[ZANG] - [Chuanjie] - ITCAssignment

Student ID: 699382953

[chuanjiezang@gmail.com](mailto:chuanjiezang@gmail.com)

Q1:

Pseudo-Code

Start

Initialize the variable sum to 0.0

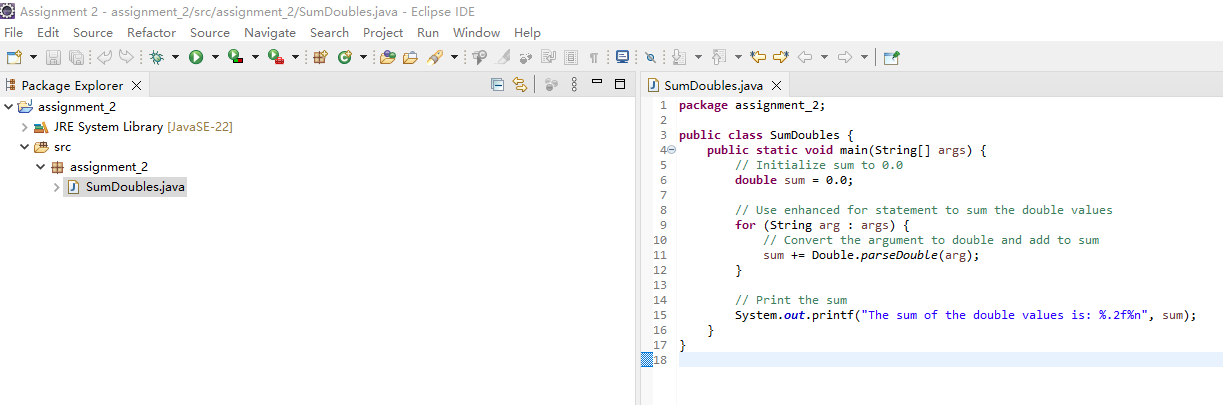
Iterate through the command line arguments using the enhanced for statement.

Use Double.parseDouble for each argument to convert it to a double value

Add the converted values to sum

Output the value of sum

End



Q2:

Pseudo-Code

Start

Create a boolean array of size 1000, initialize to true

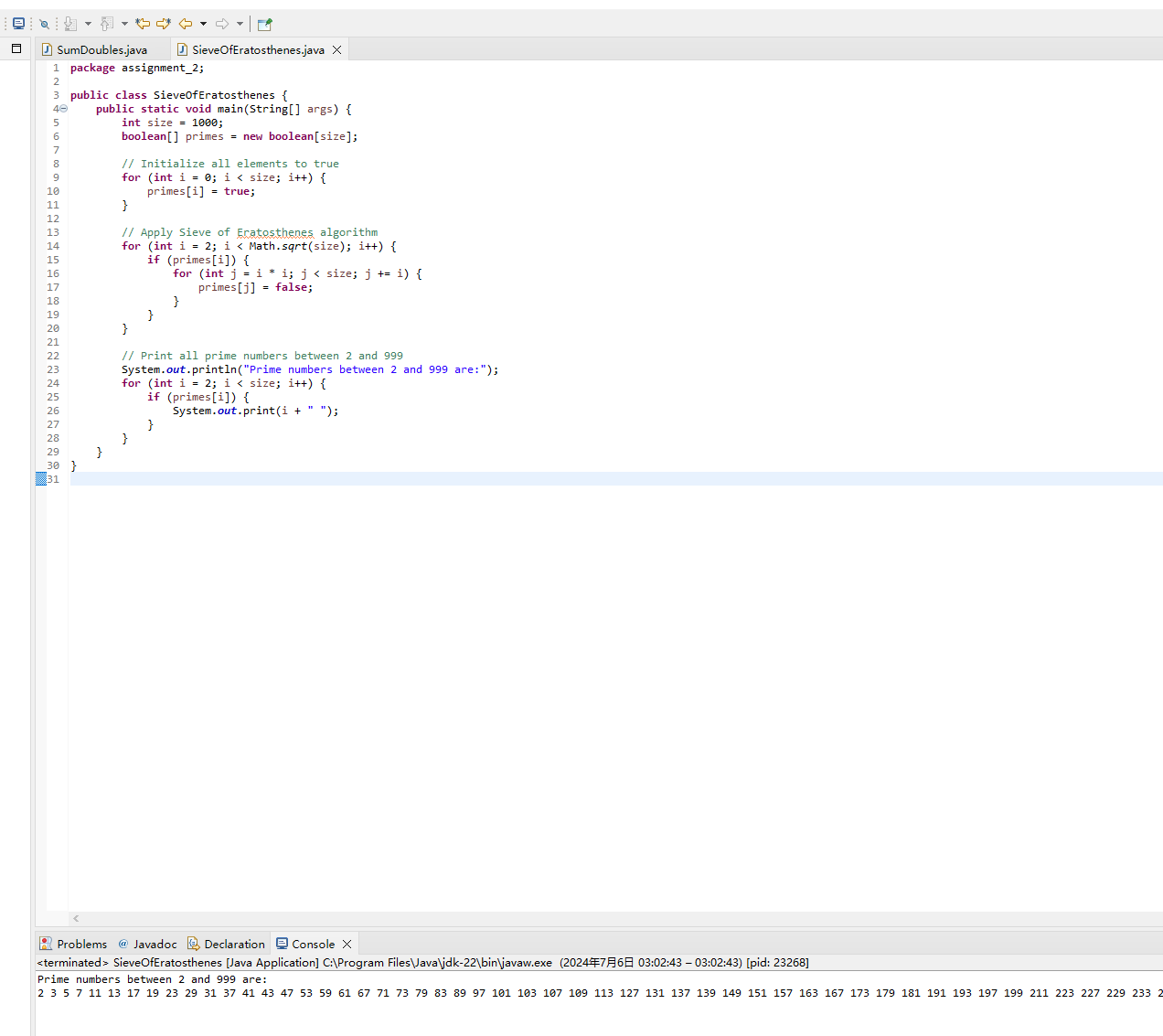
Iterate through the array starting at index 2

