



You can view this report online at : <https://www.hackerrank.com/x/tests/1049024/candidates/35421465/report>

Full Name:	Konstantina Karav.
Email:	konstantina.karavoulia@cognizant.com
Test Name:	CDE Custom Assessment - Java Practice Test
Taken On:	25 Jan 2022 14:32:19 IST
Time Taken:	119 min 15 sec/ 120 min
Work Experience:	1 years
Invited by:	Seshadri
Invited on:	20 Jan 2022 16:58:18 IST
Skills Score:	<div>Java (Basic) 65/75</div> <div>Problem Solving (Advanced) 9/75</div>
Tags Score:	<div>Algorithms 9/75</div> <div>Dynamic Programming 9/75</div> <div>Interfaces 65/75</div> <div>Java 65/75</div> <div>Medium 74/150</div> <div>OOPS 65/75</div> <div>Problem Solving 9/75</div>

52.1%

99/190

scored in **CDE Custom Assessment - Java Practice Test** in 119 min 15 sec on 25 Jan 2022 14:32:19 IST

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Employee Implementation > Coding	58 min 53 sec	65/ 75	✓
Q2	Method Reference > Multiple Choice	1 min 52 sec	5/ 5	✓
Q3	Higher Order Function > Multiple Choice	3 min 35 sec	0/ 5	✗
Q4	Lambda Expression - Easy > Multiple Choice	2 min 28 sec	5/ 5	✓
Q5	Lambda Expression - Easy > Multiple Choice	2 min 47 sec	0/ 5	✗
Q6	Domain Model > Multiple Choice	2 min 52 sec	5/ 5	✓
Q7	Service Layer - MSA > Multiple Choice	1 min 46 sec	5/ 5	✓
Q8	Spring Boot Starter > Multiple Choice	4 min 59 sec	5/ 5	✓
Q9	Auto Config > Multiple Choice	3 min 6 sec	0/ 5	✗



QUESTION 1



Correct Answer

Score 65

Employee Implementation > Coding

Java

Interfaces

OOPS

Medium

QUESTION DESCRIPTION

An interface named *Company* has the following methods:

- `void assignSalaries(int[] salaries);`
- `void averageSalary();`
- `void maxSalary();`
- `void minSalary();`

Create 2 classes, *EngineerFirm* and *AccountantFirm*, that implement the *Company* interface. The details of these classes follow.

1. Class *EngineerFirm* should have a variable of type `int[] income`. It should implement the following methods:

- *EngineerFirm(int n):*
 - Initializes the empty array *income* of length *n* where *n* is the number of engineers.
 - Assigns 0 income to all the engineers.
- *void assignSalaries(int[] salaries):* Assigns the salaries in array *salaries* to array *income*.
 - If the *salaries* and *income* arrays differ in length, assigns as many values as possible and then stops.
 - Prints "Incomes of engineers credited".
- *void averageSalary():* Prints the average salary in the format "Average salary of engineers is {averageSalary}".
 - Zero values in *income*, if any, should be included in the average calculation.
- *void maxSalary():* Prints the maximum salary in the format "Maximum salary amongst engineers is {maximumSalary}".
- *void minSalary():* Prints the minimum salary in the format "Minimum salary amongst engineers is {minimumSalary}".

2. Class *AccountantFirm* should have a variable of type `int[] income`. It should implement the following methods:

- *AccountantFirm(int n):*
 - Initializes the empty array *income* of length *n* where *n* is the number of accountants.
 - Assigns 0 income to each accountant.
- *void assignSalaries(int[] salaries):* Assigns the salaries in array *salaries* to array *income*.
 - If the *salaries* and *income* arrays differ in length, assigns as many values as possible and then stops.
 - Prints "Incomes of accountants credited".
- *void averageSalary():* Prints the average salary in the format "Average salary of accountants is {averageSalary}".
 - Zero values in *income*, if any, should be included in the average calculation.
- *void maxSalary():* Prints the maximum salary in the format "Maximum salary amongst accountants is {maximumSalary}".
- *void minSalary():* Prints the minimum salary in the format "Minimum salary amongst accountants is {minimumSalary}".

