Question - 1
Java: Classes

Which of the following are true for Java Classes?

○ The Void class extends the Class class.

○ The Float class extends the Double class.

○ The System class extends the Runtime class.

○ The Integer class extends the Number class.

Question - 2
Java: Find the Output

Consider the following Java code snippet:

```
public int divide(int a, int b) {
    int c = -1;

    try {
        c = a / b;
    }
    catch (Exception e) {
        System.err.print("Exception ");
    }
    finally {
        System.err.println("Finally ");
    }

    return c;
}
```

What will our code *print* when we call *divide(4, 0)*?

○ Exception Finally

○ Finally Exception

○ Exception

○ Finally

○ No output

○ -1

Question - 3
Java String Concatenation

What is the output of the following program?

```
public class Question {
        public static void main(String args[]) {
                String s1 = "abc";
                String s2 = "def";
                String s3 = s1.concat(s2.toUpperCase( ) );
                System.out.println(s1+s2+s3);
        }
}
```

○ abcdefabcdef

○ abcabcDEFDEF

○ abcdefabcDEF

○ None of the above

Question - 4
Java: String Array

Which of the following Java declaration of the String array is correct?

○ String temp [] = new String {"j" "a" "z"};

○ String temp [] = { "j " " b" "c"};

○ String temp = {"a", "b", "c"};

○ String temp [] = {"a", "b", "c"};

Question - 5
Java Operators

What is the value of the variable *z* in the statement `int x = 5; int y = x++; int z = ++y;` ?

○ 5

○ 6

○ 7

○ 8

Question - 6

What will be the result of compiling and running the following Java program?

```
What will be the result of compiling and running the following Java program?

public class TipTop
{
  static final Integer i1 = 1;
  final Integer i2 = 2;
  Integer i3 = 3;

  public static void main(String[] args)
  {
    final Integer i4 = 4;
    Integer i5 = 5;

    class Inner
    {
       final Integer i6 = 6;
       Integer i7 = 7;

       Inner()
       {
          System.out.print(i6 + i7);
       }
    }
  }
}
```

○ The program will fail to compile

○ The program will compile, but throw an exception when run

○ The program will compile and print 67, when run

○ The program will compile but will not print anything, when run

Question - 7

Java Interface

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 - 8
Java parseInt

What is the output of the following code?

```
public class Main{
    public static void main(String[] args) {
        try
        {
            int[] a = new int[5];
            String s="1.0";
            try
            {
                System.out.println(Integer.parseInt(s+a[4]));
            }
            catch (Exception e)
            {
                System.out.println("Exception 1 occurs. "+ e.getClass());
            }
            throw new Exception();
        }
         catch (Exception e) {
            System.out.println("Exception 2 occurs");
        }
    }
}
```

○ 1
Exception 2 occurs

○ Exception 1 occurs. class java.lang.NumberFormatException
Exception 2 occurs

○ Exception 1 occurs. class ArrayIndexOutOfBoundsException
Exception 2 occurs

○ Compile-Time Error

Question - 9
Java String Pool

How many strings will be added to the string pool upon execution of the following code block?

```
public static void main(String[] args) {
        String str = "Hello There!";
        String mtr = "Hello" + " " + "There" + "!";
        String dtr = "Hello " + "There" + "!";
        String gtr = "Hello There" + "!";
    }
```

○ 1

○ 3

○ 5

○ 7

○ 9

Question - 10
Java Inheritance Overriding

What will be the output of the following code block?
Animal.java

```
public abstract class Animal
{
        public abstract void makenoise();
}
```

Dog.java

```
public class Dog extends Animal
{
    public void makenoise()
    {
        System.out.println("Make Dog Noise...");
    }
}
```

GoldenRetriever.java

```
public class GoldenRetriever  extends Dog
{
    public void makenoise()
 {
        super.makenoise();
        System.out.println("Make Golden Retriever Noise");
 }
    public static void main(String ... arg)
```

```
  {
        Dog dog=new GoldenRetriever();
        dog.makenoise();
  }
}
```

○ Make Dog Noise...

○ Make Golden Retriever Noise

○ Make Dog Noise...
Make Golden Retriever Noise

○ throws ClassCastException on runtime

Question - 11
Java BufferedReader

file1.txt

```
Hello

World
```

file2.txt

```
Hackerrank
```

What is the output of this code?

```
import java.io.*;
public class Main{
    public static void main(String args[]) throws IOException {
        String str1, str2, str=null;
        File file1 = new File("file1.txt");
        File file2 = new File("file2.txt");
        BufferedReader br1 = new BufferedReader(new FileReader(file1));
        BufferedReader br2 = new BufferedReader(new FileReader(file2));
        while ((str1 = br1.readLine()) != null)
        {
            str2 = br2.readLine();
            if(str2 != null)
            {
                str=str1+" "+str2;
                System.out.println(str);
            }
            br1.close();
        }
    }
}
```

○ Hello World
IOException: Stream closed

○ Hello Hackerrank

World

○ Hello World Hackerrank

○ Hello Hackerrank
IOException: Stream closed

Question - 12
Value of k After Function Runs

Which is true of the following program?

```
public class TestFirstApp {
        static void doIt(int x, int y, int m) {
        if (x == 5) {
                m=y;
        } else {
                m=x;
        }
}

public static void main(String[] args) {
        int i=6, j=4, k=9;
        TestFirstApp.doIt(i,j,k);
        System.out.print(k);
}
```

○ Doesn’t matter what the values of i and j are, the output will always be 5 .

