Question - 1

SCORE: 5 points

JSP Implicit Objects

Java | Implicit Objects | Medium | JSP

Consider the following *HttpServlet* class, *TestServlet,* and its corresponding JSP file, *test.jsp*.

TestServlet

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;

public class TestServlet extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        request.setAttribute("testAttr", "Test Attribute");
        RequestDispatcher view = request.getRequestDispatcher("test.jsp");
        view.forward(request, response);
    }
}
```

test.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
    <body>
        <!-- Code block -->
    </body>
</html>
```

Which of the following code snippets can replace the *Code block* in *test.jsp* to retrieve and print the value of the *testAttr* attribute from the request?

- <%= request.getAttribute("testAttr") %>
- ${pageContext.request.getAttribute("testAttr")}
- <%= application.getAttribute("testAttr") %>
- ${requestScope.testAttr}

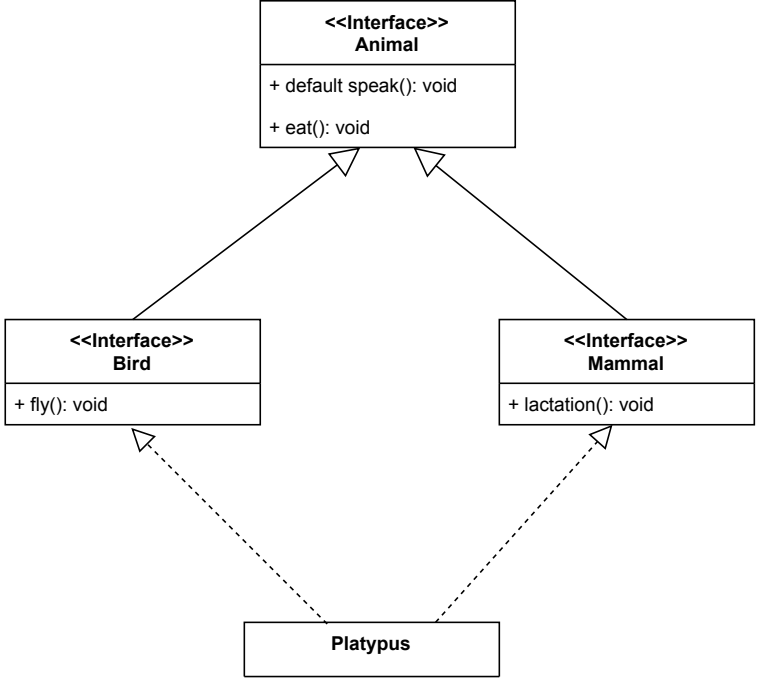
Question - 2

SCORE: 5 points

Interfaces

Interfaces | Classes | Medium | Inheritance

Consider the following diagram



Which of the following statements are true regarding it?

- The Platypus class must implement all the methods.
- The diagram represents a diamond problem.
- The Platypus class only have to implement the eat(), fly(), and lactation() method.
- The Platypus class cannot extend multiple interfaces.

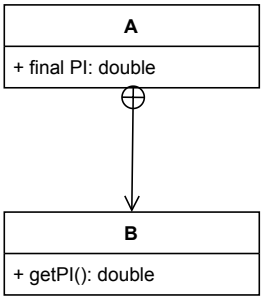
Question - 3

SCORE: **5 points**

Inner Class, Static and Final

Medium | inner class | static | final

Consider the following UML class diagram
Note - Class B is *private*.



What is the correct way to access the value of PI through the *getPI()* method in the main?

```
A.B inner = new B();
double piValue = inner.getPI();
System.out.println("The value of PI is: "+piValue);
```

A.B inner = new B(); double piValue = inner.getPI(); System.out.println("The value of PI is: "+piValue);

```
B inner = new B();
double piValue = B.getPI();
System.out.println("The value of PI is: "+piValue);
```

First, class B must be made public then we can use the following code.

```
A outerObj = new A();
A.B innerObj = outerObj.new B();
double piValue = A.B.getPI();
System.out.println("The value of PI is: "+piValue);
```

First, class B should be made public, and PI should also be declared as static then, we can use the following code.

```
A outerObj = new A();
A.B innerObj = outerObj.new B();
double piValue = A.B.getPI();
System.out.println("The value of PI is: "+piValue);
```

A outerObj = new A();
A.B innerObj = outerObj.new B(); double piValue = A.B.getPI(); System.out.println("The value of PI is: "+piValue);

Question - 4

SCORE: **5 points**

Polymorphism

Medium | Polymorphism

```
class Parent {
    void print(String text) {
        System.out.println("String: " + text);
    }

    void print(int number) {
        System.out.println("Integer: " + number);
    }
}
class Child extends Parent
{
    void print(int number)
    {
        System.out.println("I am a child" + number);
    }
    void print(float number)
    {
        System.out.println("Float: " + number);
    }

}
public class Main{
    public static void main(String[] args) {
        Parent p = new Parent();
        p.print(5);
        p=new Child();
        p.print(2.3f);
```

```
    }
}
```

Which of the following statements are true regarding this code?

- The code demonstrates only compile-time polymorphism.
- The code will run successfully.
- The code demonstrates both compile-time and run-time polymorphism.
- In the last line, we need downcasting in order to call the print() method with a floating point value.

