CDE Custom Assessment - Java Practice Test > daniel.velasqueziturrate@cognizant.com

You can view this report online at : https://www.hackerrank.com/x/tests/1049024/candidates/35421398/report

Full Name:	Daniel Velasquez	
Email:	daniel.velasqueziturrate@cognizant.com	
Test Name:	**CDE Custom Assessment - Java Practice Test**	
Taken On:	25 Jan 2022 14:30:24 IST	
Time Taken:	118 min 50 sec/ 120 min	
Work Experience:	1 years	
Invited by:	Seshadri	
Invited on:	20 Jan 2022 16:58:18 IST	
Skills Score:	Java (Basic)	0/75
	Problem Solving (Intermediate)	55/75
Tags Score:	Algorithms	55/75
	Dynamic Programming	55/75
	Interfaces	0/75
	Interviewer Guidelines	55/75
	Java	0/75
	Language Proficiency	0/75
	Medium	55/150
	Overloading	0/75
	Problem Solving	55/150
	Theme: E-commerce	55/75

44.7%

85/190

scored in **CDE Custom Assessment - Java Practice Test** in 118 min 50 sec on 25 Jan 2022 14:30:24 IST

Recruiter/Team Comments:

No Comments.

Plagiarism flagged

We have marked questions with suspected plagiarism below. Please review.

	Question Description	Time Taken	Score	Status
Q1	Shape Inheritance > **Coding**	46 min 40 sec	0/ 75	✗
Q2	Method Reference > **Multiple Choice**	4 min 54 sec	5/ 5	✓
Q3	Static Import > **Multiple Choice**	5 min 52 sec	0/ 5	✗
Q4	Lambda Expression - Easy > **Multiple Choice**	6 min 3 sec	5/ 5	✓

Q5	Lambda Expression - Easy › Multiple Choice	4 min 18 sec	5/ 5	✓
Q6	MSA - Storage › Multiple Choice	3 min 30 sec	0/ 5	☒
Q7	Monolith to MSA › Multiple Choice	6 min 38 sec	5/ 5	✓
Q8	Spring Boot Starter › Multiple Choice	2 min 3 sec	5/ 5	✓
Q9	Auto Config › Multiple Choice	12 min 38 sec	5/ 5	✓
Q10	Ways to Sum › Coding	25 min 38 sec	55/ 75	!

QUESTION 1

☒

Wrong Answer

Score 0

Shape Inheritance › Coding

Java | Interfaces | Overloading | Language Proficiency | Medium | Problem Solving

QUESTION DESCRIPTION

The area and perimeter of different geometric shapes like rectangle, circle, or square are calculated using different mathematical formulae.

Create the following three classes:

1. The *Rectangle* class should implement the *Shape* interface. It should have two class variables of float type, length and, width. Also, it should implement the following methods:

- *Rectangle(float new_length, float new_width)*: sets the class variables, *length = new_length* and *width = new_width* respectively.
- *float getArea():* returns the result of *length × width*, the area of the rectangle. Also prints "*Finding area of rectangle with length = {length} and width = {width}*", where *{length}* and *{width}* are the respective values of class variables *length* and *width*.
- *float getPerimeter():* returns the result of *2 × (length + width)*, the perimeter of the rectangle. Also prints *"Finding perimeter of rectangle with length = {length} and width = {width}"*.
- *String toString():* returns the string *"Rectangle = [length: {length}, width: {width}, area: {area}, perimeter: {perimeter}]"*.

Note: *{length}* and *{width}* respectively represent the class variables *length* and *width*. *{area}* and *{perimeter} respectively represent the area and perimeter of the rectangle.* For example, given that *length = 2* and *width = 3*, calling the method *toString()*will return "*Rectangle = [length: 2.0, width: 3.0, area: 6.0, perimeter: 10.0]*".

2. The *Square* class that should inherit the *Rectangle* class and should implement the following methods:

- *Square(float side):* sets the variables of Rectangle class, *length = side* and *width = side* respectively.
- *float getArea():* returns the result of *length × width* that denotes the area of the square. Also prints *"Finding area of square with side = {length}"*.
- *float getPerimeter():* returns the result of *4 × length* that denotes the perimeter of the square. Also prints *"Finding perimeter of square with side = {length}"*.
- *String toString():* returns the string *"Square = [side: {length}, area: {area}, perimeter: {perimeter}]"*.

