1.D

2.B

3.A

4.C

5.A

break keywordu ile mevcut dongu sonlanir ve sonraki islemlere devam edilir

6.B

7.B

For dongsunde parantez icinde uc bolum vardir**.**

8.C

9.

10.A

Do-while dongusunde once Boolean control edlir sonra do ici yapilir.

11.B

```
public class Q11 {
        public static void main(String[] args) {
                int singer = 0;
                while (singer)
                        // singer boolean olmadigi icin derlenemeyecek
                        System.out.println(singer++);
        }
}
```

12.B

```
public class Q12 {
        public static void main(String[] args) {
                List<String> drinks = Arrays.asList("can", "cup");
                for (int container = drinks.size() - 1; container >= 0;
                                container-- )
                System.out.print(drinks.get(container) + ",");
        }
}
```

13.A

```
public class Q13 {
        public static void main(String[] args) {
                List<String> bottles = Arrays.asList("glass", "plastic");
                for (int type = 0; type < bottles.size();) {
                        System.out.print(bottles.get(type) + ",");
                        break;
                }
                System.out.print("end");
        }
}
```

14.A

```
public class Q14 {
        public static void main(String[] args) {
                String letters = "";
                while (letters.length() != 2)
                        letters += "a";
                System.out.println(letters);
        }
}
```

15.D

```
public class Q15 {
        public static void main(String[] args) {
                for (int i = args.length; i >= 0; i++)
                        System.out.println("args");
        }
}
```

Sonsuz donguye giecek ve surekli args yazacaktir

16.B

```
public class Q16 {

        private static int count;
        private static String[] stops = new String[] { "Washington", "Monroe", "Jackson", "LaSalle" };

        public static void main(String[] args) {
                while (count < stops.length) {
                        if (stops[count++].length() < 8) {
                                break;
                        }
                }
                System.out.println(count);
        }
}
```

17.C

```
public class Q17 {
        public static void main(String[] args) {
                do {
                        int count = 0;
                        do {
                                count++;
                        } while (count < 2);
                        break;
                } while (true);
                System.out.println(count);
                // count yerel bir degiken oldugu icin burada gorulmeyecek
        }
}
```

18.D

Segmentlerin bos birakilmasi uygulama derlenmesine etki etmemektedir

19.D

20.A

```
public class Q20 {
        public static void main(String[] args) {
                List<String> drinks = Arrays.asList("can", "cup");
                for (int container = 0; container < drinks.size(); container++)
```

```
                        System.out.print(drinks.get(container) + ",");
        }
}
```

21.D

Normal parantez yerine suslu parantez kullanilmalidir.

22.D

Verilen kod parcalarinin hic birisi calismaz.

23.D

Break, break letters, break numbers diyerek donguden cikilabilir.

24.B

Continue letters diyerek diyagramdaki sekil saglanabilir.

25.C

Kod derlenir ama dongunu icine giremedigi icin snuc uretmez

26.C

27.B

```
public class Q27 {
        public static void main(String[] args) {
                boolean balloonInflated = false;
                do {
                        if (!balloonInflated) {
                                balloonInflated = true;
                                System.out.print("inflate-");
                        }
                } while (!balloonInflated);
                System.out.println("done");
        }
}
```

28.C

```
public class Q28 {
        public static void main(String[] args) {
                String letters = "";
                while (letters.length() != 3)
                        //leters uzunlugu 2 4 6 8 seklinde artarak geidecegi icin looptan 
cikamayacaktir.
                        letters += "ab";
                System.out.println(letters);
        }}
```

29.B

For dongusu initialization expression, boolean conditional, update statement seklinde siralanir.