Question - 5

SCORE: **5 points**

Java Garbage Collection

Garbage Collection | Medium

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 - 6

SCORE: **5 points**

Java Generics

Generics | Medium

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 - 7

SCORE: **5 points**

Empty Collection

Java | Collections | Queue | Medium | Stacks

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 - 8

SCORE: **5 points**

Java Streams

Java | Medium | Stream

Consider the following code

```
class Weather
{
    String place;
    Double temperature;

    public Weather()
    {

    }

    public Weather(String place, Double temperature)
    {
```

```
        this.place = place;
        this.temperature = temperature;
    }

    public Double getTemperature()
    {
        return temperature;
    }

    public String getPlace()
    {
        return place;
    }

    public String toString()
    {
        return new StringBuffer(" Place : ")
                .append(this.place)
                .append(" Temperature : ")
                .append(this.temperature)
                .toString();
    }
}
```

```
List<Weather> weathers = new ArrayList<>();
weathers.add(new Weather("Sunny", 33.0));
weathers.add(new Weather("Rainy", 17.0));
weathers.add(new Weather("Cloudy", 23.0));
weathers.add(new Weather("Cold", 3.0));
weathers.add(new Weather("Hot", 37.0));
weathers.add(new Weather("Windy", 13.0));
weathers.add(new Weather("Snowy", 0.0));
weathers.add(new Weather("Freezing", -15.0));

// sort & print code block
```

Which of the following options will display the output after sorting the objects by temperature?

- weathers.stream() .map(Weather::getTemperature) .sorted() .forEach(System.out::println);
- weathers.stream() .sorted(Weather::getTemperature) .forEach(System.out::println);
- weathers.stream() .sorted((p1, p2) -> p1.getTemperature().compareTo(p2.getTemperature())) .forEach(System.out::println);
- weathers.stream() .map(Weather::getTemperature) .sorted((p1, p2) -> p1.compareTo(p2)) .forEach(System.out::println);

Question - 9

SCORE: 5 points

Find the Output

Java | Java 8 | Stream | Medium

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 - 10

SCORE: **5 points**

Console Output

Medium | Java | Java 8 | Stream | Functions

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 - 11

SCORE: **5 points**

Best Three Scores

Java | Java 8 | Stream | Medium

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 - 12

SCORE: **5 points**

Even Numbers

Java 8 | Java | Stream | Medium

The following set of operations is performed on an array.

1. Select even numbers
2. Multiply each of them by 2
3. Subtract 1 from each
4. Select all the numbers that are greater than 4

Which of the following code blocks represents these operations?

```
Collection&lt;Integer&gt;c = Arrays.asList(1,2,3,4,5);
List&lt;Integer&gt; list = c.stream() .filter(i -&gt; i % 2 == 0) .map(i -&gt; i -
1)
.filter(i -&gt; i &gt; 4) .collect(Collectors.toList());
```

```
Collection<Integer>c = Arrays.asList(1,2,3,4,5);
List<Integer> list = c.stream() .filter(i -> i % 2 == 0).map(i -> i = i * 2-1)
.filter(i -> i > 4)
.collect(Collectors.toList());
```

```
Collection<Integer>c = Arrays.asList(1,2,3,4,5);
List<Integer> list = c.stream() .filter(i -> i % 2 == 0).filter(i -> i = i * 2-1)
.filter(i -> i > 4) .collect(Collectors.toList());
```

```
Collection<Integer>c = Arrays.asList(1,2,3,4,5);
List<Integer> list = c.stream() .filter(i -> i % 2 == 0).filter(i -> i = i * 2-1)
.map(i -> i > 4) .collect(Collectors.toList());
```

Question - 13

SCORE: 5 points

Java : Collections

Medium | Java | Collections | Arrays | Vectors | Language Proficiency | Problem Solving

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 - 14

SCORE: 5 points

Identify Runtime Exceptions

Java | Exception Handling | Programming | Medium

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

- NullPointerException
- ArrayIndexOutOfBoundsException
- FileNotFoundException

Question - 15

SCORE: 5 points

Empty a HashMap

Java | Collections | Hash Map | Medium

To delete all pairs of keys and values in a given HashMap, which of the following methods should be used ?

- clearAll()
- empty()
- remove()
- clear()

Question - 16

SCORE: 5 points

The Keyword Synchronized

Java | Synchronization | Medium

The keyword synchronized can be used in which of the following types of blocks:

- [x] Instance methods
- [x] Static methods
- [ ] Static classes
- [x] Code blocks inside static methods

Question - 17

SCORE: **5 points**

Java : OOPS

Java | Overloading | OOPS | Polymorphism | Language Proficiency | Problem Solving | Programming | Medium

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

- [ ] overloading
- [x] overriding
- [ ] Java does not permit methods with same name and type signature
- [ ] None of the above

Question - 18

SCORE: **5 points**

Which of the following is true about iterators ?

Java | Collections | Medium

Which of the following is true about iterators ?

- [x] Iterator is an interface
- [ ] Iterator is a member function of a class in the library
- [x] Iterators are used to traverse through the elements of a collection
- [ ] Iterator is an abstract class used for iterating all the elements of the class

Question - 19

SCORE: **5 points**

Java Classes

Medium | Java | OOPS | Programming

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

SCORE: **5 points**

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

Application Development | Java | Easy

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