Question - 1

SCORE: 2 points

Java Core

medium

Which of the following is the root class (apex) of the exception heirarchy in Java?

- (●) Throwable
- ( ) Exception
- ( ) Error
- ( ) RuntimeException

Question - 2

SCORE: 1 points

Core Java

Easy

Java Virtual Machine is an attempt to the goal that was “write once; run anywhere, any time, forever.”

- (●) TRUE
- ( ) FALSE

Question - 3

SCORE: 2 points

Java Core

easy

What is the output of the following code snippet?

```
int x = 10;
float y = 3.5f;
double z = x + y;
System.out.println(z);
```

- ( ) 13.5
- (●) 14
- ( ) 13
- ( ) 10

Question - 4

SCORE: **2 points**

Java Core

easy

What is the output of the following program?

```
 @FunctionalInterface
public interface MyInterface {
        void foo();
}

public class MyClass implements MyInterface {
    public void foo() {
        System.out.println("Hello");
    }
}

public class MyClass2 extends MyClass implements MyInterface{
    public void foo() {
        System.out.println("Hello 2");
    }
}

public class Main extends MyClass2{
    public static void main(String[] args) {
        MyClass c = new MyClass2();
        c.foo();
    }
}
```

- (●) Hello 2
- ( ) Hello
- ( ) MyClass2 cannot extend a class and an interface at the same time.
- ( ) Runtime error

Question - 5

SCORE: **5 points**

Java PriorityQueue

Collections | Queues | Easy

What is the output of the following code?

```
import java.util.*;
public class Main{
    public static void main(String args[]){
        Queue<String> pq = new PriorityQueue<>();
        pq.add("Hello");
        pq.add("Hackerrank");
        String s1=pq.poll();
        pq.remove();
        System.out.println(s1);
        String s2=pq.remove();
        System.out.println(s2);
```

```
    }
}
```

- Hackerrank
  NoSuchElementException
- Hackerrank
  Hello
- Hackerrank
  null
- Hello
  Hackerrank

Question - 6

SCORE: **2 points**

Java Core

Medium

What is true about protected constructor?

- Protected constructor can be called directly
- Protected constructor can only be called using super()
- Protected constructor can be used outside package
- protected constructor can be instantiated even if child is in a different package

Question - 7

SCORE: **2 points**

Java Core

Medium

What would be behaviour if the constructor has a return type?

- Compilation error
- Runtime error
- Compilation and runs successfully
- Only String return type is allowed

Question - 8

SCORE: **1 points**

Core Java

Medium

Which of the following is a valid syntax to synchronize the HashMap?

- Map m = hashMap.synchronizeMap();
- HashMap map =hashMap.synchronizeMap();
- Map m1 = Collections.synchronizedMap(hashMap);
- Map m2 = Collection.synchronizeMap(hashMap);

Question - 9

SCORE: **1 points**

Core Java

Easy

How many objects will be created in the following?
String a = new String("Interviewbit");
String b = new String("Interviewbit");
Strinc c = "Interviewbit";
String d = "Interviewbit";

- 2
- 3
- 4
- None

Question - 10

SCORE: **1 points**

Core Java

Easy

If we try to add duplicate key to the HashMap, What will happen ?

- It will throw an exception.
- It won't add the new Element without any exception.
- The new element will replace the existing element.
- Compiler will identify the problem and will throw an error.

Question - 11

SCORE: **1 points**

Core Java

Medium

Which are the compatible Data Types for Type Promotion or Type Casting?

- byte, char, short
- char, int, float
- float, long, double
- All the above

Question - 12
SCORE: 1 points

Core Java

Easy

Which of the collections allows null as the key ?

- HashTable
- HashMap
- TreeMap
- LinkedHashMap

Question - 13
SCORE: 1 points

Core Java

Easy

What is the advantage of using lambda expressions over anonymous inner classes?

- Lambda expressions can capture the enclosing class's variables.
- Lambda expressions have better performance.
- Lambda expressions are more readable and concise.
- Lambda expressions can implement multiple methods.

Question - 14
SCORE: 1 points

Core Java

Medium

The concept of ___is often expressed by the phrase “one interface, multiple methods.” This means that it is possible to design a generic interface to a group of related activities. This helps reduce ___ by allowing the same interface to be used to specify a general class of action

- polymorphism, simplicity
- Inheritance, complexity
- polymorphism, complexity

- Multithreading, complexity

Question - 15
SCORE: 1 points

UI with Angular

easy

How can you trigger a Bootstrap modal using JavaScript/jQuery?

- [x] modal.show()
- [ ] modal.open()
- [ ] modal.display()
- [ ] modal.toggle()

Question - 16
SCORE: 1 points

Design patterns

Easy

In how many steps Singleton class in java created?

- [ ] 1
- [x] 2
- [ ] 3
- [ ] 4

Question - 17
SCORE: 5 points

Communication Methods

Distributed Systems | HTTP | JSON | Easy

Which of these is a method of communication between microservices in a larger application?

- [ ] HTTP Protocol
- [ ] Event-driven or Message-driven
- [ ] SOAP
- [x] All of these

Question - 18
SCORE: 1 points

Authguard

Easy

What is the purpose of the "canLoad" property in the route configuration?

- [x] To prevent unauthorized access to a route module.
- [ ] To guard child routes from unauthorized access.
- [ ] To handle navigation events for lazy-loaded modules.
- [ ] To resolve data before activating a route module.

