Q text	Option/ Answer 1	Option/ Answer 2	Option/ Answer 3	Option/ Answer 4	Option/ Answer 5	Correct Answer
Which of the following are features of Java Programming Language?	Robust	Multithreaded	Interpreted	All of the above		4

Given:
 	
public class Main {
float result;
 	
public static void main(String[] args) {
result = 5 / 2;
System.out.println(result);
}
}
 	
What will be the output?
	2	2.0	2.5	Compile time error		4

Given:
public class Demo { //Line 1
public static void main (String[] args) { //Line 2
byte var1 = 127; //Line 3
byte var2 = 126; //Line 4
byte result = var1 + var2; //Line 5
}
}
 	
Which statement is true?
	Compilation succeeds and d takes the value 253.	Line 5 contains an error that prevents compilation.	Line 5 throws an exception indicating "Out of range"	Line 3 and 4 contain errors that prevent compilation.		2

Given the following code fragment:
 	
XXXX choice ; // variable choice is declared and initialized here
switch(choice) {
case 100 : System.out.println("One hundred");break ;
case 20 : System.out.println("Twenty");break ;
case 30 : System.out.println("Thirty");break ;
}
 	
Choose the declarations of choice which will not cause a compiler error:
	byte choice = 100 ;	short choice = 100 ;	int choice = 300 ;	All of the above		4

What all gets printed when the following code is compiled and run?
 	
public class test {
  public static void main(String args[]) {
  for(int i = 0; i < 2; i++) {
  for(int j = 2; j>= 0; j--) {
  if(i == j) break;
  System.out.println("i=" + i + " j="+j);
  }
  }
  }
}
 	
1. i=0 j=0
2. i=0 j=1
3. i=0 j=2
4. i=1 j=0
5. i=1 j=1
	'1,3,5	'2,4,6	'1,2,5	'3,2,6		4

Given :
public class Test {
 public static void main(String args[]) {
 int x = 12;
 while (x < 10) {
 x--;
 }
 System.out.print(x); //line 7
 }
}
 	
What is the result?
	0	10	12	Line 7 will never be reached.		3

Given :
public class Test {
 public static void main(String args[]) {
 for (int i = 0; i <= 10; i++) {
 if (i > 6)
 break;
 }
 System.out.println(i);
 }
}
 	
What is the result?
	6	7	10	Compile time error		4

What will be the output after compiling and running following code?
 	
public class Test{
  public static void main(String args[]){
  int x =5;
  x *= 3 + 7;
  System.out.println(x);
 }
}
	22	50	10	Compilation fails		2

What will be the output after compiling and running following code?
 	
public class Test{
public static void main(String[] args){
int a=5 , b=6, c=7;
System.out.println("Value is "+ b + c);
System.out.println(a + b + c);
System.out.println("String " + (b+c));
}
}
	Value is 67 18 String 13	Value is 13 18 String 13	Value is 13 18 String	Compilation fails		1

Automatic type conversion in Java takes place when:	Two type are compatible and size of destination type is shorter than source type	Two type are compatible and size of destination type is equal of source type.	Two type are compatible and size of destination type is larger than source type	All of the above		3
Which of the following statements are true?	No arg constructor is allways supplied by the Compiler	Constructors cannot be overloaded	Constructors cannot have return type.	Constructors can be static		3

What is true about the following code?
 	
enum EnumDemo {
A
}
class Test {
enum EnumD { B }
void my_method()
{
enum EnumC { D }
}
}
	The code compiles without any error	The code compiles if we remove line number 1	The code compiles if we remove line number 5	None of the above is correct		3

When access modifier is omitted from the definition of the member of a class. The member has access.	default	public	private	protected		1
Memory deallocation in java is done by:	Programmer	Operating system	Garbage collector	None of the above		3
Which of the following are Object class methods?	toString()	equals()	hashCode()	All of the above		4
This always will points to:	current class object	super class object	sub class object	All of the above		1
State True or False: We can use this in static methods.	True	False				2

Select the correct statements from the below:	enum type can be passed as an argument to switch statement	Every enum constant is always implicitly public static final	Enum declaration can be done outside a Class or inside a Class but not inside a Method.	enum can't contain constructor		1,2,3
State True or False: You can’t use a Java keyword as an identifier.	True	False				1

What is the output of the given code?
 	
public class Test {
public static void main(String args[]) {
final int i;
i = 20;
System.out.println(i);
}
}
	20	compile time error	0	runtime error		1

Integer var1 = new Integer(2); Integer var2 = new Integer(2); What happens when you do if (var1==var2)?	True	False	Compilation error	Runtime Exception		2

What will be the output of the following code snippet?
 	
