socrative

Answer Key
February 24, 2021

OCA Practice Questions Ch. 5 ('Copy')

1. Which type of loop is best known for its boolean condition that controls entry to the loop?

A do-while loop

B for (traditional)

C for-each

✓ D while

i A while loop has a condition that returns a boolean that controls the loop. It appears at the beginning and is checked before entering the loop. Therefore, Option D is correct. A traditional for loop also has a boolean condition that is checked before entering the loop. However, it is best known for having a counter variable, making Option B incorrect. Option A is incorrect because the boolean condition on a do-while loop is at the end of the loop. Option C is incorrect because there is no condition as part of the loop construct.

2. Which type of loop is best known for using an index or counter?

A do-while loop

✓ B for (traditional)

C for-each

D while

i A traditional for loop is best known for having a loop variable counting up or down as the loop progresses. Therefore, Option B is correct. Options A and D are incorrect because do-while and while loops are known for their boolean conditions. Option C is incorrect because the for-each loop iterates through without an index.

3. Which type of loop is guaranteed to have the body execute at least once?

✓ A do-while loop

B for (traditional)

C for-each

D while

i A do-while loop checks the loop condition after execution of the loop body. This ensures it always executes at least once, and Option A is correct. Option B is incorrect because there are loops you can write that do not ever enter the loop body, such as for (int i=0;i<1;i++). Similarly, Option D is incorrect because a while loop can be written where the initial loop condition is false. Option C is incorrect because a for-each loop does not enter the loop body when iterating over an empty list.

4. Which of the following can loop through an array without referring to the elements by index?

A do-while loop

B for (traditional)

✓ C for-each

D while

i While a traditional for loop often loops through an array, it uses an index to do so, making Option B incorrect. The for-each loop goes through each element, storing it in a variable. Option C is correct.

5. What keyword is used to end the current loop iteration and proceed execution with the next iteration of that loop?

A break

✓ B continue

C end

D skip

i The continue keyword is used to end the loop iteration immediately and resume execution at the next iteration. Therefore, Option B is correct. Option A is incorrect because the break statement causes execution to proceed after the loop body. Options C and D are incorrect because these are not keywords in Java.

6. What keyword is used to proceed with execution immediately after a loop?

✓ A break

B continue

C end

D skip

i The break keyword is used to end the loop iteration immediately, skip any remaining executions of the loop, and resume execution immediately after the loop. Therefore, Option A is correct. Option B is incorrect because execution proceeds at the next execution of the current loop for continue. Options C and D are incorrect because these are not keywords in Java.

7. Which type of loop has three segments within parentheses?

A do-while loop

✓ B for (traditional)

C for-each

D while

i A traditional for loop is best known for having an initialization statement, condition statement, and update statement. Option B is correct.

8. Which of the following statements is/are true?
I. A traditional for loop can iterate through an array starting from index 0.
II. A traditional for loop can iterate through an array starting from the end.

- A Only I
- B Only II
- ✓ C Both statements
- D Neither statement

i With a traditional for loop, you control the order in which indexes are visited in code. This means you can loop through an array in ascending or descending order, and Option C is correct.

9. Which of the following statements is/are true?
I. A for-each loop can iterate through an array starting from index 0.
II. A for-each loop can iterate through an array starting from the end.

- ✓ A Only I
- B Only II
- C Both statements
- D Neither statement

i With a for-each loop, the loop order is determined for you. With an array, this means starting with index 0, and Option A is correct. A traditional for loop allows you to control the order and iterate in either order.

10. Which type of loop has a boolean condition that is first checked after a single iteration through the loop?

- ✓ A do-while loop
- B for (traditional)
- C for-each
- D while

i A do-while loop has a condition that returns a boolean at the end of the loop. Therefore, Option A is correct. Option D is incorrect because a while loop has this condition at the beginning of the loop. A traditional for loop is best known for having a loop variable, making Option B incorrect. Option C is incorrect because there is no condition as part of the loop construct.

11.

```
int singer = 0;
while (singer)
   System.out.println(singer++);
```

What does the following code output?

A 0

✓ B The code does not compile.

C The loops complete with no output.

D This is an infinite loop.

i A while loop requires a boolean condition. While singer is a variable, it is not a boolean. Therefore, the code does not compile, and Option B is correct.

12.

```
List<String> drinks = Arrays.asList("can", "cup");
for (int container = drinks.size() - 1; container >= 0; container--)
  System.out.print(drinks.get(container) + ",");
```

What does the following code output?

