```
class ParentClass extends Exception {
}

class ChildClass extends ParentClass {
}

public class ExceptionDemo {
   public static void main(String[] args) {
      try {
         throw new ChildClass();
      } catch (ParentClass p) {
         System.out.println("Caught ParentClass exception");
      } catch (ChildClass c) {
         System.out.println("Caught ChildClass exception");
      }
   }
}
```

Answer :Caught ChildClass exception

2.
```
class Atom {
   Atom() {
      System.out.print("atom");
   }
}

class Rock {
   Rock(String type) {
      System.out.print(type);
   }
}

public class Mountain extends Rock {
   Mountain() {
      super("granite");
      new Rock("granite ");
   }

   public static void main(String[] a) {
      new Mountain();
   }
}
```

Answer:atom granite atom granite

3.10. interface Jumper {
```
    public void jump();
```

```
}

20. class Animal {
}

30. class Dog extends Animal {
      Tail tail;
}

40. class Beagle extends Dog implements Jumper {
      public void jump() {}
}

50. class Cat implements Jumper {
      public void jump() {}
}
```

- Cat is-a Jumper
- Dog is-a Animal
- Beagle has-a Tail

4.

```
public static void main(String[] args) {
   int pointer = 0;
   int value = 1;
   while (true) {
       ++pointer;
       if (pointer % 2 == 0)
           continue;
       else if (pointer % 5 == 0)
            break;
            value *=3;
           System.out.println(value);


       }
   }
```

Output: 3 9

5.import java.util.ArrayList;
import java.util.List;

```
public class Example {
   public static void main(String[] args) {
      List<Integer> list = new ArrayList<>();

      int[] array = {8, 9};

      list.add(array[0]);
      list.add(array[1]);

      System.out.println(list);
   }
}
```

Answer: 8,9

```
6.import java.util.ArrayList;
import java.util.List;

public class Example {
   public static void main(String[] args) {
      List<Integer> list = new ArrayList<>();
      int[] array = {0, 0, 0};

      list.add(array[0]);
      list.add(array[2]);
      list.set(1, array[1]);
      list.remove(0);

      System.out.println(list);
   }
}
```

output:[0]

```
7.public class Breaker {
   static String o = "";

   public static void main(String[] args) {
      z:
      for (int x = 2; x < 10; x++) {
         if (x == 4) continue;

         if (x == 6) break z;

         o = o + x;
      }

      System.out.println(o);
   }
}
```

Answer:235

8.Which type of code is really helpful in selecting two or more items in a list box or text area? Note that all the options of the dropdown are in td tags

a.List<WebElement> options = select.findElements (By.tagName("td")); Action multipleSelect = builder

.keyDown (Keys.CONTROL)

.click(options.get(2))

click(options.get(4))

click (options.get(6))

.build();

multipleSelect.perform();
option b:
List<WebElement> options = select.findElement(By.tagName("td"));

Action multipleSelect = builder .keyDown (Keys.CONTROL)

click (options.get(2))

(D)

```
Select dropdown = new Select (driver.findElements (By.tagName("td")));
dropdown.selectByVisibleText("iteml");

dropdown.selectByVisibleText("item2"); dropdown.selectByVisibleText("item3");
```

```
Answer:List<WebElement> options = select.findElements(By.tagName("td"));
Action multipleSelect = builder
        .keyDown(Keys.CONTROL)
        .click(options.get(2))
        .click(options.get(4))
        .click(options.get(6))
        .build();
multipleSelect.perform();
```

9.Which of the options can be added in linel to remove the error from the following code?

```
System.out.println("Car speeding");
System.out.println("Car speeding");

public class Tester implements Car, Bike {

    @Override
    public void accelerate() {
        Car.super.accelerate(); // Option 1
        Bike.super.accelerate(); // Option 2
    }

    // Other code...
}
```

10.

```
interface Fish {}
class Perch implements Fish {}
class Walleye extends Perch {} // Fixed the typo here
class Bluegill {}

public class Fisherman {
    public static void main(String[] args) {
        Fish f = new Walleye();
        Walleye w = new Walleye();
        Bluegill b = new Bluegill();

        if (f instanceof Perch)
            System.out.print("f-p ");
        if (w instanceof Fish)
            System.out.print("w-f ");
        if (b instanceof Fish)
            System.out.print("b-f ");
    }
}
```

Answer: f-p w-f

11.public class Student {

```
    public String sName; // Assuming grade is also a member of the Student class
    public String grade;

    public static void main(String[] args) {
        Student S = new Student();
        System.out.println("[" + S.sName + ":" + S.grade + "]");
    }
}
```

Answer: [null:null]

12.

```
        int[] array = {6, 9, 8};
                List<Integer> list = new ArrayList<>();
                list.add(array[0]);     // Adding 6 to the list
                list.add(array[2]);     // Adding 8 to the list
                list.set(1, array[1]);
                list.remove(0);//
                System.out.println(list);
```