○ Doesn’t matter what the values of k and j are, the output will always be 5 .

○ Doesn’t matter what the values of i and j are, the output will always be 9 .

○ Doesn’t matter what the values of k and j are, the output will always be 9 .

Question - 13
Method Overloading

```
public void foo(ArrayList<String> data)
{
    //some code
}
public void foo (ArrayList<Integer> data)
{
    //some code
}
public ArrayList<String> foo (ArrayList<String> data)
{
    //some code
}
private void foo(List<String> data)
{
    //some code
}
public void foo(ArrayList<String> data, boolean flag)
```

```
{
    //some code
}
```

Which of the statements are true regarding this code?

○ All the methods are considered as overloaded methods, except for private void foo(List data) because it has a private access modifier.

○ All methods can coexist without issue.

○ 'foo(ArrayList)' clashes with 'foo(ArrayList)', as both methods have the same erasure.

○ private void foo(List data) {} is an overloaded method, while public ArrayList foo (ArrayList data) {} is not.

Question - 14
Find the Output

Consider the following code:

```
int sum  = Stream.iterate(new int[]{0, 1},
tint sum  = Stream.iterate(new int[]{0, 1}, x -> new int[]{x[1], x[0] + x[1]})
                .limit(n)
                .map(x -> x[0])
                .collect(toList())
                .stream()
                .distinct()
                .filter(i ->  i % 2 == 0)
                .mapToInt(i->i).
                sum();
return sum;
```

What is the difference in results for *n=7* and *n=8*?

○ 2

○ 4

○ 8

○ 0

○ 11

Question - 15
Console Output

What is the output of this code?

```
Function<List<Integer>, Integer> function = x -> x
                .stream()
                .map(i-> i * 2)
                .mapToInt(i->i)
                .distinct()
                .sum();

Function<Integer, Integer> function2 = x -> x * 10;
Function<Integer, Integer> function3 = x -> x * 100;

int len = function.andThen(function2)
                .andThen(function3)
                .apply(Arrays.asList(1,2,2));
System.out.println(len);
```

○ 600

○ 60000

○ 3000

○ 6000

Question - 16

Java : Division Check

What is the result when one tries to compile and run the following code?

```
public final static void main(String[] args){
  double d = 10.0 / -0;
  if(d == Double.POSITIVE_INFINITY)
    System.out.println("Positive infinity");
  else
    System.out.println("Negative infinity");
}
```

○ Positive infinity

○ Negative infinity

○ Will fail to compile

○ Runtime exception

Question - 17

Java : OOPS

What feature allows different methods to have the same name and arguments type, but a different implementation is called?

○ overloading

○ overriding

○ Java does not permit methods with same name and type signature

○ None of the above

Question - 18
Java: Inner Classes

Examine the following Java code, it includes an inner class, what will be the output of the following code?

```
public final class Test4 {
        class Inner {
                void test() {
                        if (Test4.this.flag) {
                                sample();
                        }
                }
        }
        private boolean flag = true;
        public void sample() {
                System.out.println("Sample");
        }
        public Test4() {
                (new Inner()).test();
        }
        public static void main(String args []) {
                new Test4();
        }
}
```

○ Prints out "Sample"

○ Program produces no output but terminates correctly

○ Program does not terminate.

○ The program will not compile

Question - 19
Java Classes

Which statement is true?

○ Non-static member classes must have either default or public accessibility.

○ All nested classes can declare static member classes.

○ Methods in all nested classes can be declared static.

○ Static member classes can contain non-static methods.

Question - 20
Java: Constructor, Exception Handling and Inheritance

What should be the output of the following code?

```
class AirPlane {
    public AirPlane() throws IOException {
        System.out.print("AirPlane");
        throw new IOException();
    }
}
class AirJet extends AirPlane {
    public AirJet() throws IOException {
        System.out.println("Airjet");
        try {
            super();
        } catch (IOException e) {
            System.out.print("IOException is thrown in AirJet");
        }
    }
}
public class Tester {
    public static void main(String args[]) {
        try {
         new AirJet();
        } catch (IOException e) {
            System.out.print("IOException is thrown in Tester");
        }
    }
}
```

○ "AirPlaneIOException is thrown in AirJet" will be printed.