String str1 = new String("hello");
String str2 = "hello";
if (str1==str2)
System.out.println("Equals");
else
System.out.println("Not Equal");
	Equal	Not Equal	Compilation error	None of the above		2

The Scanner class is found in _____ package.	java.lang	java.util	java.io	None of the above		2
Which of the given LocalDate class method is used to obtain current date?	now()	parse()	format()	newInstance()		1
Which of the given syntax is correct to display date-time with the time zone in the ISO-8601 calendar system, such as 2007-12-03T10:15:30+01:00 Europe/Paris? (Assume the zone is ***Asia/kolkata.)***	ZonedDateTime objt = ZonedDateTime.of(LocalDateTime.now(), ZoneId.of("Asia/Kolkata"));	LocalDate date = LocalDate.now(ZoneId.of("Asia/Kolkata"));	ZonedDateTime time = new ZonedDateTime("Asia/Kolkata");	ZonedDateTime time = new ZonedDateTime.of("Asia/Kolkata");		1
What is the Regular Expression Matching Zero or More Specific Characters?	\	$	*	^		3

public class RegexMatches
{
private static String regex = "dog";
private static String input = "The dog says meow.
" +"All dogs say meow.";
private static String replace = "cat";
 	
public static void main(String[] args) {
//line no.1
 	
Matcher m = p.matcher(input);
INPUT = m.replaceAll(replace);
System.out.println(input);
}
}
 	
Which code need to be inserted at line no.1 to execute it successfully?
	Pattern p = Pattern.compile(regex);	Pattern p = Pattern.test(regex);	Pattern p = Pattern.matches(regex);	Pattern p = new Pattern(regex);		1

public class Demos {
 	
public static void main(String[] args) {
String input = "Hello Welcome";
String pattern = "\\sHello\\sWelcome\\s";
boolean flag = Pattern.matches(pattern, input);
System.out.println(flag);
}
}
 	
What will be the result?
	True	False	Compilation error	Exception		2

Which one of the following is a valid statement?	'char[] c = new char();	'char[] c = new char[5];	char[] c = new char(4);	char[] c = new char[];		2

Analyze the following code and choose the correct answer:
 	
int[] arr = new int[5];
arr = new int[6];
	The code has compile errors because the	The code has runtime errors because the	The code can compile and run fine. The second	The code has compile errors because we cannot assign a		3

public class test {
 	
static void methodA(short s) {
System.out.println("methodA(short) called");
}
static void methodA(int i) {
System.out.println("methodA(int) called");
}
static void methodB(float f) {
System.out.println("methodB(float) called");
}
static void methodB(double d) {
System.out.println("methodB(double) called");
}
 	
public static void main(String args[]) {
methodA(5);
methodB(5.2);
}
}
	methodA(short) called methodB(float) called	methodA(int) called methodB(double) called	methodA(int) called methodB(float) called	Compilation fails		2

What will happen when you attempt to compile and run the following class?
 	
class Base {
Base(int var) {
System.out.println("Base");
}
}
 	
class Derived extends Base {
public static void main(String argv[]) {
Derived obj = new Derived();
}
}
	Compiles without any problem	Compiles and produces output "Base"	Generates Compile time error	None of the above		3

class Demo
{}
public class SubDemo extends Demo
{
public static void main(String []args){
Demo obj = new SubDemo();
System.out.println(obj instanceof Demo);
}
}
 	
What will be the output of above code?
	FALSE	TRUE	Demo	SubDemo		2

class Shape
{
final public double calArea()
{}
}
public class Circle extends Shape
{
int radius;
public Circle(int radius){
this.radius = radius;
}
public double calArea()
{
return 3.142*radius*radius;
}
public static void main(String []args){
Shape obj = new Circle(5);
System.out.println(obj.calArea());
}
}
	It will display area of circle with radius 5	no output	compilation error	runtime Exception		3

class Employee
{
String name;
int id;
public Employee(String name,int id)
{
this.name = name;
this.id=id;
}
}
public class Manager extends Employee
{
public static void main(String []args)
{
Manager mgr = new Manager();
}
}
 	
What will happen after execution of above code?
	Manager class object will be successfully created	Compile time error as there is no default constructor in class Employee	Manager object will be successfully created after implementing default constructor in Manager class	Default constructor need to be implemented in both classes for successful creation of Manager class object		2

class Shape
{
Shape()
{}
}
public class Triangle extends Shape
{
int base,height;
public Triangle(int base,int height){
this.base = base;
this.height=height;
}
public double calArea()
{
return o.5*base*height;
}
public static void main(String []args){
Shape ref = new Triangle(3,4);
System.out.println(ref.calArea());
}
 	
Which change need to be done in the Shape class
	Declare Shape as abstract class	Remove default construct or from Shape class	declare abstract double calArea() in Shape class	declare abstract double calArea() in Shape class and declare Shape class as abstract class		4

What is the output of the given code?
 	