If the current element is true

Iterate through all multiples of that element, setting them to false

Print all indices that are true (representing prime numbers)

End



Q3:

Pseudo-Code

Start

Creating the Date Class

Declare instance variables month, day, year

Write constructor for initialization and error checking

Write nextDay method to increment the day

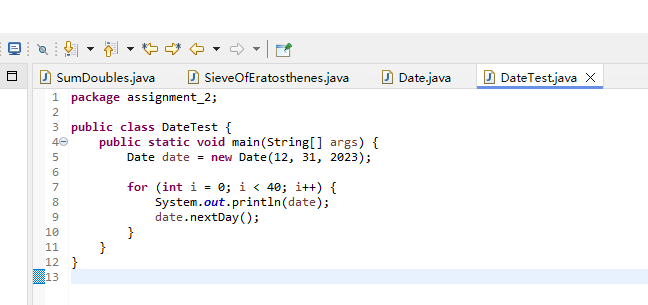
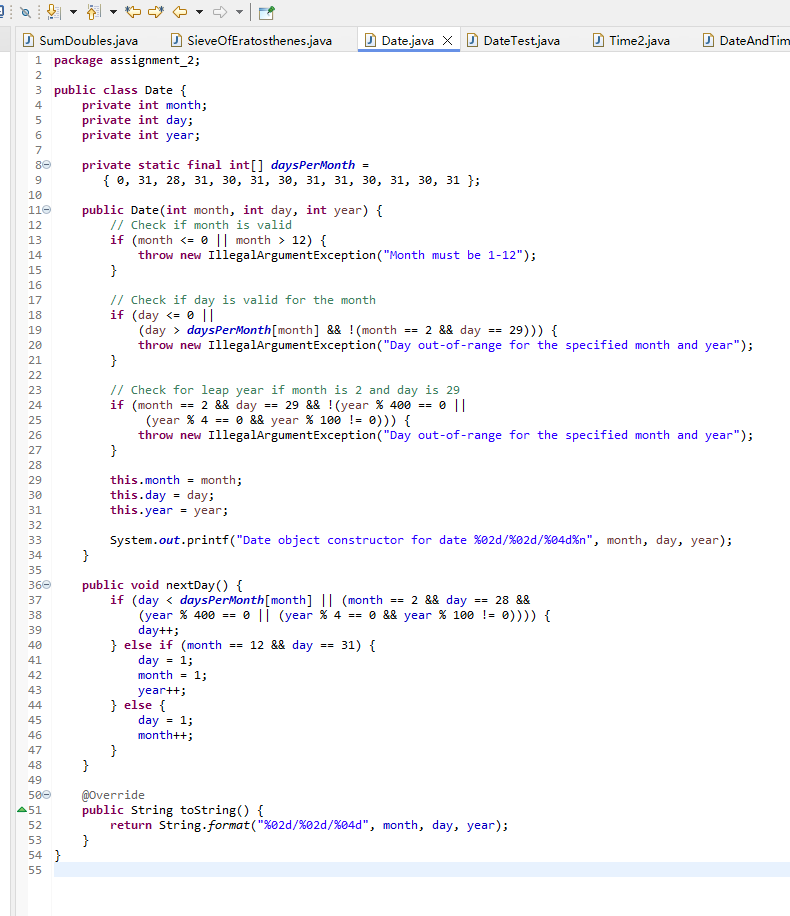
Write displayDate method to display date