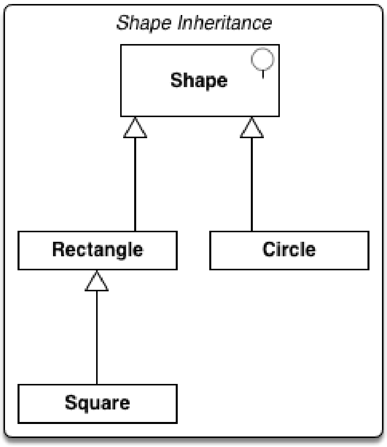
Note: *{length}* represents the variable *length* of Rectangle class. *{area}* and *{perimeter} respectively represent the area and perimeter of the square.* For example, given that *side = 2,* calling the method *toString()*will return "*Square = [side: 2.0, area: 4.0, perimeter: 8.0]*".

3. The *Circle* class that implements the *Shape* interface. It should have a class variable, radius. Also, it should implement the following methods:

- *Circle(float new_radius):* sets the float type variable of Circle class, *radius* = new_radius.
- *float getArea():* returns the result of *3.14 × radius × radius* that denotes the area of the circle. Also prints *"Finding area of circle with radius = {radius}"*.
- *float getPerimeter():* returns the result of *6.28 × radius* that denotes the perimeter of the circle. Also prints *"Finding perimeter of circle with radius = {radius}"*

- Also prints "Finding perimeter of circle with radius = {radius}".
- *String toString():* returns the string *"Circle = [radius: {radius}, area: {area}, perimeter: {perimeter}]"*.

Note: *{radius}* represents the variable *radius* of Circle class. *{area}* and *{perimeter} respectively represent the area and perimeter of the circle.* For example, given that *radius = 1,* calling the method *toString()*will return "*Circle = [radius: 1.0, area: 3.14, perimeter: 6.28]*".



The locked stub code in the editor provides the definition of the *Shape* interface. It also validates the implementation of *Rectangle*, *Square*, and *Circle* classes.

Constraints

- *1 < length, width, side, radius < 10^2*

▼ Input Format For Custom Testing

The first line contains two space-separated floats that denote the *length* and *width* of the rectangle.

The next line contains a float that denotes the *side* of the square.

The next line contains a float that denotes the *radius* of the circle.

▼ Sample Case 0

Sample Input For Custom Testing

```
2 3
2
2
```

Sample Output For Custom Testing

```
====================================
Finding area and perimeter of shapes
====================================
Finding area of rectangle with length = 2.0 and width = 3.0
Finding perimeter of rectangle with length = 2.0 and width = 3.0
Area = 6.0 and Perimeter = 10.0

Finding area of square with side = 2.0
Finding area of rectangle with length = 2.0 and width = 2.0
Finding perimeter of square with side = 2.0
Finding perimeter of rectangle with length = 2.0 and width = 2.0
Area = 4.0 and Perimeter = 8.0

Finding area of circle with radius = 2.0
Finding perimeter of circle with radius = 2.0
Area = 12.56 and Perimeter = 12.56

=========================
Printing shapes as string
=========================
Rectangle = [length: 2.0, width: 3.0, area: 6.0, perimeter: 10.0]
Square = [side: 2.0, area: 4.0, perimeter: 8.0]
```

```
Circle = [radius: 2.0, area: 12.56, perimeter: 12.56]
```

Explanation

For the rectangle, the length and width are *2.0* and *3.0*. So, the area is the result of *2.0 × 3.0 = 6.0* and the perimeter is the result of *2 × (2.0 +3.0) = 10.0* (precision to *1* place after the decimal).

For the square, the side length is *2.0*. So, the area is the result of *2.0 × 2.0 = 4.0* and the perimeter is the result of *4 × 2.0 = 8.0* (precision to *1* place).

For the circle, the radius is *2.0*. So, the area is the result of *3.14 × 2.0 × 2.0 = 12.56* and the perimeter is the result of *6.28 × 2.0 = 12.56* (precision to *2* places).

CANDIDATE ANSWER

The candidate did not manually submit any code. The last compiled version has been auto-submitted and the score you see below is for the auto-submitted version.