class A {
final void m1()
{
System.out.println("m1 called");
}
}
class B extends A {
 	
public void m2()
{
m1();
System.out.println("m2 called");
}
}
public class Main {
 	
public static void main(String[] args) {
 	
B b = new B();
b.m1();
b.m2();
}
}
	m1 called
	m2 called
		m1 called
		m1 called
		m2 called
			Compilation fails	Runtime error		2

The concept of multiple inheritance is implemented in Java by: I. Extending two or more classes. II. Extending one class and implementing one or more interfaces. III. Implementing two or more interfaces.	Only (II)	(I) and (II)	(II) and (III)	Only (I)	Only (III)	3

What will be the result of compiling and running the given code?
 	
class SuperClass {
int b = 10;
 	
private SuperClass()
{
this.b = 7;
}
 	
int f()
{
return b;
}
}
 	
class SubClass extends SuperClass {
int b;
}
 	
public class MainTest {
public static void main(String[] args) {
SuperClass a = new SubClass();
System.out.println(a.f());
}
	Compilation Fails	Prints 0	Prints 10	Prints 7		1

What is the output of the given code?
 	
class SuperClass {
int b = 10;
}
 	
class SubClass extends SuperClass {
int b = 20;
}
 	
public class MainTest {
public static void main(String[] args) {
SuperClass object = new SubClass();
System.out.println(object.b);
}
}
	10	20	Compilation fails	0		1

Consider the following code:
 	
interface Greek { }
 	
class Alpha implements Greek { }
 	
class Beta extends Alpha {}
 	
class Delta extends Beta
{
public static void main(String[] args)
{
Beta obj = new Beta(); // insert code here
}
}
 	
Which of the following code snippet when inserted individual at the commented line
(// insert code here), will cause a
	Greek objGrk = (Beta)(Alpha)obj;	Alpha objAlpha = obj;	Greek objGrk = (Alpha)obj;	Beta objBeta = (Beta)(Alpha)obj;	Greek objGrk = (Delta)obj;	5

Consider the following code:
 	
interface MyInterface {
 	
// Method declaration code
 	
}
 	
Which of the following code snippet shows the wrong way to declare Method in interface?
	public abstract boolean isValid();	public boolean isValid();	protected boolean isValid();	boolean isValid();		3

Which of the given syntax is correct for interface implementation in Java 8?	interface Demo{
	}
		interface Demo{
		default int getNumber()
		{return 0;}
		}
			interface Demo{
			static int getNumber()
			{return 0;}
			}
				interface Demo
				{
				int getNumber()
				{return 0;}
				}
						1,2,3

Which of the following statements are true related to interface?	Interface doesn't alllow to create object .	Multiple inheritence can be possible in interface.	Class can implement only one interface.	Interface can't allow to declare a member variable.		1,2
On which of the given options abstract modifier can be used?	constructor	static method	non-static methods	class		3,4
If a class inheriting an abstract class does not define all of its function then it will be known as .	Abstract	A simple class	Static class	interface		1

Which of these is not a correct statement?	Every class containing abstract method must be declared abstract	Abstract class defines only the structure of the class not its implementation	Abstract class can be initiated by new operator	Abstract class can be inherited		3
Which of the following is FALSE about abstract classes in Java?	we can use both static and non static variables in abstract classes	Abstract classes can have constructors	A class can be made abstract without any abstract method	A class can inherit from multiple abstract classes		4

Predict the output of the following program:
 	
abstract class demo {
 	
public int a;
 	
demo()
{
a = 10;
}
 	
abstract public void set();
 	
abstract final public void get();
 	
}
 	
class Test extends demo {
 	
public void set(int a) {
this.a = a;
}
 	
final public void get()
{
System.out.println("a = " + a);
}
 	
public static void main(String[] args) {
Test obj = new Test();
obj.set(20);
	a = 10	a = 20	compilation error	runtime error		3

What will be the output?
 	
interface A{
public void method();
}
class One{
public void method()
{
System.out.println("Class One method");
}
}
class Two extends One implements A{
public void method()
{
System.out.println("Class Two method");
}
}
public class Test extends Two{
public static void main(String[] args){
A a = new Two();
a.method();
}
	will print Class One method	will print Class Two method	compiles fine but print nothing	Compilation Error		2

_____ is raised if I do not provide the String array as the argument to the main method.	NullPointerException	IllegalAccessException	NoSuchMethodError	None of the above		3
try { int number = Integer.parseInt("two"); } Which could be used to create an appropriate catch block?	ClassCastException	IllegalStateException	NumberFormatException	None of the above is true		3
Which of the given statement will ensure that each resource is closed at the end of statement?	try with resource	call to close() function on the resource	try with resource and finally	try with catch		1

What is the output of the following program?
 	
class Test {
public static void main(String[] args) {
try {
doMath(5);
System.out.print("hi");
}
finally { System.out.println(" from finally"); }
}
public static void doMath(int den) {
int num = 7 / den;
}
}
	hi from finally	hi
		from
		finally
			prints hi from finally 2 times	None of the above		1