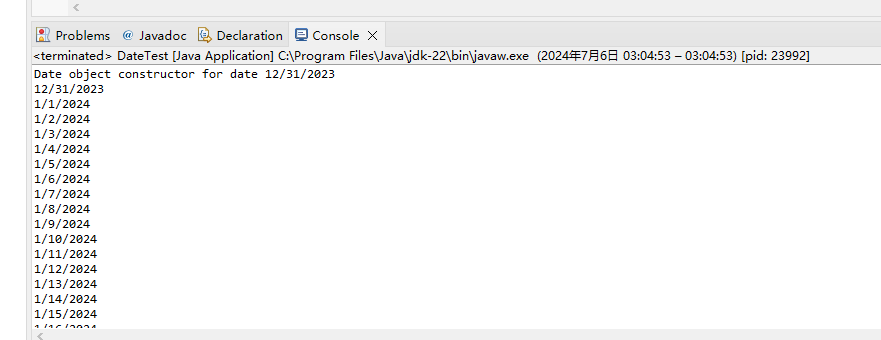
Create DateTest class

Create Date object in main method

Test nextDay method

End





Q4:

Pseudo-Code

Start

Create the Time2 class

Declare instance variables hour, minute, second

Write constructor to initialize

Write tick method to increment one second

Write incrementMinute method to increment one minute.

Write incrementHour method to increment one hour.

Write toString and toUniversalString methods.

Create DateAndTime class

Inherit Date and Time2 class

Override the incrementHour method to call the nextDay method when the time is incremented to the next day.

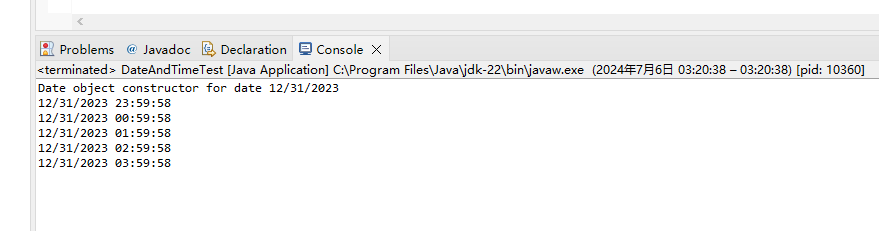
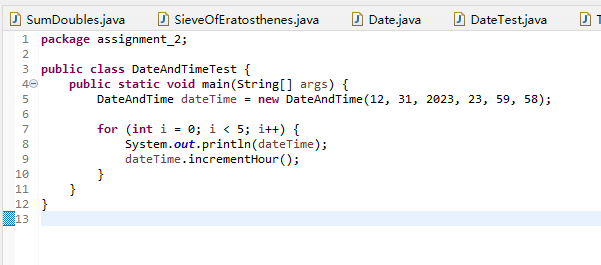
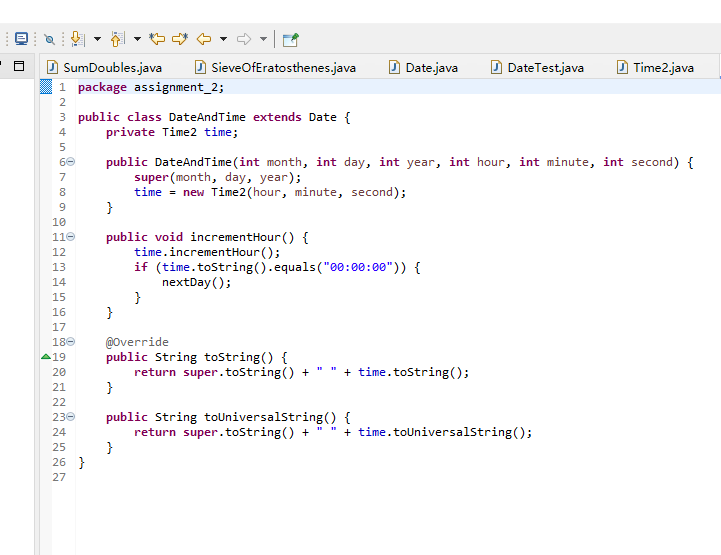
Override toString and toUniversalString methods to output date and time.