A can,cup,

✓ B cup,can,

C The code does not compile.

D None of the above

i This is a correct loop to go through an ArrayList or List starting from the end. It starts with the last index in the list and goes to the first index in the list. Option B is correct.

13.

```
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");
}
```

What does the following code output?

✓ A glass,end

B glass,plastic,end

C The code does not compile.

D None of the above

i The first time through the loop, the index is 0 and glass, is output. The break statement then skips all remaining executions on the loop and the main() method ends. If there was no break keyword, this would be an infinite loop because there's no incrementor.

14.

```
String letters = "";
while (letters.length() != 2)
   letters+="a";
System.out.println(letters);
```

What does the following code output?

✓ **A** aa

B aaa

C The loops complete with no output.

D This is an infinite loop.

i Immediately after letters is initialized, the loop condition is checked. The variable letters is of length 0, which is not equal to 2 so the loop is entered. In the loop body, letters becomes length 1 with contents "a". The loop index is checked again and now 1 is not equal to 2. The loop is entered and letters becomes length 2 and contains "aa". Then the loop index is checked again. Since the length is now 2, the loop is completed and aa is output. Option A is correct.

15.

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

What is the result of the following when run with java peregrine.TimeLoop September 3 1940?

A args

B argsargs

C The code does not compile.

✓ **D** None of the above

i There are three arguments passed to the program. This means that i is 3 on the first iteration of the loop. The program prints args. Then i is incremented to 4. Which is also greater than or equal to 0. Since i never gets smaller, this code produces an infinite loop and the answer is Option D.

16. What is the output of the following code?

A 1

✓ **B** 2

C 3

D The code does not compile.

```
package chicago;
public class Loop {
   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);
   }
}
```

i Since count is a class variable that isn't specifically initialized, it defaults to 0. On the first iteration of the

loop, "Washington", is 11 characters and count is set to 1. The if statement's body is not run. The loop then proceeds to the next iteration. This time, the post-increment operator uses index 1 before setting count to 2. "Monroe" is checked, which is only 6 characters. The break statement sends the execution to after the loop and 2 is output. Option B is correct.

17. What is the result of the following code?

```
do {
   int count = 0;
   do {
      count++;
   } while (count < 2);
      break;
} while (true);
System.out.println(count);
```

A 2

B 3

✓ C The code does not compile.

D This is an infinite loop.

i At first this code appears to be an infinite loop. However, the count variable is declared inside the loop. It is not in scope after the loop where it is referenced by the println(). Therefore, the code does not compile, and Option C is correct.

18.

```
for (segmentA; segmentB; segmentC) {
}
```

Which of the following segments of a for loop can be left blank?

A segmentA

B segmentB

C segmentC

✓ D All of the above

i A for loop is allowed to have all three segments left blank. In fact, for(;;) {} is an infinite loop.

19. How many of the loop types (while, do while, traditional for, and enhanced for) allow you to write code that creates an infinite loop?

A One

B Two

✓ C Three

D Four

i It is not possible to create an infinite loop using a for-each because it simply loops through an array or ArrayList. The other types allow infinite loops, such as, for example, do { } while(true), for(;;) and while(true). Therefore, Option C is correct. And yes, we know it is possible to create an infinite loop with for-each by creating your own custom Iterable. This isn't on the OCA or OCP exam though. If you think the answer is Option D, this is a great reminder of what not to read into on the real exam!

20.

```
List<String> drinks = Arrays.asList("can", "cup");
for (int container = 0; container < drinks.size(); container++)
   System.out.print(drinks.get(container) + ",");
```

What is the output of the following?

✓ **A** can,cup,

B cup,can,

C The code does not compile.

D None of the above

i This is a correct loop to go through an ArrayList or List starting from the beginning. It starts with index 0 and goes to the last index in the list. Option A is correct.

21.

```
do (
   System.out.println("helium");
) while (false);
```

What happens when running the following code?

A It completes successfully without output.

B It outputs helium once.

C It keeps outputting helium.

✓ **D** The code does not compile.

i Braces are optional around loops if there is only one statement. Parentheses are not allowed to surround a loop body though, so the code does not compile, and Option D is correct.

22.

```
for (int i=0; i<fun.length; i++)
   System.out.println(fun[i]);
```

Which of the following is equivalent to this code snippet given an array of String objects?

A for (String f = fun) System.out.println(f);

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

C for (String = fun) System.out.println(it);

D None of the above

i The for-each loop uses a variable and colon as the syntax, making Option B correct.