Predict the output of following Java program:
 	
class Main {
  public static void main(String args[]) {
  try {
  throw 10;
  }
  catch(int e) {
  System.out.println("Got the Exception " + e);
  }
 }
}
	Got the Exception 10	Got the Exception 0	Compilation fails	None of the above		3

What will be the output of below code?
 	
class Test
{
 public static void main(String[] args)
 {
  try
  {
  int a[]= {1, 2, 3, 4};
  for (int i = 1; i <= 4; i++)
  {
  System.out.println ("a[" + i + "]=" + a[i] + "n");
  }
  }
 	
  catch (Exception e)
  {
  System.out.println ("error = " + e);
  }
 	
  catch (ArrayIndexOutOfBoundsException e)
  {
  System.out.println
("ArrayIndexOutOfBoundsException");
  }
 }
}
	Compiler error	Run time error	ArrayIndexOutOfBoundsException	a[1]=2n
				a[2]=3n
				a[3]=4n
						1

Predict the output of following Java program:
 	
class Test extends Exception { }
 	
class Main {
  public static void main(String args[]) {
  try {
  throw new Test();
  }
  catch(Test t) {
  System.out.println("Got the Test Exception");
  }
  finally {
  System.out.println("Inside finally block ");
  }
 }
}
}
	Got the Test Exception Inside finally block	Got the Test Exception	Inside finally block	Compilation fails		1

What will be the output of following program?
 	
public class Foo
{
  public static void main(String[] args)
  {
  try
  {
  return;
  }
  finally
  {
  System.out.println("Finally");
  }
  }
}
	Finally	Compilation fails	No output	Runtime Exception		1

Which of the following statements are true related to exception handling in java?	UserDefined exception can be created by extending from RunTimeException class	Throwable is the base class of Error and Exception class	Checked Exception need to be handled either by try and catch block or by using throws keyword in a code before compilation	All of the above		4
When the JVM runs out of memory, which exception will be thrown?	MemoryBoundException	OutOfMemoryError	OutOfMemoryException	OutOfRangeException		2

A _____ is used to walk through a collection and can remove elements from the collection during the iteration.	Enumeration	Iterator	ArrayList	Vector		2
A programmer has an algorithm that requires a java.util.List that provides an efficient implementation of add(0,object), but does NOT need to support quick random access. What supports these requirements?	ArrayList	Queue	Linear List	LinkedList		4

import java.util.*;
class Test {
public static void main(String[] args) {
// insert code here
obj.add("one");
obj.add("two");
obj.add("TWO");
System.out.println(x.poll());
}
}
 	
Which, inserted at // insert code here, will compile?
	List<String> obj = new LinkedList<String>();	TreeSet<String> obj = new TreeSet<String>();	HashSet<String> obj = new HashSet<String>();	Queue<String> obj= new PriorityQueue<String>();		4

What is the output of the following?
 	
import java.util.*;
public class Test {
public static void main(String[] args) {
Set set = new TreeSet();
set.add("anu");
set.add("anil");
set.add("sunil");
for(Object str:set){
System.out.print(str + " ");}
}
}
	anil anu sunil	anu anil sunil	Compilation error	Exception		1

Which among the following Sets maintains insertion order?	HashSet	TreeSet	LinkedHashSet	Both B & C		3

class GenericsDemo<T>
{
T data;
public GenericsDemo(T data)
{
this.data = data;
}
}
 	
Which of the given statement is true about above code?
	GenericsDemo object can be created by passing any type of parameter	GenericsDemo object can be created as given below GenericsDemo<String>obj = new GenericsDemo<String>();	GenericsDemo object can be created as given below GenericsDemo<String>obj = new GenericsDemo<String>("xyz");	Can not create object of given class		3

What will happen after the execution of below code?
 	
class CreateObject
{
public static void main(String []args)
{
Set set = new TreeSet();
set.add("Priya");
set.add("Ritu");
set.add(100);
}
}
	All elements will be successfully added to set	last element 100 will not be added to set	ClassCast Exception	IlleagalState Exception		3

What will happen after the execution of below code?
 	
public class HashSetTest {
 	
public static void main(String[] args) {
 	
Set haseSet = new HashSet<>();
haseSet.add("1");
haseSet.add(1);
haseSet.add("null");
haseSet.add(null);
System.out.println(haseSet);
}
}
	[null, 1, 1, null]	[1,null]	[null,1,null]	[1,1,null]		1

Which collection class allows you to grow or shrink its size and provides indexed access to its elements, but whose methods are not synchronized?	java.util.HashSet	java.util.Vector	java.util.ArrayList	java.util.List		3

What will happen after the execution of below code?
 	