Create DateAndTimeTest class

Create DateAndTime object in main method.

Test incrementHour and other methods

End



Q5:

Pseudo-Code

Start

Creating the Date3 class

Declare instance variables month, day, year

Write constructor for initialization and error checking

Write nextDay method to increment the day

Write displayDate method to display date

Create Time3 class

Inherit Date3 class

Declare instance variables hour, minute, second

Write constructor to initialize date and time

Write tick method to increment one second

Write incrementMinute method to increment one minute.

Write incrementHour method to increment one hour.

If the time is incremented to the next day, call the nextDay method.

Write toString method to output date and time

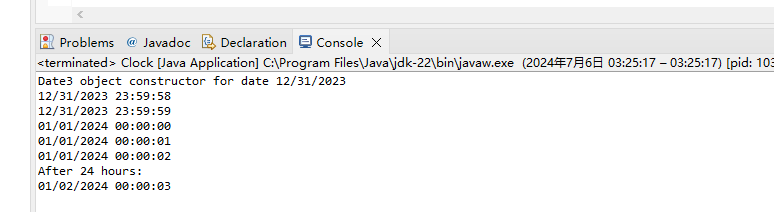
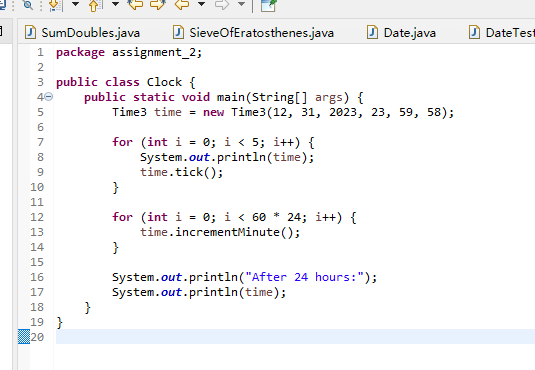
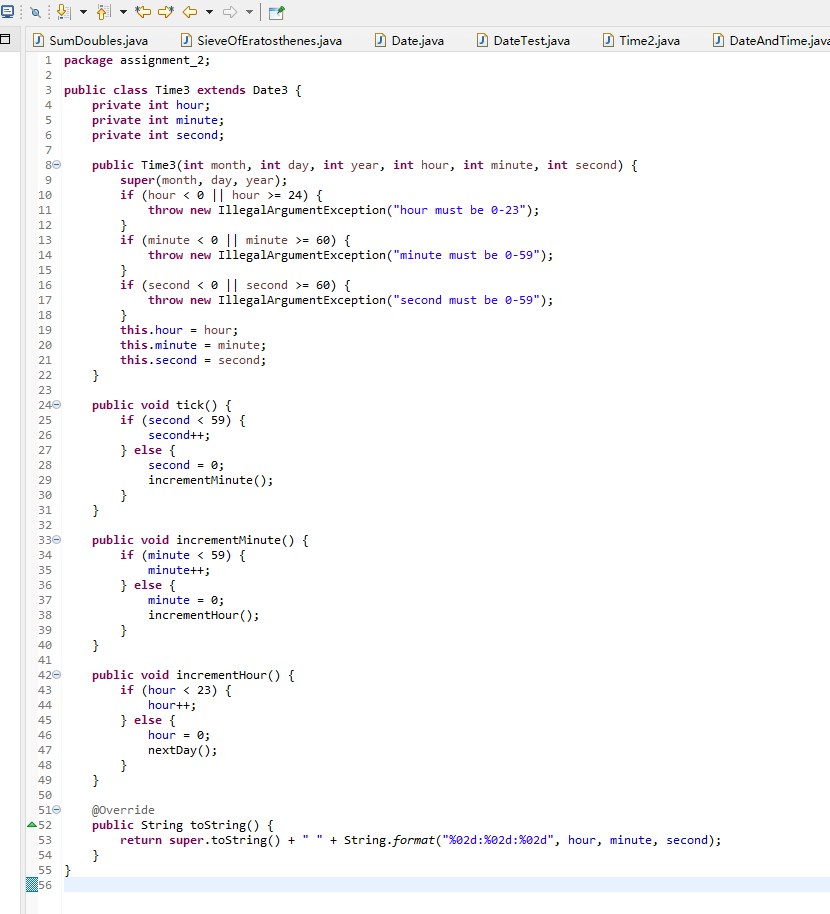
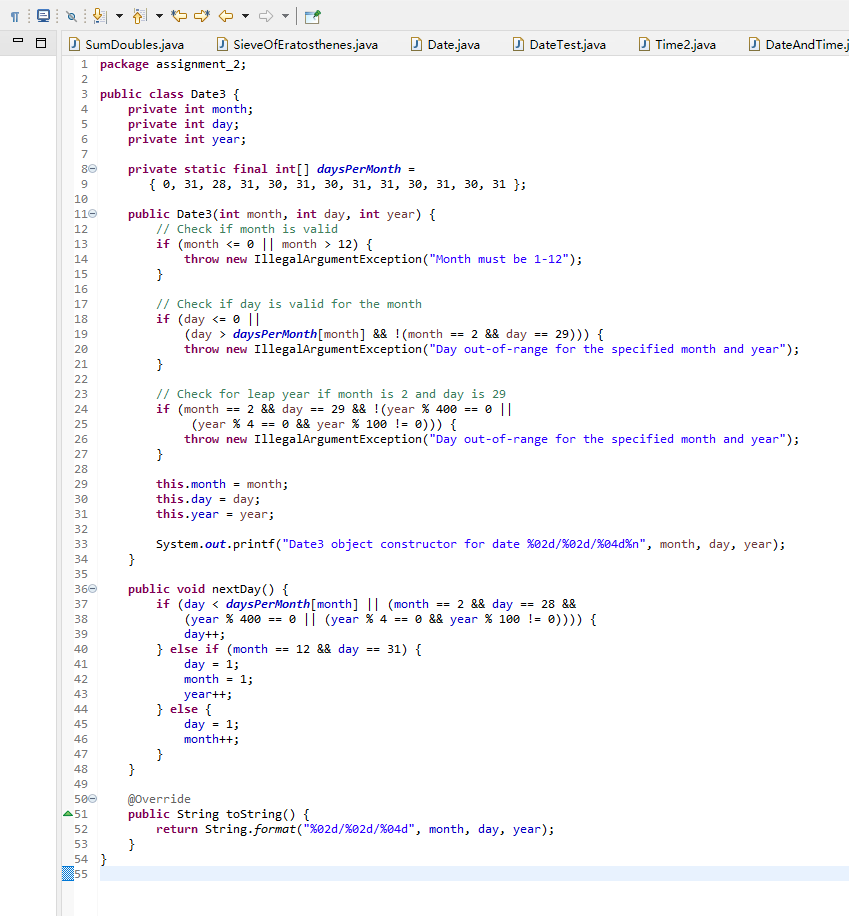
Create Clock class

Create Time3 object in main method

Call incrementMinute method until it exceeds 24 hours.

Output result

End



Q6:

Pseudo-Code

Getting Started

Create the Shape class

Declare the getArea method (abstract)

Create TwoDimensionalShape class to inherit from Shape.