Output:9

13

13.public static void main(String[] args) {

```
            int num1 = 0;
            int num2 = 0;
            for (int var = 0; var < 5; var++) {
                if ((++num1> 2) && (++num2 > 2)) {
                    num1++;
                }

            }
            System.out.println(num1 + " and " + num2);
        }
```

Output:6 &3

14

```
class Phase2TestNg {
            public static boolean funcA(int a) {
               boolean b = (a==10)?true:false;
               System.out.println(b);
               return b;
            }
            public static void main(String args[]) {
               if(funcA(10) && funcA(5))
                  System.out.println("pass");
            }
         }
```

Output:
True
Flase

15.abstract class Demo{

```
   public char alpha;
   Demo(){
      alpha='D';
   }
}

public class Tester extends Demo{
   final public void setAlpha(char alpha) {
      this.alpha=alpha;
   }

   final public void getAlpha() {
      System.out.println("alpha = "+alpha);
   }

   public static void main(String[] args) {
      Tester obj=new Tester();
      obj.setAlpha('A');
      obj.getAlpha();
   }
}
```

Output:
alpha = A

16. `driver.manage().timeouts().setScriptTimeout(-10, SECONDS);`.

The correct answer is: The script will be allowed to run indefinitely.

```
17. Class TestQuestion {

   String s1 = "overloading main String s[]";
   String s2 = "overloading main int s[]";
   public static void main(String args[]) {
       System.out.println("inside main 1");
   }
   public static void main(int args[]) {
       System.out.println("inside main 2");
   }
}
```

Output: inside main1

```
18.public static void main(String[] args) {
    int[] arr = {10, 0};
    int i = 0;
    try {
        int answer = arr[i] / arr[i + 1];
    } catch (Exception e) {
        System.out.println("Unknown issues.");
    } catch (ArithmeticException ae) {
        System.out.println("Invalid divisor.");
    }
}
```

output:A compilation error occurs

19.

```
class Thingy {Meter m = new Meter(); }
class Component { void go() { System.out.print("c"); }
class Meter extends Component { void go()  {System.out.print("m"); }}

class DeluxeThingy extends Thingy {
    public static void main(String[] args) {
        DeluxeThingy dt = new DeluxeThingy();
        dt.m.go();
        Thingy t = new DeluxeThingy();
        t.m.go();
    }
}
```

Given above code which of the below statements are true?
The output is mm
the output is mc
component is-a Meter
component has-a Meter
DeluxeThingy is-a component
DeluxeThingy has-a component

```
20.class ParentClass extends Exception { };
class ChildClass extends ParentClass { };
public class Phase2TestNg {
 public static void main(String[] args) {
  try {
    throw new ChildClass();
  }
  catch(ParentClass p) {
    System.out.println("Caught parent class exception");
  }
  catch(ChildClass c) {
    System.out.println("Caught child class exception");
  }
 }
}
```

Caught Parent Class Exception
Caught Child Class Exception
Error because the Child Class is not Throwable
Error because the Parent Class Exception is caught before Child Class

```
21.class Vehicle {
    int vno;
    String name;
    public Vehicle (int vno, String name) {
        this.vno = vno;
        this.name = name;
    }
    public String toString () {
        return vno + ":" + name;
    }
}
```

and this code fragment:

```
Set<Vehicle> vehicles = new TreeSet<> ();
vehicles.add(new Vehicle (10123, "Ford"));
vehicles.add(new Vehicle (10124, "BMW"));
System.out.println(vehicles);
```

what will be the output for the following code ?

Output: [10123:Ford, 10124:BMW]

22.

```
public class Employee{
private int empId;
private String name;
private String city;

Employee() {
   this.city = "New York";
}

Employee(String name, int empId) {
   this();
   this.name = name;
   this.empId = empId;
}

public static void main(String[] args) {
   Employee employee1 = new Employee("John", 101);
   Employee employee2 = new Employee();

   System.out.println(employee1.name + " " + employee1.empId + " "
      + employee1.city);

   System.out.println(employee2.name + " " + employee2.empId + " "
      + employee2.city);
}
```

Output:

```
John 101 New York
null 0 New York
```

23.What is the valid syntax to select all the checkboxes in the page using java ?

[a]

```
List<WebElement> all = driver.findElement(By.xpath("//input[@type='checkbox']"));

for (WebElement element : all)
{
   element.click();
}
```

[b]

```
List<WebElement> all = driver.findElements(By.xpath("/input[@type='checkbox']"));

for (WebElement element : all)
{
   element.click();
}
```

[c]

```
List<WebElement> all = driver.findElement(By.xpath(*/input[@type='checkbox']"));

for (WebElement element : all)
{
   element.click();
}
```

[d]

```
List<WebElement> all = driver.findElements(By.xpath("//input[@type='checkbox']"));

for (WebElement element : all)
{
   element.click();
}
```