23.

```
break;
break letters;
break numbers;

letters: for (char ch='a'; ch<='z'; ch++) {
    numbers: for (int n=0; n<=10; n++) {
     System.out.println(ch);

    }
}
```

How many of these statements can be inserted after the println to have the code flow follow the arrow in this diagram?

A None

B One

✓ C Two

D Three

i In this figure, we want to end the inner loop and resume execution at the letters label. This means we only want to break out of the inner loop. A break statement does just that. It ends the current loop and resumes execution immediately after the loop, making break; a correct answer. The break numbers; statement explicitly says which loop to end, which does the same thing, making it correct as well. By contrast, break letters; ends the outer loop, causing the code only to run the println() once. Therefore, two statements correctly match the diagram, and Option C is correct.

24.

```
continue;
continue letters;
continue numbers;
```

Using the diagram in the previous question, how many of these statements can be inserted after the println to have the code flow follow the arrow in the diagram?

A None

✓ B One

C Two

D Three

i In this figure, we want to end the inner loop and resume execution at the letters label. The continue letters; statement does that. The other two statements resume execution at the inner loop. Therefore, only the second statement correctly matches the diagram, and Option B is correct.

25.

```
int singer = 0;
while (singer > 0)
   System.out.println(singer++);
```

What does the following code output?

A 0

B The code does not compile.

✓ C The loops completes with no output.

D This is an infinite loop.

i A while loop checks the boolean condition before entering the loop. In this code, that condition is false, so the loop body is never run. No output is produced, and Option C is correct.

26.

```
for (Object obj : taxis) {
}
```

Which of the following types is taxis not allowed to be in order for this code to compile?

A ArrayList<Integer>

B int[]

✓ C StringBuilder

D All of these are allowed.

i A for-each loop is allowed to be used with arrays and ArrayList objects. StringBuilder is not an allowed type for this loop, so Option C is the answer.

27.

```
boolean balloonInflated = false;
do {
   if (!balloonInflated) {
      balloonInflated = true;
      System.out.print("inflate-");
   }
} while (! balloonInflated);
System.out.println("done");
```

What is the output of the following?

A done

✓ B inflate-done

C The code does not compile.

D This is an infinite loop.

i This is a correct do-while loop. On the first iteration of the loop, the if statement executes and prints inflate-. Then the loop condition is checked. The variable balloonInflated is true, so the loop condition is false and the loop completes.

28.

```
String letters = "";
while (letters.length() != 3)
   letters+="ab";
System.out.println(letters);
```

What does the following code output?

A ab

B abab

C The loop completes with no output.

✓ D This is an infinite loop.

i Immediately after letters is initialized, the loop condition is checked. The variable letters is of length 0, which is not equal to 3, so the loop is entered. In the loop body, letters becomes length 2 and contains "ab". The loop index is checked again and now 2 is not equal to 3. The loop is entered and letters becomes length 4 with contents "abab". Then the loop index is checked again. Since the length 4 is not equal to 3, the loop body is entered again. This repeats for 6, 8, 10, etc. The loop never ends, and Option D is correct.

29. What describes the order in which the three expressions appear in a for loop?

A boolean conditional, initialization expression, update statement

✓ B initialization expression, boolean conditional, update statement

C initialization expression, update statement, boolean conditional

D None of the above

i In a for loop, the segments are an initialization expression, a boolean conditional, and an update statement in that order. Therefore, Option B is correct.

30.

```
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());
```

What is the result of the following?

A 3

✓ B 4

C The code does not compile.

D None of the above

i On the first iteration through the outer loop, chars becomes 1 element. The inner loop is run once and

count becomes 9. On the second iteration through the outer loop, chars becomes 2 elements. The inner loop runs twice so count becomes 7. On the third iteration through the outer loop, chars becomes 3 elements. The inner loop runs three times so count becomes 4. On the fourth iteration through the outer loop, chars becomes 4 elements. The inner loop runs four times so count becomes 0. Then both loops end. Therefore, Option B is correct.

31.

```
int k = 0;
for (int i = 10; i > 0; i--) {
  while (i > 3) i -= 3;
  k += 1;
}
System.out.println(k);
```

What is the result of the following?