Note: Please use inheritance and encapsulation to minimize code repetition. The locked code stub provides the interface *Company* and also validates the implementation of the *EngineerFirm* and *AccountantFirm* classes.

Input Format For Custom Testing

The first line contains two space-separated integers, n and m , the number of engineers and accountants respectively.

The second line contains space-separated integers that represent the incomes of engineers.

The third line contains space-separated integers that represent the incomes of accountants.

▼ Sample Case 0

Sample Input For Custom Testing

```
5 5
6848 9329 9984 5543 7986
9317 7796 3352 7068 9500
```

Sample Output

```
Incomes of engineers credited
Incomes of accountants credited
Average salary of engineers is 7938.00
Maximum salary amongst engineers is 9984
Minimum salary amongst engineers is 5543
Average salary of accountants is 7406.60
Maximum salary amongst accountants is 9500
Minimum salary amongst accountants is 3352
```

CANDIDATE ANSWER

Language used: **Java 7**

```
1  class EngineerFirm implements Company{
2      private int[] income;
3      void setIncome (int[] income){
4          this.income = income;
5      }
6
7      int[] getIncome (){
8          return this.income;
9      }
10
11     public EngineerFirm(int n){
12         this.income = new int[n];
13     }
14
15     public void assignSalaries(int[] salaries){
16
17         int i=0;
18         for(int salary: salaries){
19             this.getIncome()[i] = salary;
20             i++;
21
22             if(i == this.getIncome().length)
23                 break;
24         }
25
26         System.out.println("Incomes of engineers credited");
27     }
28
29     public void averageSalary(){
30         double average = 0;
31         int i = 0;
32         int sum = 0;
33         for(int inc: this.getIncome()){
```

```

34         sum += inc;
35         i++;
36     }
37     average = (double)sum / i;
38     System.out.print("Average salary of engineers is ");
39     System.out.printf("%.2f", average);
40     System.out.println("");
41 }
42
43 public void maxSalary(){
44     int max = 0;
45     for(int inc: this.getIncome()){
46         if(inc > max)
47             max = inc;
48     }
49     System.out.print("Maximum salary amongst engineers is ");
50     System.out.println(max);
51 }
52
53 public void minSalary(){
54     int min = 999999999;
55     for(int inc: this.getIncome()){
56         if(inc < min)
57             min = inc;
58     }
59     System.out.print("Minimum salary amongst engineers is ");
60     System.out.println(min);
61 }
62
63 class AccountantFirm implements Company{
64     private int[] income;
65     void setIncome (int[] income){
66         this.income = income;
67     }
68
69     int[] getIncome (){
70         return this.income;
71     }
72
73     public AccountantFirm(int n){
74         this.income = new int[n];
75     }
76
77     public void assignSalaries(int[] salaries){
78
79         int i=0;
80         for(int salary: salaries){
81             this.getIncome()[i] = salary;
82             i++;
83             if(i == this.getIncome().length)
84                 break;
85         }
86
87         System.out.println("Incomes of accountants credited");
88     }
89
90     public void averageSalary(){
91         int i = 0;
92         int sum = 0;
93         double average = 0;
94         for(int inc: this.getIncome()){
95             sum += inc;
96             i++;
97         }