○ "AirPlaneIOException is thrown in AirJetIOException is thrown in Tester" will be printed.

○ "AirPlaneIOException is thrown in Tester" will be printed.

○ The code does not compile because of an error in the constructor of class Airjet.

Question - 21

Given the correct import(s) and the following code, what is the output?

Given the correct import(s) and the following code, what is the output?

```
class Limits
{
    private int x = 2;
    protected int y = 3;
    private static int x1 = 4;
    protected static int x2 = 5;

public static void main(String[] args)
{
    int x = 6; int y = 7;
    int x1 = 8, int x2 = 9;
    new Limits().new Secret().go();
}
class Secret
{
    void go()
    {
        System.out.println(x + " " + y + " " + x1 + " " + x2);
    }
}
}
```

○ 2 7 4 9

○ 2 3 4 5

○ 6 3 8 4

○ 6 7 8 9

Question - 22
Concurrent Threads

Given the code below:

```
class MyThread extends Thread {
    MyThread() {
        System.out.print(" MyThread");
    }

    public void run() {
        System.out.print(" bar");
    }

    public void run(String s) {
        System.out.print(" baz");
    }
}

public class TestThreads {
    public static void main(String[] args) {
        Thread t = new MyThread() {
            public void run() {
                System.out.print(" foo");
            }
        };
        t.start();
    }
}
```

What is the expected result?

○ foo

○ MyThread foo

○ MyThread bar

○ foo bar

○ foo bar baz

○ bar foo

○ Compilation fails

○ An exception is thrown at runtime

Question - 23
Java Concatenate

In main(), after creating an object of MyClass, which code option returns the string *s* concatenated with the sum of two numbers *x* and *y*?
For example, if *s = 'HackerRank', x = 2,* and *y = 3,* the output should be HackerRank 5

```
public interface MyInterface {
        String foo(String s, int x, int y);
    }
```

```
public class MyClass {
    public String foo2(MyInterface i, String s, int x, int y) {
        return i.foo(s, x, y);
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        MyClass myClass = new MyClass();
        // code to write
    }
}
```

○ String result = myClass.foo2((s, x, y) -> s + " " + x + y, "HackerRank", 2, 3); System.out.println(result);

○ String result = myClass.foo2((s, x, y) -> s + " " + (x + y), "HackerRank", 2, 3); System.out.println(result);

○ String result = myClass.foo2->(s + " " + x + y, "HackerRank", 2, 3); System.out.println(result);

○ String result = myClass.foo2((s, x, y) -> "HackerRank " + 2 + 3); System.out.println(result);

Question - 24
Similar Classes

Janitor and *Cashier* classes have a *print* function that is similar. How can we refactor this code to reduce repetition?

```
class Janitor {
 ...
 print(){
 console.log("Employee Name: " + self.name)
 }
}
class Cashier {
 ...
 print(){
 console.log("Employee Name: " + self.name)
 }
}
```

○ Replace 'self.name' with a getter method to encapsulate the member variable.

○ Replace the 'print' method with a 'toString' method. Use this new method to display data to the console.

○ Move the print methods from both the classes to a superclass.

○ Create a 'print' function outside both classes and call that function in the print method of both classes.

Question - 25
Java Generics

What is the result of compiling and/or running this code?

```
public class Generic<T> {
    private T value;
    public Generic(T value) {
        this.value = value;
    }
    public T getValue() {
        return value;
    }
}
```

```
import java.util.ArrayList;
public class Main {
    public static void main(String args[]) {
       ArrayList<Generic>  g = new ArrayList<>();
       Generic<?> g1=new Generic<>(10);
       Generic<?> g2=new Generic<>("Hello");
       g.add(g1);
       g.add(g2);
       int i=g.get(0).getValue();
       String s=g.get(1).getValue();
       System.out.println(s);
       System.out.println(i);
    }
}
```

○ Compile-time error while retrieving the values from array list and assigning them to their respective datatypes.

○ Compile-time error while creating the generic object with int value, because generics does not work with primitive data type int

○ Hello 10

○ Run-time error because an array list cannot contain generic objects of different types (string and int)

Question - 26
Field Names in Classes

A public data member with the same name is provided in the base and its derived classes. Which of the following is true?

○ It is a compiler error to provide a field with the same name in both the base and derived classes.

○ The program will compile, and this feature is called 'overloading'.

○ The program will compile, and this feature is called 'overriding'.

○ The program will compile, and this feature is called 'hiding' or 'shadowing'.

Question - 27
Which of the following are true about the Error and Exception classes?

Which of the following are true about the Error and Exception classes?

○ Both classes extend Throwable.

○ The Error class is final and the Exception class is not.

○ The Exception class is final and the Error is not.