Implement getArea method

Create ThreeDimensionalShape class to inherit Shape.

Declare getVolume method (abstract)

Implement the getArea method

Create concrete shape classes (e.g. Circle, Square, Sphere, Cube).

Implement getArea and getVolume methods (as appropriate).

Create ShapeTest class

Create Shape array

Add different types of shape objects

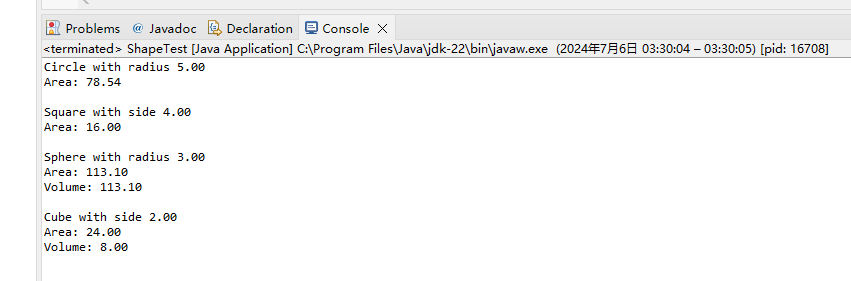
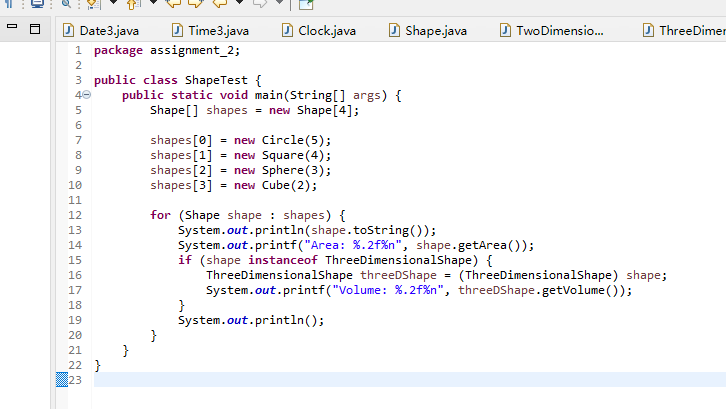
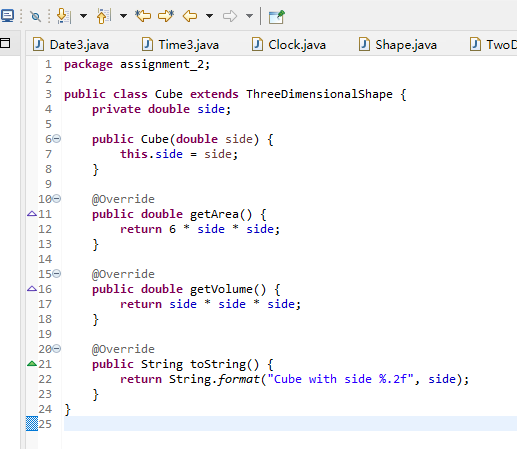
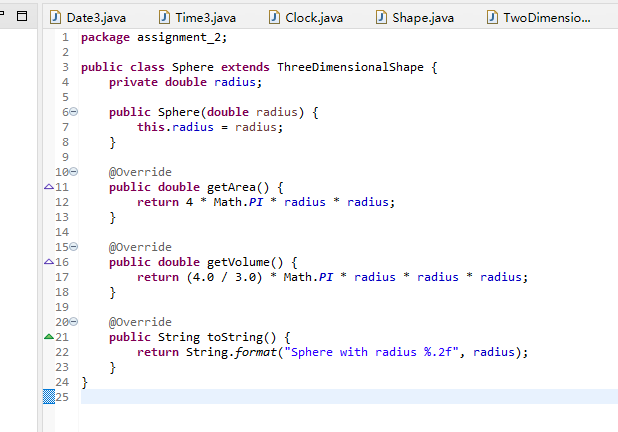
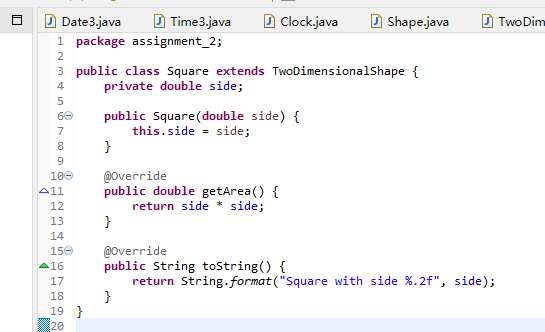
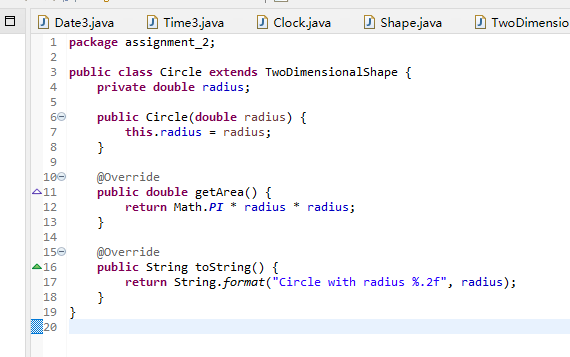
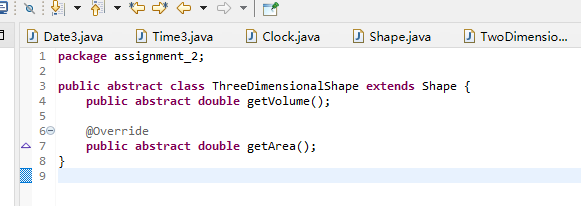
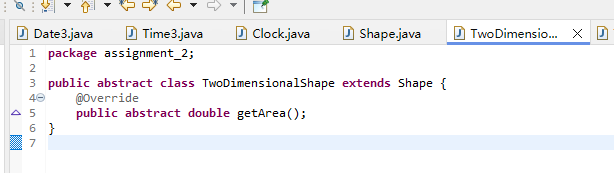
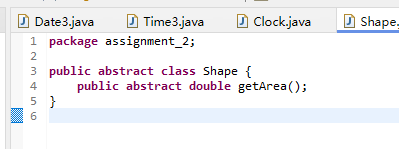
Iterate through the array