✓ **A** 1

B 2

C 3

D 4

i On the first iteration of the outer loop, i starts out at 10. The inner loop sees that 10 > 3 and subtracts 3, making the 7 the new value of i. Since 7 > 3, we subtract 3 again, making i set to 4. Yet again 4 > 3, so i becomes 1. Then k is finally incremented to 1. The outer loop decrements i i, making it 0. The boolean condition sees that 0 is not greater than 0. The outer loop ends and 1 is printed out. Therefore, Option A is correct.

32.

```
for (int i=fun.length-1; i>=0; i--)
  System.out.println(fun[i]);
```

Which of the following is equivalent to this code snippet given an array of String objects?

A for (String f = fun) System.out.println(f);

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

C for (String f fun) System.out.println(it);

✓ **D** None of the above

i Options A and C do not compile as they do not use the correct syntax for a for-each loop. The for-each loop is only able to go through an array in ascending order. It is not able to control the order, making Option C incorrect. Therefore, Option D is the answer.

33.

```
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");
}
```

What does the following code output?

A glass,end

B glass,plastic,end

✓ C The code does not compile.

D None of the above

i Since there are no brackets around the for statement, the loop body is only one line. The break statement is not in the loop. Since break cannot be used at the top level of a method, the code does not compile, and Option C is correct.

34.

```
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] + "-");
}
```

What is the result of the following?

A Downtown Day-

B Downtown Day-Uptown Night-

✓ C The code does not compile.

D The code compiles but throws an exception at runtime.

i Multiple update expressions are separated with a comma rather than a semicolon. Tricky, we know. But it is an important distinction. This makes Option C correct.

35.

```
package peregrine;
public class TimeLoop {
  public static void main(String[] args) {
    for (int i = args.length; i>=0; i--)
       System.out.println(args[i]);
  }
}
```

What is the result of the following when run with java peregrine.TimeLoop September 3 1940?

A September

B 1940

C The code does not compile.

✓ D None of the above

i There are three arguments passed to the program. This means that i is 3 on the first iteration of the loop. The program attempts to print args[3]. Since indexes are zero based in Java, it throws an

ArrayIndexOutOfBoundsException.

36.

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

What is the output of the following?

A null

✓ B shoelace

C shoelaceshoelace

D None of the above

i The first time the loop condition is checked, the variable tie is null. The loop body executes, setting tie. Despite the indention, there are no brackets surrounding the loop body so the print does not run yet. Then the loop condition is checked and tie is not null. The print runs after the loop, printing out shoelace once, making Option B correct.

37.

```
23:  String race = "";
24:  loop:
25:  do {
26:     race += "x";
27:     break loop;
28:  } while (true);
29:  System.out.println(race);
```

The following code outputs a single letter x. What happens if you remove lines 25 and 28?

A It prints an empty string.

B It still outputs a single letter x.

✓ C It no longer compiles.

D It becomes an infinite loop.

i The code compiles as is. However, we aren't asked about whether the code compiles as is. Line 27 refers to a loop label. While the label is still present, it no longer points to a loop. This causes the code to not compile, and Option C is correct.

38.

```
package chicago;
public class Loop {
   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);
   }
}
```

What is the output of the following code?

A 1

B 2

✓ C 4

D The code does not compile.

i The continue statement is useless here since there is no code later in the loop to skip. The continue statement merely resumes execution at the next iteration of the loop, which is what would happen if the if-then statement was empty. Therefore, count increments for each element of the array. The code outputs 4, and Option C is correct.

39.

```
StringBuilder builder = new StringBuilder();
String str = new String("Leaves growing");
do {
   System.out.println(str);
} while (builder);
System.out.println(builder);
```

What is the output of the following?

A Leaves growing

B This is an infinite loop.

✓ C The code does not compile.

D The code compiles but throws an exception at runtime.

i A do-while loop requires a boolean condition. The builder variable is a StringBuilder and not a boolean. The code does not compile, and Option C is correct.

40.

```
6:  int count = 0;
7:  do {
8:     do {
9:        count++;
10:    } while (count < 2);
11:    break;
12: } while (true);
13: System.out.println(count);
```

What is the result of the following code?

✓ **A** 2

B 3

C The code does not compile.

D This is an infinite loop.

i At first this code appears to be an infinite loop. However, there is a break statement. On line 6, count is set to 0. On line 9, it is changed to 1. Then the condition on line 10 runs. count is less than 2 so the inner loop continues. Then count is set to 2 on the next iteration of the inner loop. The loop condition on line 10 runs again and this time is false. The inner loop is completed. Then line 11 of the outer loop runs and sends execution to after the loop on line 13. At this point count is still 2, so Option A is correct.