Question - 19

SCORE: **1 points**

Authguard

Easy

How do you handle asynchronous authentication checks in an AuthGuard?

- [ ] By using the "resolve" property in the route configuration.
- [ ] By using the "canActivateAsync" property in the AuthGuard class.
- [x] By using the "async" keyword in front of the "canActivate" method.
- [ ] By using the "CanActivateChild" interface.

Question - 20

SCORE: **1 points**

Authguard

Easy

What is the purpose of the "canActivateChild" property in the route configuration?

- [x] To guard child routes from unauthorized access.
- [ ] To handle navigation events for child routes.
- [ ] To resolve data before activating a child route.
- [ ] To prevent unauthorized access to the parent route.

Question - 21

SCORE: **50 points**

JSON Diff Tool

Strings Easy Real-World

Implement a simple prototype service to find the difference between two JSON (JavaScript Object Notation) objects.

To keep the prototype simple, a JSON will contain only key-value pairs and no nested objects or arrays in it. Given two JSON strings, *json1* and *json2,* find the list of keys for which the values are different. If a key is present in only one of the JSONs, it should not be considered in the result. The list of keys should be sorted in lexicographically ascending order.

Example:

Suppose *json1* = "{"hello":"world","hi":"hello","you":"me"}" and *json2* = "{"hello":"world","hi":"helloo","you":"me"}"

The only common key where the values differ is "hi". Hence the answer is ["hi"].

Function Description

Complete the function *getJSONDiff* in the editor below.

getJSONDiff has the following parameter(s):

json1: the first JSON string

json2: the second JSON string

Returns

string[]: a sorted list of keys that have different associated values in the two JSONs

Constraints

- $1 \leq |json1|, |json2| \leq 10^5$
- There are no white spaces in the string.

▼ Input Format For Custom Testing

The first line contains a string, *json1.*

The next line contains a string, *json2.*

▼ Sample Case 0

Sample Input For Custom Testing

```
STDIN                                   FUNCTION
-----                                   --------
{"hacker":"rank","input":"output"}      →   json1 = {"hacker":"rank","input":"output"}
{"hacker":"ranked","input":"wrong"}     →   json2 = {"hacker":"ranked","input":"wrong"}
```

Sample Output

```
hacker
input
```

Explanation

Neither key's values are the same in both strings.

▼ Sample Case 1

Sample Input For Custom Testing

```
STDIN                                   FUNCTION
-----                                   --------
{"hacker":"rank","input":"output"}      →   json1 = {"hacker":"rank","input":"output"}
{"hacker":"rank","input":"output"}      →   json2 = {"hacker":"rank","input":"output"}
```

Sample Output

```

```

Explanation

Both values match in both strings, so return an empty list.

Question - 22

SCORE: 75 points

Menu Recommendation

OOPS | Medium | Interfaces

Design a menu recommendation system for a restaurant. It suggests an item that one might want to try from a restaurant.

The recommendation is made using the following logic:

- If the head chef decides to offer a dish as "the deal of the day", it is recommended.
- If there is no "deal of the day" item, the item with the highest average rating is recommended.
- If there is a "deal of the day" item, but it is out of stock, the in-stock item with the highest average rating is recommended.

The average rating of an item is calculated as: *(sum of ratings for an item) / (total number of people who have rated this item).*

Complete the class *MenuRecommendation* that implements the interface *IMenuRecommendation*:

- *void addItem(int itemId, String displayName)*: Create and store a *MenuItem* object from the given information. The definition of class *MenuItem* is given in the code stub.
- *MenuItem getRecommendedItem()*: Return the recommended *MenuItem*. If there is no such *MenuItem*, return *null*.
- *void outOfStockItem(int itemId)*: Mark *itemId* as out of stock.
- *void restockItem(int itemId):* Mark *itemId* as back in stock.
- *void makeDealOfTheDayItem(int itemId):* Mark *itemId* as the deal of the day and the recommended item.
- *void rateItem(int itemId, int rating):* A user rated the item with *itemId* as *rating* number of points.

The driver code takes care of input and calls the relevant functions. There are *totalNumberOfRequests,* and each of the next lines is a request that is one of 6 types of function call.

Constraints

- $1 \le totalNumberOfRequests \le 10^5$
- $1 \le itemId \le 10^5$
- $1 \le rating \le 5$
- $1 \le |displayName| \le 10$

▼ Input Format For Custom Testing

The first line contains an integer, *totalNumberOfRequests,* the number of requests.
Each *i* of the next *totalNumberOfRequests* contains a request described above.

▼ Sample Case 0

Sample Input For Custom Testing

```
8
getRecommendedItem
addItem 1 Item1
rateItem 1 5
getRecommendedItem
outOfStockItem 1
rateItem 1 4
rateItem 1 4
getRecommendedItem
```

Sample Output

```
N/A
1 Item1 Rating: 5.0
N/A
```

Explanation

- *getRecommendedItem* - there are no item entries so this outputs 'N/A'
- *addItem 1 Item1* - Adds *Item1* with *itemId* 1.
- rateItem 1 5 - Adds a rating of 5 to *Item1.*
- *getRecommendedItem* - there is only 1 item added yet with 1 rating of 5.
- *outOfStockItem 1* - marks *Item1* as out of stock
- *rateItem 1 4* - Adds a rating of 4 to *Item1.*
- *rateItem 1 4* - Adds a rating of 4 to *Item1.*
- getRecommendedItem - There are no items in stock to recommend.