○ Only the Exception class extends Throwable

Question - 28
Empty Collection

Which code displays an empty (`[]`) collection when executed?

○

```
Stack<Integer> stack = new Stack<>();
Queue queue = new LinkedList();

for (int i = 0; i < 10; i++)
    stack.add(i);

for (int i = 0; i < stack.size() - 1; i++) {
    int n = stack.remove(0);
    queue.add(n);
}

for (int i = 0; i < queue.size() - 1; i++) {
    int n = (int) queue.remove();
    i = i - 1;
}

System.out.println(queue);
```

○

```
Stack<Integer> stack = new Stack<>();
Queue queue = new LinkedList();

for (int i = 0; i < 10; i++)
    stack.add(i);

for (int i = 0; i < stack.size(); i++) {
    int n = stack.remove(0);
    queue.add(n);
}

for (int i = 0; i < queue.size(); i++) {
    int n = (int) queue.remove();
}

System.out.println(queue);
```

○

```
Stack<Integer> stack = new Stack<>();
Queue queue = new LinkedList();

for (int i = 0; i < 10; i++)
    stack.add(i);

for (int i = 0; i < stack.size(); i++) {
    int n = stack.remove(0);
    queue.add(n);
    i = i - 1;
}

for (int i = 0; i < queue.size(); i++) {
    int n = (int) queue.remove();
}

System.out.println(queue);
```

○

```
Stack<Integer> stack = new Stack<>();
Queue queue = new LinkedList();

for (int i = 0; i < 10; i++)
    stack.add(i);

for (int i = 0; i < stack.size(); i++) {
    int n = stack.remove(0);
    queue.add(n);
    i = i - 1;
}

for (int i = 0; i < queue.size(); i++) {
    int n = (int) queue.remove();
    i = i - 1;
}

System.out.println(queue);
```

Question - 29
Identify Runtime Exceptions

Which of the following is **not** an example of RuntimeException ?

○ NullPointerException

○ ArrayIndexOutOfBoundsException

○ FileNotFoundException

Question - 30
Java : Collections

Which of the statements is true about ArrayList and Vector in Java?

○ Vector can be resized while ArrayList cannot be

○ Vector is synchronized while ArrayList is not

○ ArrayLists can grow but cannot shrink in size, while Vector can both grow and shrink

○ Vectors allow duplicate values while ArrayList doesnot

Question - 31
Best Three Scores

int [] scores = {1,3,8,3,5,6,2,4}

Which code returns the largest three numbers from the *scores* array?

○ List<Integer>bestScore = Arrays .stream(scores) .boxed() .sorted() .skip(5) .collect(Collectors.toList());

○ List<Integer>bestScore = Arrays .stream(scores) .boxed() .sorted() .skip(3) .collect(Collectors.toList());

○ List<Integer>bestScore = Arrays .stream(scores) .sorted() .skip(5) .collect(Collectors.toList());

○ List<Integer>bestScore = IntStream.of(scores) .sorted() .boxed() .skip(4) .collect(Collectors.toList());

Question - 32
Java Garbage Collection

Which of the following statements are true about garbage collection in Java?

○ The garbage collector can only free memory that was allocated using the new keyword.

○ The garbage collector can free memory that is being referenced by a static variable.

○ The system.gc() method can be used to guarantee that garbage collection will occur.

○ The finalize() method can be used to prevent an object from being garbage collected.

Question - 33
Spring AOP

LoggerAspect.java

```
@Aspect
@Component
@Slf4j
public class LoggerAspect {
    @Around("execution(public * *(..)) && within(com.hackerrank.api.*Controller*)")
    private Object logAroundEveryPublicMethod(ProceedingJoinPoint pjp) throws Throwable {
        String builder = "\n-----------beginning around advice---------"
                + "\narguments: "
                + Arrays.stream(pjp.getArgs()).filter(Objects::nonNull)
                .map(Object::toString).map(param -> param.length() >= 200 ? param.substring(0, 200) : param)
                .collect(Collectors.toList())
                + "\nmethod signature: " + pjp.getSignature()
                + "\n---------around advice concluded---------\n";
        log.info(builder);
        return pjp.proceed();
    }
}
```

Which of the following is true about the LoggerAspect class's functionality?

○ It will log every public class method execution with parameter names.

○ It will log every public class method execution with parameter values and method names.

○ It will log every public class whose name ends with Controller, method execution with parameter names.

○ It will log all parameters and method names of public method execution of classes that contain Controller in their name in the com.hackerrank.api package.

Question - 34
Spring Annotation

What does the annotation in this Spring-based code segment do?

```
@Component
public class ComponentExample {
  int x;
  public void show(){
    System.out.println("Hello");
  }
}
```

○ It allows Spring to create containers.

○ It allows Spring to create components.

○ It allows Spring to collect bean instances.

○ It allows Spring to detect custom beans automatically.