public class HashSetTest {
public static void main(String[] args) {
Map map = new HashMap();
map.put(new String("a"), "Audi");
map.put(new String("a"), "Ferrari");
System.out.println(map);
}
}
	{a=Ferrari}	{a=Audi}	{a=Audi,a=Ferrari}	{a=Ferrari,a=Audi}	compilation fails	1

class DemoCmp //line 1
{
int number;
public DemoCmp(int num)
{
number=num;
}
//line 2
}
public class CreateDemo
{
public static void main(String []args){
TreeSet<DemoCmp>set = new
TreeSet<DemoCmp>();
set.add(new Demo(8));
set.add(new Demo(2));
set.add(new Demo(3));
}}
 	
Which code need to be inserted at line 1 and line 2 for successful execution of above code?
	//line 1
	class DemoCmp implements Comparable
	
	//line 2
	public int compareTo(Object obj)
	{
	return number-obj.number;
	}
		//line 1
		class DemoCmp implements Comparator
		
		//line 2
		public int compare(Object obj1,Object obj2)
		{
		return obj1.number-obj2.number;
		}
			//line 1
			class DemoCmp implements Comparable
			
			//line 2
			public int compare(Object obj1,Object obj2)
			{
			return obj1.number-obj2.number;
			}
				//line 1
				class DemoCmp implements Comparator
				
				//line 2
				public int compareTo(Object obj)
				{
				return number-obj1.number;
				}
						1

Which of the given method must be overridden by a class, after implementing Comapartor?	int compare(Object obj);	int compareTo(Object obj);	int compareTo(Object obj1,Object obj2);	int compare(Object obj1,Object obj2);		4
Which of the following I/O classes helps you in persistence storage of Objects?	ObjectOutputStream	DataOutputStream	FileWriter	ObjectWriter		1
Which of the following methods of the File class will delete a directory or file?	The file class does not allow you to delete a file or directory	remove()	delete()	deleteFile()		3
Which of the following correctly illustrate how an InputStreamReader can be created?	new InputStreamReader(new FileInputStream("data"));	new InputStreamReader(new FileReader("data"));	new InputStreamReader("data");	new InputStreamReader(System.in);		1,4

class Demo{
public static void main(String []args){
Path javaHome = Paths.get("C:/Program Files/Java/jdk1.8.0_25");
System.out.println(javaHome.getNameCount());}
}
 	
What will be the output of above code?
	1	2	3	4		3

Which of these exception is thrown in cases when the file specified for writing is not found?	IOException	FileException	FileNotFoundException	FileInputException		3
Which of these values is returned by read() method is end of file (EOF) is encountered?	0	1	-1	Null		3

What will be the output of the following Java program?
 	
public class FileDemo {
 	
public static void main(String[] args) {
 	
String obj = "abcdef";
int length = obj.length();
char c[] = new char[length];
obj.getChars(0, length, c, 0);
 	
CharArrayReader input1 = new CharArrayReader(c);
CharArrayReader input2 = new CharArrayReader(c, 0, 3);
int i;
try {
while ((i = input2.read()) != -1) {
System.out.print((char) i);
}
} catch (IOException e) {
e.printStackTrace();
}
}
	abc	abcd	abcde	Compilation fails		1

What will be the output of the following Java program?
 	
public class FileDemo {
 	
public static void main(String[] args) throws IOException {
 	
InputStream obj = new FileInputStream("src/inputoutput.txt");
System.out.print(obj.available());
 	
}
}
 	
Note: inputoutput.txt is stored in the disk.
	True	False	prints number of bytes in file	prints number of characters in the file		3

What will be the output of the following Java code?
 	
class ThreadDemo{
 	
public static void main(String args[])
{
Thread t = Thread.currentThread();
System.out.println(t);
}
}
	Thread[5, main]	Thread[main,5]	Thread[main,0]	Thread[main,5,main]		4

What will be the output of the following Java code?
 	
public class ThreadDemo {
public static void main(String args[]) {
Thread t = Thread.currentThread();
System.out.println(t.getPriority());
}
}
	1	10	5	undefined		3

What state does Thread enter in when it has been created and started?	new	runnable	running	waiting		3
How can you ensure all threads that started from main must end in order in which they started and also main should end in last?	join() method	sleep() method	wait() method	run() method		1

What will be the order of output of the program?
 	
class Test extends Thread {
public void run()
{
System.out.println("Run");
}
}
class Myclass {
public static void main(String[] args) {
Test t = new Test();
t.start();
System.out.println("Main");
}
}
	Main Run	Run Main	Depend upon Program	Depend upon JVM		4

What will be the output of the given program?
 	
class Test implements Runnable {
public void run()
{
System.out.println("Run");
}
}
class Myclass {
public static void main(String[] args) {
Test t = new Test();
t.start();
System.out.println("Main");
}
}
	Main Run	Run Main	Compile time error	Depend upon JVM		3

What will be the output of the given program?
 	
