Microsoft

98-388

Introduction to Programming Using Java

QUESTION & ANSWERS

Question: 1

HOTSPOT

You are writing a Java method named safeRoot. The method must meet the following requirements:

- Accept two `double` parameters `radicand` and `index`
- If `radicand` is negative and `index` is even, return `null`
- If `radicand` is negative and `index` is odd, return `-Math.pow(-radicand, 1 / index)`
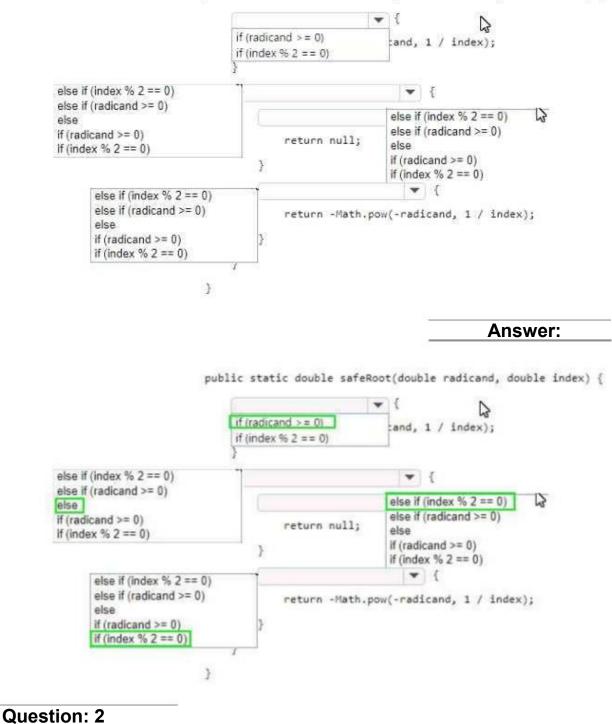- Otherwise, return `Math.pow(radicand, 1 / index)`

How should you complete the code? To answer, select the appropriate code segments in the answer area. NOTE: Each correct selection is worth one point.

Answer Area

```
public static double safeRoot(double radicand, double index) {
                         ▼  {
    if (radicand >= 0)       and, 1 / index);
    if (index % 2 == 0)
    }
                         ▼  {
                         ▼  {
        return null;
    }
                         ▼  {
        return -Math.pow(-radicand, 1 / index);
    }
    }
}
```

```
public static double safeRoot(double radicand, double index) {
```

```
                                    ▼ {                    ⊿
    if (radicand >= 0)
    if (index % 2 == 0)                     and, 1 / index);
    }
```

```
else if (index % 2 == 0)                            ▼ {
else if (radicand >= 0)
else                                     else if (index % 2 == 0)    ⊿
if (radicand >= 0)          return null;  else if (radicand >= 0)
if (index % 2 == 0)                      else
                                         if (radicand >= 0)
                   }                     if (index % 2 == 0)
```

```
    else if (index % 2 == 0)                            ▼ {
    else if (radicand >= 0)
    else                           return -Math.pow(-radicand, 1 / index);
    if (radicand >= 0)
    if (index % 2 == 0)                }
                   }
```

```
}
```

Answer:

```
public static double safeRoot(double radicand, double index) {
```

```
                                    ▼ {                    ⊿
    if (radicand >= 0)
    if (index % 2 == 0)                     and, 1 / index);
    }
```

```
else if (index % 2 == 0)                            ▼ {
else if (radicand >= 0)
else                                     else if (index % 2 == 0)    ⊿
if (radicand >= 0)          return null;  else if (radicand >= 0)
if (index % 2 == 0)                      else
                                         if (radicand >= 0)
                   }                     if (index % 2 == 0)
```

```
    else if (index % 2 == 0)                            ▼ {
    else if (radicand >= 0)         return -Math.pow(-radicand, 1 / index);
    else
    if (radicand >= 0)                 }
    if (index % 2 == 0)
                   }
```

```
}
```

Question: 2

HOTSPOT
You work as an intern Java programmer at Adventure Works. Your team lead asks you to create a method. The method must meet the following requirements:

- Accept an int array
- Check for duplicate values in the array
- Stop the outer loop as soon as a duplicate value has been detected and return true
- Return false if all values in the array are unique

How should you complete the code? To answer, select the appropriate code segments in the answer are

a. NOTE: Each correct selection is worth one point.

```
public static boolean duplicate(int[] array) {

    boolean isDuplicate = false;

    for (                [▼]              [▼]   x++) {

        for (int y = x + 1; y < array.length;        [▼] )

            if (array[x] == array[y])

                isDuplicate = true;

        if (isDuplicate)

            [▼]

    }

    return isDuplicate;

}
```

```
public static boolean duplicate(int[] array) {

    boolean isDuplicate = false;

    for (            ▼                    )
                                              x < array.length - 2;
                                              x < array.length - 1;
                                              x <= array.length;
                                              x <= array.length - 1;

        x = 0;
    for x = 1;         1; y < array.length;          ▼  )
        int x = 1;
        int x = 0;        == array[y])        x = x + 1
                                              y++
        isDuplicate = true;                   y = y - 1
                                              x--

    if (isDuplicate)

                          ▼
        break;
        switch;
    }   finally;
        continue;
    return isDuplicate;

}
```

Answer:

```
public static boolean duplicate(int[] array) {

    boolean isDuplicate = false;

    for (                     ▼                              ▼   x++) {
            x = 0;                  x < array.length - 2;
            x = 1;                  x < array.length - 1;
            int x = 1;              x <= array.length;
            int x = 0;              x <= array.length - 1;

        for (int y = x + 1; y < array.length;                ▼  )
                                                x = x + 1
            if (array[x] == array[y])           y++
                                                y = y - 1
                isDuplicate = true;             x--

        if (isDuplicate)

                                  ▼
                break;
                switch;
                finally;
                continue;

    }

    return isDuplicate;

}
```

Question: 3

HOTSPOT
You are interviewing for a job as a Java developer. You need to demonstrate your understanding of switch statements.