Question - 35
Dispatcher Servlet

What is the purpose of Dispatcher Servlet in Spring Boot ?

○ It is used for transaction management.

○ It is used for database connection pool management.

○ It is used to handle all HTTP requests and responses.

○ It is used heavily for dependency injection.

Question - 36
Dependency Injection

Which of the following options can be used for dependency injection in Spring Boot?

○ Setter injection

○ jsonConfiguration

○ Constructor injection

○ @Autowired

Question - 37
Spring AOP Usage

Consider the following code.

NotifierMetricLogger.java

```
package spring.listener;

import org.aspectj.lang.annotation.Around;
import org.aspectj.lang.annotation.Aspect;
import org.aspectj.lang.annotation.Before;
import org.aspectj.lang.annotation.AdviceName;
import org.springframework.stereotype.Component;
import org.aspectj.lang.JoinPoint;
import org.aspectj.lang.ProceedingJoinPoint;
import java.util.logging.Logger;

//X
@Component
public class NotifierMetricLogger {
    private static final Logger log = Logger.getLogger(NotifierAspect.class.getName());
    //Y
    public Object beforeNotifyLogging(ProceedingJoinPoint joinPoint) throws Throwable {
        long startDate = System.currentTimeMillis();
        Object proceed = joinPoint.proceed();
        long executionTime = System.currentTimeMillis() - startDate;
        log.info("Notify process time :" + executionTime);
        return proceed;
    }
}
```

TwitterNotifier.java

```
package spring.service.impl;

import spring.service.Notifier;
import org.springframework.stereotype.Component;
import java.util.logging.Logger;

@Component
public class TwitterNotifier implements Notifier {
    private static final Logger log = Logger.getLogger(TwitterNotifier.class.getName());
    @Override
    public void notify(String message) {
        log.info("TwitterNotifier: " + message);
        //send notification to home page
    }
}
```

Notifier.java

```
package spring.service;

public interface Notifier {
    void notify(String message);
}
```

Assuming the Spring Boot application is configured to use AOP with @EnableAspectJAutoProxy(proxyTargetClass = true) annotation, to capture *TwitterNotifier.notify(String message)* method's process time, which of the following options should be placed in the *X* and *Y* positions in NotifierMetricLogger.java?

○ X = @Aspect
Y = @Before("execution(* spring.service.impl.*.notify(..))")

○ X = @AdviceName("NotifierMetricLogger")
Y = @Around("execution(* spring.service.impl.*.notify(..))")

○ X = @AdviceName("NotifierMetricLogger")
Y = @Before("execution(* spring.service.impl.*.notify(..))")

○ X = @Aspect
Y = @Around("execution(* spring.service.Notifier.notify(..))")

Question - 38
Path Variable

In the Spring MVC controller, which of these are valid uses of the @PathVariable annotation?

○ @RequestMapping(value="/users/{userId}/addresses/{addressId}") public String viewUserAddress(@PathVariable String userId, @PathVariable String addressId, Model m)

○ @RequestMapping(value="/users/{userId}") public String viewUser(@PathVariable("users") String user, Model m)

○ @RequestMapping(value="/users/{userId}") public String viewUser(@PathVariable String userId, Model m)

○ @RequestMapping(value="/users/{userId}") public String viewUser(@PathVariable("userId") String personnelId, Model m)

Question - 39
Spring Dependency Injection

```
@Service
public ProductService {
    private ProductRepository productRepository;
    private ProductMapper productMapper;

    @Autowired
    public ProductService(ProductRepository productRepository,
                          ProductMapper productMapper){
        this.productRepository = productRepository;
        this.productMapper = productMapper;
    }

}
```

Which type of injection is implemented in the ProductService?

○ Setter

○ Getter

○ Property

○ Construction

Question - 40
Spring IOC Container

Which of the following options act as a Spring IOC Container?

○ ApplicationContext

○ DispatcherServlet

○ SpringRunner

○ BeanFactory

○ None of the above

Question - 41
Springboot: Music Streaming

This project is a practical assessment designed to evaluate understanding and skills in working with RESTful APIs using Spring Boot.

Below are examples of the JSON data for this project.:

Example of Artist data JSON Object

```
{
  "id": 1,
  "artistName": "Alex Doe",
  "bio": "Alex Doe is a popular artist...",
  "genre": "Pop",
  "origin": "USA",
  "formedYear": "2000",
  "socialLink": "https://www.alexdoe.com",
  "image": "https://www.alexdoe.com/image.jpg",
  "tracksProduced": [
    {
      "id": 1,
      "title": "Track 1",
      "albumName": "Album 1",
      "releaseDate": "2022-01-01",
      "duration": "3:30",
      "genre": "Pop",
      "description": "This is a description of Track 1",
      "playCount": 1000,
      "fileUrl": "https://www.alexdoe.com/track1.mp3",
      "coverImage": "https://www.alexdoe.com/track1.jpg"
    }
  ]
}
```