class MyThread extends Thread {
@Override
public void run()
{
System.out.println("child thread");
}
}
 	
public class ThreadDemo {
public static void main(String args[]) {
 	
MyThread thread = new MyThread();
thread.start();
thread.start();
}
}
	child thread followed by RuntineException	child thread child thread	child thread	Compilation fails		1

Select the correct 3 constants of Thread class:	public static int MINIMUM_PRIORITY	public static long MINIMU	public static int MIN_PRIORITY	public static long MIN_PRIORITY		3

What will be the output of the given program?
 	
class Test implements Runnable {
public void run()
{
System.out.println("Run");
}
}
class Myclass {
public static void main(String[] args) {
Thread t1 = new Thread();
t1.start();
System.out.println("Main");
}
}
	Run	Main	Compile time error	Run Main		2

Select the correct statement to create a thread pool 6of 5 fixed thread:	ExecutorService executor = Executors.newThreadPool(5);	ExecutorService executor = Executors.newFixedThreadPool(5);	ExecutorService executor = new ExecutorService(5);	ExecutorService executor = new ThreadPoll(5);		2
What are the two types of Streams offered by java 8?	sequential and parallel	Sequential and random	parallel and random	random and synchronized		1
What is Optional object used for?	Optional is used for optional runtime argument	Optional is used for optional spring profile	Optional is used to represent null with absent value	Optional means it's not mandatory for method to return object		3
In Java 8 Interfaces, methods can be:	default	abstract	static	All of the above		4
Select the correct code snippet to display the output as 30:	class Calculator { public static int add(int a, int b) { return a + b; } } public class MethodRefDemo { public static void main(String[] args) { BiFunction<Integer, Integer, Integer> adder = Calculator::add; int result = adder.appl	class Calculator { public static int add(int a, int b) { return a + b; } } public class MethodRefDemo { public static void main(String[] args) { BiFunction<Integer, Integer, Integer> adder = Calculator.add(); int result	class Calculator { public static int add(int a, int b) { return a + b; } } public class MethodRefDemo { public static void main(String[] args) { BiFunction<Integer, Integer, Integer> adder = Calculator::add(); int result = adder.app	class Calculator { public static int add(int a, int b) { return a + b; } } public class MethodRefDemo { public static void main(String[] args) { BiFunction<Integer, Integer, Integer> adder = Calculator:add(); int result = adder.apply(10, 20); System.out.println(resul		1

Select the output of the given code:
 	
public class ForEachExample {
public static void main(String[] args) {
 	
List<String> gamesList = new ArrayList<String>();
gamesList.add("Football");
gamesList.add("Cricket");
gamesList.add("Chess");
gamesList.add("Hocky");
gamesList.add("Cricket");
gamesList.add("Chess");
gamesList.add("Football");
gamesList.add("null");
gamesList.add(null);
 	
Set<String> set =
gamesList.stream().collect(Collectors.toSet());
set.forEach(game -> System.out.println(game));
}
}
	Chess
	Hocky
	null
	Cricket
	null
	Football
		Chess
		Hocky
		null
		Cricket
		Football
			Football
			Cricket
			Chess
			Hocky
			Cricket
			Chess
			Football
			null
				Football
				Cricket
				Chess
				Hocky
				Cricket
				Chess
				Football
				null
				null
						1

What is the output of the given code?
 	
class Employee {
 	
int id;
String name;
public Employee() {
}
public Employee(int id, String name) {
this.id = id;
this.name = name;
}
//setters and getters
}
 	
public class ComparableTest {
public static void main(String[] args) {
 	
List<Employee> list = new ArrayList<>();
list.add(new Employee(123, "Sugan"));
list.add(new Employee(101, "Kavita"));
list.add(new Employee(132, "Anju"));
 	
Comparator<Employee> comparator = (a, b) -> {
return a.getName().compareTo(b.getName());
};
list.sort(comparator);
 	
list.forEach(e ->
System.out.println(e.getName()));
	Sugan
	Kavita
	Anju
		Anju
		Kavita
		Sugan
			Kavita
			Sugan
			Anju
				Compilation fails		2

What is the output of the givne code?
 	
public class ForEachExample {
 	
public static void main(String[] args) {
 	
List<String> trainers = new ArrayList<String>();
trainers.add("Soham");
trainers.add("Tina");
trainers.add("Anju");
trainers.add("Kavya");
trainers.add(null);
trainers.add("null");
trainers.add("Soham");
trainers.add("Tina");
 	
Set<String> list = trainers.stream().filter(s -> s.startsWith("S")).collect(Collectors.toSet());
System.out.println(list);
}
}
	[Tina,Soham,null]	[Soham,Tina,Soham,Tina]	[Soham, Tina]	Compilation fails	NullPoinerExcepti on	5

Select the correct code to display the message "Good Luck":	MyLambda {
	public String test();
	}
	
	public class LambdasTest {
	public static void main(String[] args) {
	
	MyLambda my = () -> {
	return "Good Luck";
	};
	