For each of the following code segments, select Yes if the code segment can be changed to a switch statement with up to three case statements. Otherwise, select No. NOTE: Each correct selection is worth one point.

	Yes	No

```
if (age >= 25) {
    discount = 0.50;
} else if (age >= 21) {
    discount = 0.25;
} else {
    discount = 0.0;
}
```
○ Yes   ○ No

```
if (grade == "A") {
    message = "Exceeds Standards";
} else if (grade == "B") {
    message = "Meets Standards";
} else {
    message = "Needs Improvement";
}
```
○ Yes   ○ No

```
if (gpa == 4.0) {
    priority = 1;
} else if (gpa >= 3.0) {
    priority = 2;
} else if (gpa >= 2.5) {
    priority = 3;
}
```
○ Yes   ○ No

Answer:

	Yes	No

```
if (age >= 25) {
    discount = 0.50;
} else if (age >= 21) {
    discount = 0.25;
} else {
    discount = 0.0;
}
```
Yes ◉   No ○

```
if (grade == "A") {
    message = "Exceeds Standards";
} else if (grade == "B") {
    message = "Meets Standards";
} else {
    message = "Needs Improvement";
}
```
Yes ◉   No ○

```
if (gpa == 4.0) {
    priority = 1;
} else if (gpa >= 3.0) {
    priority = 2;
} else if (gpa >= 2.5) {
    priority = 3;
}
```
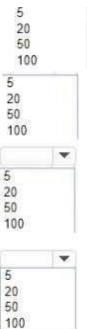Yes ◉   No ○

Question: 4

HOTSPOT
You need to evaluate the following code. Line numbers are included for reference only.

```
01 public static int fee(char model) {
02   int price = 0;
03   switch (model) {
04     case 'A':
05       price = 50;
06       break;
07     case 'T':
08       price = 20;
09     case 'C':
10       price = 5;
11       break;
12     default:
13       price = 100;
14     break;
15   }
16   return price;
17 }
```

Use the drop-down menus to select the answer choice that answers each question based on the information presented in the code.

What is the return value when model has a value of 'A'?

5
20
50
100

What is the return value when model has a value of 'T'?

5
20
50
100

What is the return value when model has a value of 'C'?

5
20
50
100

What is the return value when model has any other value?

5
20
50
100

Answer:

What is the return value when model has a value of 'A'?

5
20
50
100

What is the return value when model has a value of 'T'?

5
20
50
100

What is the return value when model has a value of 'C'?

5
20
50
100

What is the return value when model has any other value?

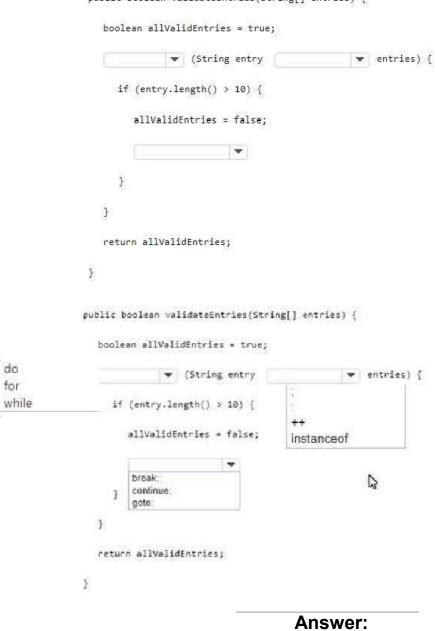| 5 |
|---|
| 20 |
| 50 |
| 100 |

Question: 5

HOTSPOT
You are writing a Java method.
The method must meet the following requirements:

- Accept a string array named entries
- Iterate through entries
- Stop the iteration and return false if any element has more than 10 characters
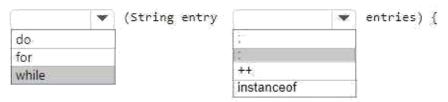- Otherwise, return true

```
public boolean validateEntries(String[] entries) {

    boolean allValidEntries = true;

    [_____ ▼] (String entry [_____ ▼] entries) {

        if (entry.length() > 10) {

            allValidEntries = false;

            [_____ ▼]

        }

    }

    return allValidEntries;

}
```

```
public boolean validateEntries(String[] entries) {

    boolean allValidEntries = true;

    [_____ ▼] (String entry [_____ ▼] entries) {

        if (entry.length() > 10) {

            allValidEntries = false;

            [_____ ▼]

        }

    }

    return allValidEntries;

}
```

do
for
while

;
;
++
instanceof

break;
continue;
goto;

Answer:

```
public boolean validateEntries(String[] entries) {

    boolean allValidEntries = true;

    [ _____ ▼ ] (String entry [ _____ ▼ ] entries) {
     | do    |                         | :         |
     | for   |                         | :         |
     | while |                         | ++        |
                                       | instanceof|

      if (entry.length() > 10) {

          allValidEntries = false;

          [ _____ ▼ ]
           | break;    |
           | continue; |
           | goto;     |

      }

    }

    return allValidEntries;

}
```