Language used: **Java 7**

```
public interface Shape{

}

public class Rectangle implements Shape{
    public float lenght;
    public float width;

    public Rectangle(float new_lenght, float new_width){

    }
}

public class Square extends Rectangle{

}

public class Circle implements Shape{
    private double ratio;
}




```

Result: Compilation Failed

Compile Message

```
                Solution.java:9: error: class Shape is public, should be
declared in a file named Shape.java
public interface Shape{
       ^
Solution.java:13: error: class Rectangle is public, should be declared in
a file named Rectangle.java
public class Rectangle implements Shape{
       ^
Solution.java:22: error: class Square is public, should be declared in a
file named Square.java
public class Square extends Rectangle{
       ^
Solution.java:26: error: class Circle is public, should be declared in a
file named Circle.java
public class Circle implements Shape{
       ^
```

```
Solution.java:44: error: constructor Square in class Square cannot be
applied to given types;
        Rectangle square = new Square(side);
                           ^
  required: no arguments
  found: float
  reason: actual and formal argument lists differ in length
Solution.java:45: error: constructor Circle in class Circle cannot be
applied to given types;
        Shape circle = new Circle(radius);
                       ^
  required: no arguments
  found: float
  reason: actual and formal argument lists differ in length
Solution.java:52: error: cannot find symbol
        System.out.println("Area = " + square.getArea() + " and Perimeter
= " + square.getPerimeter() + "\n");
                                             ^
  symbol:   method getArea()
  location: variable square of type Rectangle
Solution.java:52: error: cannot find symbol
        System.out.println("Area = " + square.getArea() + " and Perimeter
= " + square.getPerimeter() + "\n");

^
  symbol:   method getPerimeter()
  location: variable square of type Rectangle
8 errors
```

No Comments

QUESTION 2

✓ Correct Answer

Score 5

Method Reference › Multiple Choice

QUESTION DESCRIPTION

Which one below is the example of Method reference?

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

○ list.replaceAll(String::toUpperCase())

✓ ● list.replaceAll(String::toUpperCase)

○ list.replaceAll(s -> s.toUpperCase())

○ None of the listed options.

No Comments

QUESTION 3

⊗

Wrong Answer

Score 0

Static Import › Multiple Choice

QUESTION DESCRIPTION

What is the static import equivalent notation of:
list.sort(Comparator.comparing(p -> p.getName()));

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

- ○ list.sort(comparing(Person::getName()));
- ● list.sort(comparing(Person p -> p.getName()));
- ✓ ○ list.sort(comparing(Person::getName));
- ○ list.sort(comparator(Person::getName));

No Comments

QUESTION 4

✓

Correct Answer

Score 5

Lambda Expression - Easy › Multiple Choice

QUESTION DESCRIPTION

Which of the following is an example of the internal iteration?

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

- ○ for(Person p: list) { System.out.println(p); }
- ○ for(int i=0; i<list.size();i++) { Person p = list[i]; System.out.println(p); }
- ✓ ● list.forEach(s-> System.out.println(s));
- ○ System.out.println(list);

No Comments

QUESTION 5

✓ Correct Answer

Score 5

Lambda Expression - Easy › Multiple Choice

QUESTION DESCRIPTION

What is TRUE about Lambda?

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

✓ ● Lambda expression enable functions to be passed as argument.

○ It is neither a function nor a interface.

○ Lambda is denoted with => sign.

○ None of the given options.

No Comments

QUESTION 6

⊗ Wrong Answer

Score 0

MSA - Storage › Multiple Choice

QUESTION DESCRIPTION

How does Data Storage differ between a monolithic application and a microservices based application?

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

○ Monolithic applications share data between applications with independent data storage

● Microservices based applications uses small database tables with single responsibility

○ In Microservices Architecture, there is likely a shared central database

✓ ○ Microservices Architecture utilizes independent Data Storage

No Comments

QUESTION 7

✓ Correct Answer

Score 5

Monolith to MSA › Multiple Choice

QUESTION DESCRIPTION

You are breaking a HealthCare based Billing Application from Monolithic to Microservices based application. The Monolithic App has 3 layers.

Layer 1: User Interface
Layer 2: Business Logic
Layer 3: Database

The first capability you choose to decouple from the monolithic is the company's Billing Algorithm which is confined to server side and does not deal with User Interface or Database elements. What is wrong in this approach?