```

```


98         average = (double)sum / i;
99         System.out.print("Average salary of accountants is ");
10         System.out.printf("%.2f", average);
10         System.out.println("");
10     }
10
10     public void maxSalary(){
10         int max = 0;
10         for(int inc: this.getIncome()){
10             if(inc > max)
10                 max = inc;
10         }
10         System.out.print("Maximum salary amongst accountants is ");
10         System.out.println(max);
10     }
10
10     public void minSalary(){
10         int min = 999999999;
10         for(int inc: this.getIncome()){
10             if(inc < min)
10                 min = inc;
10         }
10         System.out.print("Minimum salary amongst accountants is ");
10         System.out.print(min);
10     }
10 }
10
10
10 /* model output for cut and paste
10 Incomes of ____ credited
10 Average salary of ____ is ____
10 Maximum salary amongst ____ is ____
10 Minimum salary amongst ____ is ____
10 */
10
10
10

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	✔ Success	1	0.1529 sec	24.2 KB
Testcase 1	Easy	Sample case	✔ Success	1	0.1042 sec	24.3 KB
Testcase 2	Medium	Hidden case	✔ Success	10	0.1932 sec	24.2 KB
Testcase 3	Medium	Hidden case	✔ Success	10	0.1428 sec	24.3 KB
Testcase 4	Hard	Hidden case	✔ Success	15	0.1272 sec	24.3 KB
Testcase 5	Hard	Hidden case	✘ Wrong Answer	0	0.107 sec	24.2 KB
Testcase 6	Hard	Hidden case	✔ Success	14	0.1053 sec	24.4 KB
Testcase 7	Hard	Hidden case	✔ Success	14	0.1389 sec	24.3 KB

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

Higher Order Function > Multiple Choice

QUESTION DESCRIPTION

Which one of these is an example of Higher Order Function?


CANDIDATE ANSWER

Options: (Expected answer indicated with a tick)



☐ list.sort() -> p1.getName().compareTo(p2.getName());



☐ list.sort((p1, p2) -> p1.getName().compareTo(p2.getName()));



☒ list.sort((Person p1, Person p2) -> p1.getName().compareTo(p2.getName()));



 ☐ list.sort(Comparator.comparing(p -> p.getName()));

No Comments


QUESTION 4  Correct Answer	Lambda Expression - Easy > Multiple Choice
Score 5	QUESTION DESCRIPTION Which of the following is an example of the internal iteration?
	CANDIDATE ANSWER Options: (Expected answer indicated with a tick) <div><input type="radio"/> for(Person p: list) { System.out.println(p); } <input type="radio"/> for(int i=0; i<list.size();i++) { Person p = list[i]; System.out.println(p); }  <input checked="" type="radio"/> list.forEach(s-> System.out.println(s)); <input type="radio"/> System.out.println(list);</div>
	No Comments

QUESTION 5  Wrong Answer	Lambda Expression - Easy > Multiple Choice
Score 0	QUESTION DESCRIPTION What is TRUE about Lambda?
	CANDIDATE ANSWER Options: (Expected answer indicated with a tick) <div> <input type="radio"/> Lambda expression enable functions to be passed as argument. <input type="radio"/> It is neither a function nor a interface. <input type="radio"/> Lambda is denoted with => sign. <input checked="" type="radio"/> None of the given options.</div>
	No Comments

QUESTION 6  Correct Answer	Domain Model > Multiple Choice
Score 5	QUESTION DESCRIPTION What does the Domain Model in Microservices Application incorporate?
	CANDIDATE ANSWER Options: (Expected answer indicated with a tick) <div><input type="radio"/> Transactions</div> <div> <input checked="" type="radio"/> Data and Behavior</div> <div><input type="radio"/> Procedures</div> <div><input type="radio"/> Data Models</div> No Comments

QUESTION 7  Correct Answer	Service Layer - MSA > Multiple Choice
Score 5	QUESTION DESCRIPTION What role does Service Layer play in the Domain Facade Implementation approach?
	CANDIDATE ANSWER Options: (Expected answer indicated with a tick) <div><input type="radio"/> It uses multiple resource and provides coordination for several responses to an action.</div> <div> <input checked="" type="radio"/> It establishes a boundary and set of operations through which client layers interact with the application.</div> <div><input type="radio"/> It is used to programatically control the transactions</div> <div><input type="radio"/> It factors responsibility to avoid duplication.</div> 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



Wrong Answer

Score 0

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



Correct Answer

Score 9

Grouping Options > Coding Dynamic Programming Problem Solving Medium Algorithms

QUESTION DESCRIPTION

Given a number of people n and a number of groups k , find the distinct options to form k contiguous groups out of the n people while respecting the following conditions:

- In each option, the total of group sizes is equal to the number of people.

- In each option, each group's size should be greater than or equal to the group to its left.
- The groups formed in each option are distinct, meaning that they differ in at least one group. For example, $[1, 1, 1, 3]$ is distinct from $[1, 1, 1, 2]$ but not from $[1, 3, 1, 1]$.

Example

$people = 8$

$groups = 4$

- The 5 distinct options to form 4 groups with 8 people under the rules are:
- The options are $[1, 1, 1, 5]$, $[1, 1, 2, 4]$, $[1, 1, 3, 3]$, $[1, 2, 2, 3]$ and $[2, 2, 2, 2]$.
- In each option, the groups are distinct and each group's size is greater than or equal to the group to its left.

Function Description

Complete the function `countOptions` in the editor below.

`countOptions` has the following parameters:

int people: an integer that denotes the number of people in the row

int groups: an integer that denotes the number of groups to form

Returns:

int: a long integer that denotes the number of ways that n participants can be divided into k groups satisfying the conditions mentioned above.

Constraints

- $1 \leq people, groups \leq 200$

▼ Input Format For Custom Testing

The first line contains an integer, *people*, the number of people in the row.

The next line contains an integer, *groups*, the number of groups to form.

▼ Sample Case 0

Sample Input For Custom Testing

STDIN	Function
-----	-----
7 →	people = 7
3 →	groups = 3

Sample Output

4

Explanation

- The distinct options to form 3 groups with 7 people under the rules, are exactly 4:
- The options are $[1, 1, 5]$, $[1, 2, 4]$, $[1, 3, 3]$ and $[2, 2, 3]$.
- In each option, the groups are distinct and each group is greater than or equal to a group to its left

▼ Sample Case 1

Sample Input For Custom Testing

STDIN	Function
-----	-----
4 →	people = 4
2 →	groups = 2

Sample Output

2

Explanation


- The distinct options to form 2 groups with 4 people under the rules, are exactly 2:
- The options are [1,3] and [2,2].
- In each option, the groups are distinct and each group is greater than or equal to a group to its left

CANDIDATE ANSWER

Language used: **Java 7**