Example of Playlist data JSON Object

```
{
  "id": 1,
  "name": "My Playlist",
  "description": "This is my favorite playlist",
  "tracks": [
    {
      "id": 1,
      "name": "Track 1",
      "duration": "3:45",
      "artist": {
        "id": 1,
        "name": "Artist 1"
      }
    },
    {
      "id": 2,
      "name": "Track 2",
      "duration": "4:30",
      "artist": {
        "id": 2,
        "name": "Artist 2"
      }
    }
  ]
}
```

Example of Track data JSON Object

```
{
  "id": 1,
  "title": "Track 1",
  "albumName": "Album 1",
  "releaseDate": "2022-01-01",
  "duration": "3:30",
  "genre": "Pop",
  "description": "This is a description of Track 1",
  "playCount": 1000,
  "fileUrl": "https://www.alexdoe.com/track1.mp3",
  "coverImage": "https://www.alexdoe.com/track1.jpg",
  "artist": {
    "id": 1,
    "artistName": "Alex Doe",
    "bio": "Alex Doe is a popular artist...",
    "genre": "Pop",
    "origin": "USA",
    "formedYear": "2000",
    "socialLink": "https://www.alexdoe.com",
    "image": "https://www.alexdoe.com/image.jpg"
  }
}
```

Implement the following APIs:

ArtistController

- *POST /music/platform/v1/artists* - Create a new artist.

- *GET /music/platform/v1/artists* - Get all artists.
- *GET /music/platform/v1/artists/{artistId}* - Get an artist by ID.
- *PUT /music/platform/v1/artists/{artistId}* - Update an artist by ID.
- *DELETE /music/platform/v1/artists/{artistId}* - Delete an artist by ID.

PlayListController

- *GET /music/platform/v1/playlists/{playlistId}* - Get a playlist by ID.
- *POST /music/platform/v1/playlists* - Create a new playlist.
- *DELETE /music/platform/v1/playlists/{playlistId}* - Delete a playlist by ID.

TrackController

- *GET /music/platform/v1/tracks* - Get all tracks.
- *POST /music/platform/v1/tracks* - Create a new track.
- *GET /music/platform/v1/tracks/{trackId}* - Get a track by ID.
- *PUT /music/platform/v1/tracks/{trackId}* - Update a track by ID.
- *DELETE /music/platform/v1/tracks/{trackId}* - Delete a track by ID.

Complete the project so that it passes all the test cases when running the provided unit tests. The project, by default, supports the use of the H2 database.

▼ Example Requests and Responses

POST request to /music/platform/v1/artists

Request data JSON Object:

```
{
  "artistName": "Alex Doe",
  "bio": "Alex Doe is a popular artist...",
  "genre": "Pop",
  "origin": "USA",
  "formedYear": "2000",
  "socialLink": "https://www.alexdoe.com",
  "image": "https://www.alexdoe.com/image.jpg"
}
```

Response JSON Object:

```
{
  "id": 1,
  "artistName": "Alex Doe",
  "bio": "Alex Doe is a popular artist...",
  "genre": "Pop",
  "origin": "USA",
  "formedYear": "2000",
  "socialLink": "https://www.alexdoe.com",
  "image": "https://www.alexdoe.com/image.jpg"
}
```

GET request to /music/platform/v1/artists

Response JSON Object:

```
[
  {
    "id": 1,
    "artistName": "Alex Doe",
    "bio": "Alex Doe is a popular artist...",
    "genre": "Pop",
    "origin": "USA",
    "formedYear": "2000",
    "socialLink": "https://www.alexdoe.com",
    "image": "https://www.alexdoe.com/image.jpg"
  }
]
```

GET request to /music/platform/v1/artists/{artistId}

Response JSON Object:

```
{
  "id": 1,
  "artistName": "Alex Doe",
  "bio": "Alex Doe is a popular artist...",
  "genre": "Pop",
  "origin": "USA",
  "formedYear": "2000",
  "socialLink": "https://www.alexdoe.com",
  "image": "https://www.alexdoe.com/image.jpg"
}
```

PUT request to /music/platform/v1/artists/{artistId}

Request data JSON Object:

```
{
  "artistName": "Alex Doe",
  "bio": "Alex Doe is a popular artist...",
  "genre": "Pop",
  "origin": "USA",
  "formedYear": "2000",
  "socialLink": "https://www.alexdoe.com",
  "image": "https://www.alexdoe.com/image.jpg"
}
```

Response JSON Object:

```
{
  "id": 1,
  "artistName": "Alex Doe",
  "bio": "Alex Doe is a popular artist...",
  "genre": "Pop",
  "origin": "USA",
  "formedYear": "2000",
  "socialLink": "https://www.alexdoe.com",
  "image": "https://www.alexdoe.com/image.jpg"
}
```

DELETE request to /music/platform/v1/artists/{artistId}

No response data. Returns a 204 No Content status.