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

○ You've selected a capability that doesn't include the UI to start the conversation with.

○ You've selected a capability that doesn't include the database to start the conversation with.

○ You've selected a single capability to start the conversation with.

✓ ● You've selected a high-risk business critical capability to start the conversation with.

No Comments

QUESTION 8

✓ Correct Answer

Score 5

Spring Boot Starter › Multiple Choice

QUESTION DESCRIPTION

You are developing a Spring Boot Application and need to add support for JDBC. Which Spring Boot starter would you use?

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

○ springboot-starter-jdbc

○ spring-boot-jdbc-starter

✓ ● spring-boot-starter-jdbc

○ jdbc-spring-boot-starter

No Comments

QUESTION 9

✓

Correct Answer

Score 5

Auto Config › Multiple Choice

QUESTION DESCRIPTION

Which of the following is/are TRUE?

A. If you added @SpringBootApplication annotation to the class, you do not need to add the @EnableAutoConfiguration, @ComponentScan and @SpringBootConfiguration annotation.
B. If you added @SpringBootApplication annotation to the class, you do not need to add the @SpringBootConfiguration annotation.
C. @SpringBootApplication annotated class should have the main method to run the Spring Boot application
D. Either @EnableAutoConfiguration annotation or @SpringBootApplication annotation on your main class will ensure that the Spring Boot Application is automatically configured

CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)

○ A and D

✓ ● A, C and D

○ A only

○ B and D

No Comments

QUESTION 10

!

Needs Review

Score 55

Ways to Sum › Coding

Medium | Dynamic Programming | Problem Solving | Algorithms | Theme: E-commerce | Interviewer Guidelines

QUESTION DESCRIPTION

An automated packaging system is responsible for packing boxes. A box is certified to hold a certain weight. Given an integer *total*, calculate the number of possible ways to achieve *total* as a sum of the weights of items weighing integer weights from *1* to *k,* inclusive.

Example

total = 8

k = *2*

To reach a weight of 8, there are 5 different ways that items with weights between 1 and 2 can be combined:

- [1, 1, 1, 1, 1, 1, 1, 1]
- [1, 1, 1, 1, 1, 1, 2]
- [1, 1, 1, 1, 2, 2]
- [1, 1, 2, 2, 2]
- [2, 2, 2, 2]

Function Description

Complete the function *ways* in the editor below.

ways has the following parameter(s):

int total: the value to sum to

int k: the maximum of the range of integers to consider when summing to *total*

Returns

int: the number of ways to sum to the *total*; the number might be very large, so return the integer modulo *1000000007 (10^9+7)*

Constraints

- 1 ≤ *total* ≤ 1000
- 1 ≤ *k* ≤ 100

▼ Input Format For Custom Testing

The first line contains an integer, *total*, that denotes the target sum.

The second line contains an integer, *k*, that denotes the maximum value in the range of integers to be considered, i.e, from *1* to *k*.

▼ Sample Case 0

Sample Input For Custom Testing

```
STDIN     Function
-----     --------
5       →  total = 5
3       →  k = 3
```

Sample Output

```
5
```

Explanation

The sum required is *5*. *k = 3* so the integers that can be considered to reach the sum are *[1, 2, 3]*.

The *5* ways to reach the target sum are:

1. *1 + 1 + 1 + 1 + 1 = 5*
2. *1 + 1 + 1 + 2 = 5*
3. *1 + 2 + 2 = 5*
4. *1 + 1 + 3 = 5*
5. *2 + 3 = 5*

5 modulo 1000000007 = 5

▼ Sample Case 1

Sample Input For Custom Testing

```
STDIN    Function
-----    --------
4      →  total = 4
2      →  k = 2
```

Sample Output

```
3
```

Explanation

The sum required is *4,* and the range of integers is *[1, 2]*

There are *3* ways to reach the target sum:

1. *1 + 1 + 1 + 1 = 4*
2. *1 + 1 + 2 = 4*
3. *2 + 2 = 4*

3 modulo 1000000007 = 3.

