1. The % operator means mod, which calculates the remainder after division i.e. a % b = c means a = b * k + c that k is an integer.
2. Override: There are two methods with the same name and same parameters. One of the methods is in the parent class and the other is in the child class. And overriding happens while the program is running.
   Overload: Two or more methods in the same class have the same name but different parameters and maybe different kinds of return values.
3. The result will be float.
4. intVal = Integer.parseInt(String.valueOf(objs[0]));
5. The output is 20. Because myObj is an Integer, the toString() method has been override by the subclass Integer, which returns the value by the type of String.
6. The result of (s1==s2) is false. This is to judge whether s1 and s2 are the same object. Exactly they are not.
   The result of s1.compareTo(s2) is -1. The method compareTo() is to calculate which string is greater by value. Return value 0 means the value of the two strings are the same. Return value 1 means the value of s1 is greater than s2. Return value -1 means the value of s1 is less than s2.
7. Fruit is an abstract class. An abstract class cannot have an instance.
8. When defining an interface, we care more about method. We define method but not implement them, which means any class that implements this interface has to implement all the method defined in the interface i.e. we want to make sure this class has some functions of the interface. And the member variables of the interface should be public static final.
   When designing an object as a subclass of another object, we care about the relationship between these classes. The common variables and methods should be defined in the super class, which means the general property. And an abstract super can also define an abstract method without implementing it.
9.
```
Class Main {
      Public static void main(String[] args) throws IOException{
            FileReader file = new FileReader(“data.txt”);
            BufferedReader fileInput = new BufferedReader(file);
            try {
                  String inLine = fileInput.readLine();
                  while (inLine!= null) {
                        String[] numbers = inLine.split(“ “);
                        Integer a = Integer.parseInt(numbers[0]);
                        Integer b = Integer.parseInt(numbers[1]);
                        Integer c = a/b;
                        System.out.println(“result = “ + c);
                        inLine = fileInput.readLine();
                  }
            } catch (ArithmeticException e) {
                  e.printStackTrace();
            } catch (IndexOutOfBoundsException e) {
                  e.printStackTrace();
            }
      }
}
```

IndexOutOfBoundsException, IOException, and ArithmeticException(divide 0)

10. The parameters passed to a lambda function should be final or effectively final of the enclosing scope.
11. TimePrinter is the inner class of TimerMessage. Therefore it can get access to the private variable msg. However, the main method belongs to another class TimerMessageRunner, which cannot access the private variable of the calss TimerMessage.
12.

```
public class TestFrame extends JFrame {

    private static final int FRAME_WIDTH = 200;
    private static final int FRAME_HEIGHT = 75;
    private JButton closeButton = new JButton("Close");

    public TestFrame()
    {
         setSize(FRAME_WIDTH, FRAME_HEIGHT);
         this.setLayout(new FlowLayout);
         closeButton.addActionListener((e) -> System.exit(0));
         this.add(closeButton);
    }

    public static void main(String[] args) {
         TestFrame testFrame = new TestFrame();
         testFrame.setVisible(true);
    }
}
```