41.

```
t: while (true) {
    f: while(true) {
        ___________
    }
}
```

Fill in the blank so this code compiles and does not cause an infinite loop.

A break;

B break f;

✓ **C** break t;

D None of the above

i Option A breaks out of the inner loop, but the outer loop is still infinite. Option B has the same problem. Option C is correct because it breaks out of both loops.

42.

```
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] + "-");
}
```

What is the result of the following?

A Downtown Day-

✓ B Downtown Day-Uptown Night-

C The code does not compile.

D The code compiles but throws an exception at runtime.

i This code is correct. It initializes two variables and uses both variables in the condition check and the update statements. Since it checks the size of both arrays correctly, it prints the first two sets of elements, and Option B is correct.

43.

```
import java.util.*;
public class Exams {
   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);
   }
}
```

How many lines does the following code output?

A One

✓ B Four

C The code does not compile.

D The code compiles but throws an exception at runtime.

i Looping through the same list multiple times is allowed. The outer loop executes twice. The inner loop executes twice for each of those iterations of the outer loop. Therefore, the inner loop executes four times, and Option B is correct.

44.

```
for (alpha; beta; gamma) {
   delta;
}
```

Which of the following best describes the flow of execution in this for loop if beta always returns false?

A alpha

✓ B alpha, beta

C alpha, beta, gamma

D None of the above

The initializer, which is alpha, runs first. Then Java checks the condition, which is beta, to see if loop execution should start. Since beta returns false, the loop is never entered, and Option B is correct.

45.

```
for (alpha; beta; gamma) {
  delta;
}
```

Which of the following best describes the flow of execution in this for loop if the loop body is run exactly once?

- A alpha, delta, gamma, beta
- ✓ B alpha, beta, delta, gamma, beta
- C alpha, delta, gamma, alpha, beta
- D alpha, beta, delta, gamma, alpha, beta

The initializer, which is alpha, runs first. Then Java checks the condition, which is beta, to see if loop execution should start. Then the loop body, which is delta, runs. After the loop execution, the updater, which is gamma, runs. Then the loop condition, which is beta, is checked again. Therefore, Option B is correct.

46. Which of the following iterates a different number of times than the others?

- A for (int k=0; k < 5; k++) {}
- B for (int k=1; k <= 5; k++) {}
- ✓ C int k=0; do { } while(k++ < 5)
- D int k=0; while (k++ < 5) {}

Option A goes through five indexes on the iterations: 0, 1, 2, 3 and 4. Option B also goes through five indexes: 1, 2, 3, 4 and 5. Option D goes through five iterations as well, from 0 to 4. However, Option C goes through six iterations since the loop condition is at the end of the loop. Therefore it is not like the others, and Option C is the answer.

47.

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

What is the output of the following?

- A null
- B shoelace
- C shoelaceshoelace
- ✓ D None of the above

i The first time the loop condition is checked, the variable tie is null. However, the loop body is empty due to the semicolon right after the condition. This means the loop condition keeps running with no opportunity for tie to be set. Therefore, this is an infinite loop, and Option D is correct.

48.

```
12:  int result = 8;
13:  for: while (result > 7) {
14:     result++;
15:     do {
16:        result--;
17:     } while (result > 5);
18:     break for;
19:  }
20:  System.out.println(result);
```

What is the output of the following?

A 5

B 8

✓ C The code does not compile.

D The code compiles but throws an exception at runtime.

i Remember to look for basic errors before wasting time tracking the flow. In this case, the label of the loop is trying to use the keyword for. This is not allowed, so the code does not compile. If the label was valid, Option A would be correct.

49.

```
boolean baloonInflated = false;
do {
   if (!baloonInflated) {
      baloonInflated = true;
      System.out.print("inflate-");
   }
} while (baloonInflated);
System.out.println("done");
```

What is the output of the following?

A done

B inflate-done

C The code does not compile.

✓ D This is an infinite loop.

i On the first iteration of the loop, the if statement executes printing inflate-. Then the loop condition is checked. The variable baloonInflated is true, so the loop condition is true and the loop continues. The if statement no longer runs, but the variable never changes state again, so the loop doesn't end.

50.

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

Which of the following can fill in the blank to have the code compile successfully?

A int i=0; j=0;

✓ B int i=0, j=0;

C int i=0; int j=0;

D int i=0, int j=0;

i In a for loop, the type is only allowed to be specified once. A comma separates multiple variables since they are part of the same statement. Therefore, Option B is correct.