POST request to /music/platform/v1/playlists:

Request data JSON Object:

```
{
  "name": "My Playlist",
  "description": "This is my favorite playlist"
}
```

Response JSON Object:

```
{
  "id": 1,
  "name": "My Playlist",
  "description": "This is my favorite playlist"
}
```

GET request to /music/platform/v1/playlists/{playlistId}

Response JSON Object:

```
{
  "id": 1,
  "name": "My Playlist",
  "description": "This is my favorite playlist"
}
```

DELETE request to /music/platform/v1/playlists/{playlistId}

No response data. Returns a 204 No Content status.

POST request to /music/platform/v1/tracks

Request data JSON Object:

```
{
  "title": "Track 1",
  "albumName": "Album 1",
  "releaseDate": "2022-01-01",
  "duration": "3:30",
  "genre": "Pop",
  "description": "This is a description of Track 1",
  "playCount": 1000,
  "fileUrl": "https://www.alexdoe.com/track1.mp3",
  "coverImage": "https://www.alexdoe.com/track1.jpg"
}
```

Response JSON Object:

```
{
  "id": 1,
  "title": "Track 1",
  "albumName": "Album 1",
  "releaseDate": "2022-01-01",
  "duration": "3:30",
  "genre": "Pop",
  "description": "This is a description of Track 1",
  "playCount": 1000,
  "fileUrl": "https://www.alexdoe.com/track1.mp3",
  "coverImage": "https://www.alexdoe.com/track1.jpg"
}
```

GET request to /music/platform/v1/tracks

Response JSON Object:

```
[
  {
    "id": 1,
    "title": "Track 1",
    "albumName": "Album 1",
    "releaseDate": "2022-01-01",
    "duration": "3:30",
    "genre": "Pop",
    "description": "This is a description of Track 1",
    "playCount": 1000,
    "fileUrl": "https://www.alexdoe.com/track1.mp3",
    "coverImage": "https://www.alexdoe.com/track1.jpg"
  }
]
```

GET request to /music/platform/v1/tracks/{trackId}

Response JSON Object:

```
{
  "id": 1,
  "title": "Track 1",
  "albumName": "Album 1",
  "releaseDate": "2022-01-01",
  "duration": "3:30",
  "genre": "Pop",
  "description": "This is a description of Track 1",
  "playCount": 1000,
  "fileUrl": "https://www.alexdoe.com/track1.mp3",
  "coverImage": "https://www.alexdoe.com/track1.jpg"
}
```

PUT request to /music/platform/v1/tracks/{trackId}
Request data JSON Object:

```
{
  "title": "Track 1",
  "albumName": "Album 1",
  "releaseDate": "2022-01-01",
  "duration": "3:30",
  "genre": "Pop",
  "description": "This is a description of Track 1",
  "playCount": 1000,
  "fileUrl": "https://www.alexdoe.com/track1.mp3",
  "coverImage": "https://www.alexdoe.com/track1.jpg"
}
```

Response JSON Object:

```
{
  "id": 1,
  "title": "Track 1",
  "albumName": "Album 1",
  "releaseDate": "2022-01-01",
  "duration": "3:30",
  "genre": "Pop",
  "description": "This is a description of Track 1",
  "playCount": 1000,
  "fileUrl": "https://www.alexdoe.com/track1.mp3",
  "coverImage": "https://www.alexdoe.com/track1.jpg"
}
```

DELETE request to /music/platform/v1/tracks/{trackId}
No response data. Returns a 204 No Content status.

Question - 42
Zig-Zag Array

A zig-zag array alternates between elements being less than and greater than their adjacent elements. Transform an array of integers into a zig-zag array with the minimum number of replacements.

You can replace any element with any integer (positive, negative, or zero). Your task is to determine the minimum number of replacements needed.

Example
arr = [1, 2, 3, 4, 5]

There are two possible patterns:

1. LHLHL pattern (starting with a lower value):
   - [1, 2, ?, 4, ?]
   - Need to replace the 3 and 5 (2 replacements)
2. HLHLH pattern (starting with a higher value):
   - [?, 2, 3, ?, 5]
   - Need to replace the 1 and 4 (2 replacements)

Both patterns require 2 replacements, so the answer is 2.

Function Description

Complete the function *minOperations* in the editor with the following parameters:

int *arr[n]:* an array of integers

Returns

int: the minimum number of operations required to turn *arr* into a zig-zag array

Constraints

- $1 \le n \le 10^5$
- $1 \le arr[i] \le 10^9$

▼ Input Format For Custom Testing

The first line contains an integer, *n*, the number of elements in *arr*.

Each line *i* of the *n* subsequent lines (where $0 \le i < n$) contains an integer, *arr[i]*.