```
1  class Result {
2
3      /*
4       * Complete the 'countOptions' function below.
5       *
6       * The function is expected to return a LONG_INTEGER.
7       * The function accepts following parameters:
8       * 1. INTEGER people
9       * 2. INTEGER groups
10      */
11
12     public static long countOptions(int people, int groups) {
13         long number = 0L;
14
15         if(groups == 0){
16             return 1L;
17         }
18
19         int maxFirst = people/groups;
20
21         for(int i=1; i<=maxFirst; i++){
22             if(groups < 2 && people < 10){
23                 number++;
24                 break;
25             }
26             if(groups >=2)
27                 countOptions(people -1 , groups - i);
28         }
29         return number;
30
31     }
32 }
33
34 }
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample case	⊗ Wrong Answer	0	0.1549 sec	22 KB
TestCase 1	Easy	Sample case	⊗ Wrong Answer	0	0.0629 sec	21.9 KB
TestCase 2	Easy	Hidden case	⊗ Wrong Answer	0	0.117 sec	22.5 KB
TestCase 5	Easy	Sample case	⊗ Terminated due to timeout	0	4.0101 sec	23.2 KB
TestCase 6	Medium	Hidden case	✓ Success	9	0.1135 sec	22.1 KB
TestCase 7	Medium	Hidden case	⊗ Wrong Answer	0	0.0681 sec	22.1 KB
TestCase	Hard	Sample	⊗ Terminated due to	0	4.013 sec	23.2 KB

10		case	timeout				
TestCase	Hard	Hidden case	 Terminated due to	0	4.029 sec	23.3 KB	
11			timeout				

No Comments

PDF generated at: 25 Jan 2022 11:03:16 UTC