30.B

```
public class Q30 {
        public static void main(String[] args) {
                int count = 10;
                List<Character> chars = new ArrayList<>();
                do {
                        chars.add('a');
                        for (Character x : chars)
                                count -= 1;
                } while (count > 0);
                System.out.println(chars.size());
        }
}
```

31.A

```
public class Q31 {
        public static void main(String[] args) {
                int k = 0;
                for (int i = 10; i > 0; i--) {
                        while (i > 3)
                                i -= 3;
                        k += 1;
                }
                System.out.println(k);
        }
}
```

32.B

for (String f : fun) System.out.println(f);

ifadesiyle f icine fun dizisi degerleri sirasiyla atanacak

33.C

```
public class Q33 {
        public static void main(String[] args) {
                List<String> bottles = Arrays.asList("glass", "plastic");
                for (int type = 0; type < bottles.size();)
                        System.out.print(bottles.get(type) + ",");
                break;
                //break for dongsunun disina konulmus
                System.out.print("end");
        }
}
```

34.C

```
public class Q34 {
public static void main(String[] args) {
        String[] nycTourLoops = new String[] { "Downtown", "Uptown", "Brooklyn" };
        String[] times = new String[] { "Day", "Night" };
        for (int i = 0, j = 0; i < nycTourLoops.length
                        && j < times.length; i++; j++)
                // yukarida i++ dan sona ; yerine , kullanilmaliydi
        {
        System.out.print(nycTourLoops[i] + " " + times[j] + "-");
        }
}
}
```

35.D

ArrayIndexOutOfBoundsException hatası vermektedır.

36.B

```
public class Q36 {
        public static void main(String[] args) {
                String tie = null;
                while (tie == null)
                        tie = "shoelace";
                System.out.print(tie);
        }
}
```

37.C

`break loop;` satirinda loop dongusu ortadan kalktgi icin hata alinir

38.C

```
public class Q38 {
        private static int count;
        private static String[] stops = new String[] { "Washington", "Monroe", "Jackson", "LaSalle" };

        public static void main(String[] args) {
                while (count < stops.length) {
                        if (stops[count++].length() < 8) {
                                continue;
                        }
                }
                System.out.println(count);
        }
}
```

39.C

`while (builder);` satirinda builder boolean turunde olmadigi icin hata alinir

40.A

```
public class Q39 {
        public static void main(String[] args) {
                int count = 0;
                do {
                        do {
                                count++;
                        } while (count < 2);
                        break;
                } while (true);
                System.out.println(count);
        }
}
```

41.C

Break t ile butun loptan cikilir

42.B

```
public class Q42 {
        public static void main(String[] args) {
                String[] nycTourLoops = new String[] { "Downtown", "Uptown", "Brooklyn" };
                String[] times = new String[] { "Day", "Night" };
                for (int i = 0, j = 0; i < nycTourLoops.length && j < times.length; i++, j++) {
                        System.out.print(nycTourLoops[i] + " " + times[j] + "-");
                }
        }
}
```

43.B

```
public class Q43 {
        public static void main(String[] args) {
                List<String> exams = Arrays.asList("OCA", "OCP");
                for (String e1 : exams)
                        for (String e2 : exams)
                                System.out.println(e1 + " " + e2);
        }
}
```

44.B

Once alpha calisacak sonra beta ve beta false oldugu icin tekrar alpa sonra betaya bakilacak

45.B

Once alpha calistirilacak, sonra beta true ise delta yaplacak, sonra gama yapilacak. tekrar betanin durumuna gore donguye girilecek

46.C

C secenegi 6 tekrar yaparken digerleri 5 yapiyor.

47.D

Tie = “shoelace” dongunun icinde olmadigi icin donguden cikilmayacaktir.

48.C

For tanimli bir isim oldugu icin etiket olarak kullanilmaz.

49.D

inflate- yazildiktan sonra donguden cikamayacaktir.

50.B

```
public class Q50 {
        public static void main(String... args) {
                String[] nycTourLoops = new String[] { "Downtown", "Uptown", "Brooklyn" };
                String[] times = new String[] { "Day", "Night" };
                for (int i = 0, j = 0; i < 1; i++, j++)
                        System.out.println(nycTourLoops[i] + " " + times[j]);
        }
}
```