	System.out.println(my.test());
	}
	}
		interface MyLambda {
		public String test();
		}
		
		public class LambdasTest {
		public static void main(String[] args) {
		
		MyLambda my = (String s) -> {
		return "Good Luck";
		};
		
		System.out.println(my.test())
			MyLambda {
			public String test();
			}
			
			public class LambdasTest {
			public static void main(String[] args) {
			
			MyLambda my = () -> {
			return "Good Luck";
			}
			
			System.out.println(my.test());
			}
			}
				interface MyLambda {
				public String test();
				}
				
				public class LambdasTest {
				public static void main(String[] args) {
				MyLambda my = () -> {
				return "Good Luck";
				};
				my.test();
				}
				}
						1

In java 8, void accept(T t) is a method of:	Predicate	Supplier	Function	Consumer		4
In java 8, R apply(T t) is a method of:	Function	Process	Predicate	Consumer		1

JUnit is used for _____ in Java Application.	Integration Testing	Code Review	Unit Testing	Black Box Testing		3
Which of the given annotation is used to do initialization of some task before each test run?	@Test	@After	@BeforeClass	@Before		4

Which of the given option is correct for adding exception to the test method given below?
 	
Public void divideByZeroTest()
{
calobj.divide(15,0);
}
 	
Note : calobj is an object of Calculate class which contains divide(int x,int y) method.
	@Test(expected = ArithmeticException.class)	@Test(ArithmeticExceptionn.class)	@Before(ArithmeticExceptionn.class)	@Before(expected = ArithmeticException.class)		1

Consider the following code snippet:
 	
import static org.junit.Assert.assertEquals;
import org.junit.Test;
 	
class Calculator {
public long add(long number1, long number2) {
return number1 + number2;
}
}
 	
public class CalculatorTest {
@Test
public void add() throws Exception {
Calculator cal = new Calculator();
assertEqual(4, cal.add(1, 3));
}
}
 	
The above code returns compile time error.
 	
What change needs to be done in order to run the test successfully?
	There is no error in the code above	The TestCase class has not been extended hence it not recognized as a Test class	Change the assertEquals method to assertTrue(4,cal.add(1,3))	The assertEqual method should be written as assertEquals(4,cal.add(1,3))		4

Which of the given statement is true about org.junit.runners.Suite class?	It runs group of test cases	It specifies runner class to run the annotated class	Both of the above	None of the above		1

Which of the given is the correct implementation of Test suites?	import org.junit.runner.RunWith; import org.junit.runners.Suite; @RunWith @Suite.SuiteClasses({ TestCalAdd.class, TestCalSubtract.class, TestCalMultiply.class, TestCalDivide.class }) public class CalSuite { }	org.junit.runner.RunWith; import org.junit.runners.Suite; @RunWith(Suite.class) @Suite.SuiteClasses({ TestCalAdd.class, TestCalSubtract.class, TestCalMultiply.class, TestCalDivide.class }) public class CalSuite { }	import org.junit.runner.RunWith; import org.junit.runners.Suite; @Run(Suite.class) @Suite.SuiteClasses({ TestCalAdd.class, TestCalSubtract.class, TestCalMultiply.class, TestCalDivide.class }) public class CalSuite { // the class remains completel	org.junit.runner.RunWith; import org.junit.runners.Suite; @Suite.SuiteClasses({ TestCalAdd.class, TestCalSubtract.class, TestCalMultiply.class, TestCalDivide.class }) public class CalSuite { // the class remains completely empty, // being used only as a holder for the above annotations }		2
Choose the correct statements about parameterized test:	It allows to run the same test with different parameters	It can be specified as : @RunWith(Parameterized.class)	To use it, we need to add static method which returns collection of data.	It can be done with @RunWith(Suite.class)		1,2,3
Which of the given options are mock frameworks?	DynaMock	Jmock	EasyMock	StaticMock		1,2,3
Which of the given method is used to create EasyMock?	EasyMock.createMock();	EasyMock.createMock(Classname.class);	EasyMock mock = new EasyMock();	EasyMock mock = new EasyMock(Classname.class);		2

What is the output of the given code?
 	
public class MyClassTest {
 	
@Before
public void setUp()
{
System.out.println("in setUp method");
}
 	
@After
public void destroy()
{
System.out.println("in destroy method");
}
 	
@Test
public void test1()
{
assertTrue(true);
System.out.println("test case 1 executed");
}
	in setUp method
	test case 1 executed
	in destroy method
		test case 1 executed	in setUp method
			in destroy method
				Compilation fails		1