INTERVIEWER GUIDELINES

INTERVIEWER GUIDELINES

▼ Hint 1

We can approach the problem by first thinking of a recursive solution and then convert it into dynamic programming. We can break this into 2 parts for a particular weight:

Use this weight

Do not use this weight

Let the following be the recursive function which evaluates the number of different ways:

```
int recur(w, sum, total, k)
```

where, w = current weight to be considered
sum = weight sum of all the weights that has been taken
total = total weight sum that is required
k = maximum weight allowed

Hence, for both the cases we have the following recurrence:

```
Case 1: Use this weight → recur(w, sum + w, total, k)
Case 2: Do not use this weight → recur(w + 1, sum, total, k)
```

▼ Solution

Concepts covered: Dynamic Programming

Optimal Solution:

We can solve the problem using dynamic programming. Let's suppose we have a sub-problem dp[j] which denotes the number of ways to make the sum of weights equal to j. Also, suppose we already know dp[j]. We need to calculate the transitions possible using this state. It can be seen that we can go from j to i + j for any i ∈ [1, k]. Using this our recurrence is as follows:

```
dp[i + j] += dp[j] for all j ∈ [1, k]
```

But, to ensure that permutations of the same weights do not count as different ways, you need to calculate dp[i] for all i ∈ [1, n] with respect to every weight one by one. For more details see the code below.

```
M = 10**9 + 7
def ways(total, k):
    dp = [0] * (total + 1)
    dp[0] = 1
    for i in range(1, k + 1):
        for j in range(total + 1):
            if j + i <= total:
                dp[j + i] = (dp[j + i] + dp[j]) % M
    return dp[total]
```

Brute Force Approach: Passes 6 of 13 test cases

```
ans = 0
def recur(i, sum, total, k):
    global ans
    if i == k + 1:
        if sum == total:
            ans += 1
        return
    recur(i + 1, sum, total, k)
    if sum + i <= total:
        recur(i, sum + i, total, k)
def ways(total, k):
    recur(1, 0, total, k)
    return ans
```

Error Handling: Candidates may interchange the order of the first and second loops. This leads to counting ways that have the same values of weights but in different positions which will fail most test cases.

▼ Complexity Analysis

Time Complexity - O(total * k).

There are two loops in the given solution:the first one takes *total* passes and the second one takes *k* passes. The overall time complexity is O(total * k)

Space Complexity - O(total).

Since an additional array of length *total* is required for the answers of sub-problems, the space complexity is O(total).

CANDIDATE ANSWER

Language used: **Java 7**

```
class Result {

    /*
     * Complete the 'ways' function below.
     *
     * The function is expected to return an INTEGER.
     * The function accepts following parameters:
     *  1. INTEGER total
     *  2. INTEGER k
     */

    public static int ways(int total, int k) {
        int[] l = new int[total + 1];

        l[0] = 1;

        for(int i = 1; i < k + 1; i++){
            for(int j = 1; j < total + 1; j++){
                if(j >= i){
                    l[j] = l[j] + l[j - i];
                }
            }
        }

        return(l[total]);

    }



}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample case	✓ Success	1	0.0664 sec	22.1 KB
TestCase 1	Easy	Sample case	✓ Success	1	0.0769 sec	22.1 KB

TestCase 2	Easy	Sample case	✓ Success	8	0.0902 sec	22.1 KB
TestCase 3	Easy	Hidden case	✓ Success	8	0.1195 sec	21.9 KB
TestCase 4	Easy	Hidden case	✓ Success	8	0.0832 sec	22.1 KB
TestCase 5	Medium	Sample case	✓ Success	7	0.1242 sec	22.1 KB
TestCase 6	Medium	Hidden case	✓ Success	7	0.0956 sec	22.1 KB
TestCase 7	Medium	Hidden case	✓ Success	7	0.1119 sec	21.9 KB
TestCase 8	Medium	Hidden case	✓ Success	8	0.0771 sec	21.9 KB
TestCase 9	Hard	Sample case	⊗ Wrong Answer	0	0.1252 sec	22.8 KB
TestCase 10	Hard	Hidden case	⊗ Wrong Answer	0	0.1159 sec	22.8 KB
TestCase 11	Hard	Hidden case	⊗ Wrong Answer	0	0.1598 sec	22 KB
TestCase 12	Hard	Hidden case	⊗ Wrong Answer	0	0.0858 sec	22.8 KB

No Comments

PDF generated at: 25 Jan 2022 11:01:21 UTC