-1);
		System.out.println(xmas.getDayOfMonth());
	}
}
```

33.A

```
public class Legos {
	public static void main(String[] args) {
		StringBuilder sb = new StringBuilder();
		sb.append("red");
		sb.deleteCharAt(0);
		sb.delete(1, 2);
		System.out.println(sb);
	}
}
```

34.B

```
import java.util.function.Predicate;

public class Q34 {
	public static void main(String[] args) {
		Predicate clear = c -> c.equals("clear");
		System.out.println(clear.test("pink"));
	}
}
```

35.?

36.

37.

38.B

```
import java.time.*;
import java.time.format.*;

public class HowLong {
	public static void main(String[] args) {
		LocalDate newYears = LocalDate.of(2017, 1, 1);
		Period period = Period.ofDays(1);
		DateTimeFormatter format = DateTimeFormatter.ofPattern("MM-dd-yyyy");
		System.out.print(format.format(newYears.minus(period)));
	}
}
```

39.C

```
public class Q39 {
	public static void main(String[] args) {
```

```
                String happy = " :) - (: ";
                String really = happy.trim();
                String question = happy.substring(1, happy.length() - 1);
                System.out.println(really.equals(question));
        }
}
```

40.?

41.D

```
public class Countdown {
        public static void main(String[] args) {
                StringBuilder builder = new StringBuilder("54321");
                builder.substring(2);
                System.out.println(builder.charAt(1));
        }
}
```

42.B

```
import java.util.ArrayList;
import java.util.List;

public class Q42 {
        public static void main(String[] args) {
                List<Integer> pennies = new ArrayList<>();
                pennies.add(3);
                pennies.add(2);
                pennies.add(1);
                pennies.remove(2);
                System.out.println(pennies);
        }
}
```

43.C

```
public class Q43 {
        public static void secret(StringBuilder mystery) {
                char ch = mystery.charAt(3);
                mystery = mystery.insert(1, "more");
                int num = mystery.length();
        }
}
```

44.C

LocalTime en küçük Nanosecond ayıntısındadır Bu yüzden de D(Picosecond) yanlış C dogru oluyor.

45.D

```
import java.time.*;
import java.time.format.*;

public class Q45 {
        public static void main(String[] args) {
                LocalDate newYears = LocalDate.of(2017, 1, 1);
                Period period = Period.ofDays(1);
                DateTimeFormatter format = DateTimeFormatter.ofPattern("mm-dd-yyyy");
                System.out.print(format.format(newYears.minus(period)));
        }
}
// java.time.temporal.UnsupportedTemporalTypeException
```

47.D

```
import java.util.*;
import java.util.function.*;

public class PrintNegative {
	public static void main(String[] args) {
		List<String> list = new ArrayList<>();
		list.add("-5");
		list.add("0");
		list.add("5");
		print(list, e -> e < 0);
	}

	public static void print(List<String> list, Predicate<Integer> p) {
		for (String num : list)
			if (p.test(num))
				// The method test(Integer) in the type Predicate<Integer> is not
applicable for
				// the arguments (String)
				System.out.println(num);
	}
}
```

48.D

java.lang.IndexOutOfBoundsException hatası vermektedir.

49.C

```
public class Costume {
	public static void main(String[] black) {
		String witch = 'b';
		String tail = "lack";
		witch = witch.concat(tail);
		System.out.println(witch);
	}
}
```

50.C

```
import java.time.LocalDate;

public class Q50 {
	public static void main(String[] args) {
		LocalDate xmas = LocalDate.of(2016, 12, 25);
		xmas.setYear(2017);
		//setYear metodu yok, yerine withYear var
		System.out.println(xmas.getYear());
	}
}
```