Which of the following keyword is used with Data Control Language (DCL) statements?	SELECT	INSERT	DELETE	GRANT		4
Which operator performs pattern matching?	BETWEEN operator	LIKE operator	EXISTS operator	NOT EXISTS		2
Which of the SQL statements is correct?	SELECT Username AND Password	SELECT Username, Password	SELECT Username, Password	SELECT Username, Password FROM Users		2
Select thr correct set of TCL command from the below:	GRANT REVOKE SAVEPOINT	COMMIT ROLLBACK SAVEPOINT	GRANT ROLLBACK ALTER COMMIT	ALTER TRUNCATE ACCESS CHECK		2
In SQL, which command is used to SELECT only one copy of each set of duplicable rows?	SELECT DISTINCT	SELECT UNIQUE	SELECT DIFFERENT	SELECT SAME		1
Which of the SQL statements is correct?	DELETE FROM employee WHERE first_name = 'Sneha', salary >= 40000;	DELETE FROM employee WHERE first_name = 'Sneha' AND salary >= 40000;	DELETE employee WHERE first_name = 'Sneha' AND salary >= 40000;	DELETE employee WHERE first_name = 'Sneha', salary >= 40000;		2

Consider the below emp table and select the output of the given query: select *from emp; ID NAME SALARY ---------- ---------- -- ---------- 1 Prem 54000 2 Sneha 84000 3 Riya 45000 4 Raja 65000 SELECT name FROM emp WHERE salary>ANY(35000,50000,75000);	Sneha	Prem Sneha Riya Raja	Prem Raja	Sneha Raja		2
Which of the SQL statements are correct?	SELECT name FROM emp WHERE salary between (50000 and 90000);	SELECT name FROM emp WHERE salary between 50000 to 90000;	SELECT name FROM emp WHERE salary between (50000,90000);	SELECT name FROM emp WHERE salary between 50000 and 90000;		4
Select the correct statement to create PreparedStatement interface Object:	PreparedStatement stmt = connection.prepareStatement(String sql);	PreparedStatement stmt = new prepareStatement(String sql);	PreparedStatement stmt = connection.prepareStatement();	PreparedStatement stmt = new prepareStatement();		1
Select the correct statement to create ResultSet interface Object:	PreparedStatement stmt = connection.prepareStatement(String sql); ResultSet rs = new ResultSet();	PreparedStatement stmt = connection.prepareStatement(String sql); ResultSet rs = stmt.executeQuery();	PreparedStatement stmt = new PrepareStatement(String sql); ResultSet rs = stmt.executeQuery();	PreparedStatement stmt = new PrepareStatement(); ResultSet rs = stmt.executeQuery(String sql);		2

What is true about Connection Pooling?	It increase the performance system	It promotes reusability	Connection pool implementation not disconnect the link with the database even though client close the connection	All of the above		4
Which of the following are pure java drivers?	Type 1 - JDBC-ODBC Bridge	Type 2 - Java Native API	Type 3 - Java to Network Protocol	Type 4 - Java to Database Protocol		3,4
The scenario where you want to take data from user each time differently, this data you want to pass as an argument to the SQL query. Which of the following statement you will use to execute such query?	Statement	Prepared Statement	CallableStatement	All of the above		2
Which of the following methods is best suitable for executing a DML statement?	executeQuery()	executeUpdate()	execute()	getResultSet()		2
Which of the following are interfaces in JDBC API?	DriverManager	Prepared Statement	Connection	ResultSet	Statement	2,3,4,5

Which of the following code snippet is most appropriate to roll back the transaction when an error occurs in the transaction?	conn.setAutoCommit(false); // perform transactions
	
	conn.commit();
	
	con.setAutoCommit(true);
	conn.rollback();
	;
		try {
		
		conn.setAutoCommit(false); // perform transactions
		
		conn.commit();
		
		con.setAutoCommit(true);
		} catch (SQLException e) {
		conn.rollback();
		;
		}
			try {
			
			conn.setAutoCommit(true); // perform transactions
			
			conn.commit();
			
			con.setAutoCommit(false);
			} catch (SQLException e) {
			conn.rollback();
			;
			}
				Transaction cannot be rolled back		2

Which of the following methods of ResultSet interface helps in retrieving the type of each column in the resultSet?	getData()	getMetaData()	getType()	getColumnType()		2

Pick up the valid statement to execute the following sql query:
 	
PreparedStatement pstmt = con.prepareStatement("insert into student values(?,?)");
pstmt.setString(1,"Asha");
pstmt.setString(2,"Basha");
	ResutlSet rs = pstmt.executeQuery();
	;
		Pstmt.executeQuery();	ResultSet rs = pstmt.executeUpdate();	None of the above		4

ResultSet rs = stmt.executQuery("SELECT name, rank, serialNo FROM employee"); Which of the following will get the value of 'name' from the above ResultSet rs?	rs.getString(0)	rs.getString(1)	rs.getString("name")	None of the above		2,3
Which driver is called as thin-driver in JDBC?	Type-4 driver	Type-1 driver	Type-3 driver	Type-2 driver		1