▼ Sample Case 0

Sample Input For Custom Testing

```
STDIN     Function
-----     --------
8     →   n = 8
2     →   arr = [2, 1, 2, 3, 4, 5, 2, 9]
1
2
3
4
5
2
9
```

Sample Output

```
2
```

Explanation

```
Original: [2, 1, 2, 3, 4, 5, 2, 9]
           L  H  L  H  L  H  L  H
LHLHLHLH: [2, +, 2, 3, -, 5, 2, 9]

           H  L  H  L  H  L  H  H
HLHLHLHL: [2, 1, 2, -, 4, -, 2, -]
```

For the LHLH... pattern, replace the second value (1) with a number greater than 2 and the fifth value (4) with a number less than 3.

For the HLHL... pattern, replace the fourth value (3) with a number less than 2, and the sixth value (5) and the eighth value (9) with a number less than 2.

The LHLH... pattern only requires two replacements.

▼ Sample Case 1

Sample Input For Custom Testing

```
STDIN     Function
-----     --------
6    →    arr[] size n = 6
1    →    arr = [1, 2, 4, 4, 5, 6]
2
4
4
5
6
```

Sample Output

```
2
```

Explanation

```
Original: [1, 2, 4, 4, 5, 6]
           L  H  L  H  L  H
LHLHLHLH: [1, 2, -, 4, -, 6]

           H  L  H  L  H  L
HLHLHLHL: [+, 2, 4, -, 5, -]
```

Starting with a low value takes 2 replacements, while starting with a high value takes 3. Return 2.

Question - 43
Java: Log File Analyzer

Develop a function to process large log files. Each log entry is a string that contains a timestamp, an event type, and a message, formatted as "*[timestamp] [event_type] [message]*". The goal is to quickly retrieve a list of event messages among the *n* number of log entries, that occurred between *startTime* and *endTime*, inclusive, sorted by timestamp.

Example

Input:

```
logEntries = [
  "2023-06-21 14:35:00 INFO Application started successfully",
  "2023-06-21 15:00:00 ERROR Failed to connect to database",
  "2023-06-21 15:05:00 WARNING Low memory usage detected"
]
startTime = "2023-06-21 15:00:00"
endTime = "2023-06-21 16:00:00"
```

Output:

```
Failed to connect to database
Low memory usage detected
```

Explanation:

The second and third log entries lie within the time range. Thus, the messages for those entries ("Failed to connect to database" and "Low memory usage detected") are returned in order of their timestamps.

Function Description

Complete the function *getEventsBetween* in the editor below.

Function Parameters

List<String> logEntries: the list of event strings
String startTime: the start of the interval
String endTime: the end of the interval

Returns

List<String>: messages from all distinct events that occurred between *startTime* and *endTime*, inclusive and sorted by timestamp.

Constraints

- The *timestamp* is in the format YYYY-MM-DD HH:MM:SS.
- The *event_type* is a single word without spaces.
- The message may contain spaces and other non-alphanumeric characters, but it is always contained in one line.
- $1 \leq n < 10^5$
- The *startTime* and *endTime* are in the format YYYY-MM-DD HH:MM:SS.
- 2023-01-01 00:00:00 < *startTime* < 2024-12-31 00:00:00

- 2023-01-01 00:00:00 < *endTime* < 2024-12-31 00:00:00

▼ Input Format for Custom Testing

The first line contains an integer, *n*, the size of the string list.

Each of the next *n* lines contains a string, *logEntries[i]*.

The next line contains a string, *startTime*.

The next line contains a string, *endTime*.

▼ Sample Case 0

Sample Input 0

```
STDIN                                                       Function
-----                                                       --------
5                                                           size of LogEntries[] n = 5
2023-06-21 14:35:00 INFO Application started successfully  log entries
2023-06-21 15:00:00 ERROR Failed to connect to database
2023-06-21 15:05:00 WARNING Low memory usage detected
2023-06-21 13:59:00 DEBUG Initializing system components
2023-06-21 16:01:00 INFO Scheduled maintenance completed
2023-06-21 14:00:00                                          startTime = "2023-06-21 14:00:00"
2023-06-21 16:00:00                                          endTime = "2023-06-21 16:00:00"
```

Sample Output 0

```
Application started successfully
Failed to connect to database
Low memory usage detected
```

Explanation

The first three log entries are within the time range, and their messages are returned.

▼ Sample Case 1

Sample Input 1

```
3
2023-07-04 12:00:00 INFO Lunch break started
2023-07-04 09:00:00 DEBUG System check performed
2023-07-04 18:00:00 ERROR Unexpected server reboot
2023-07-04 08:00:00
2023-07-04 20:00:00
```

Sample Output 1

```
System check performed
Lunch break started
Unexpected server reboot
```

Explanation

All log entries lie within the time range, and their messages are returned.