Print the shape description

Show area

If the shape is three-dimensional, display the volume

End



Q7:

Pseudo-Code

Getting Started

Creating the Employee abstract class

Declaring the earnings method (abstract)

Implement the toString method

Create the HourlyEmployee class inheriting from Employee.

Declare instance variables wage and hours

Implement the earnings method to calculate wages

Override the toString method

Create SalariedEmployee class inheriting from Employee

Declare instance variable salary

Implement earnings method to calculate salary

Override toString method

Create PieceWorker class to inherit from Employee.

Declare instance variables wagePerPiece and pieces

Implement earnings method to calculate wage

Override toString method

Create EmployeeTest class

Create Employee array

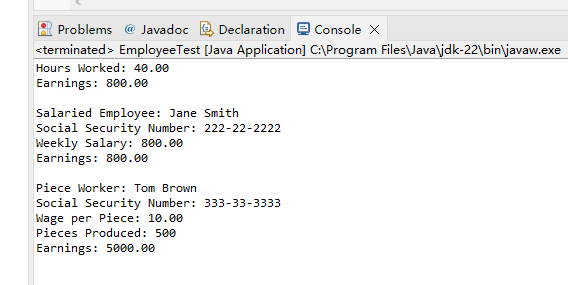
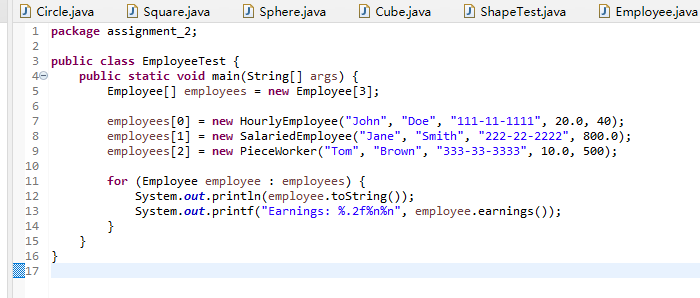
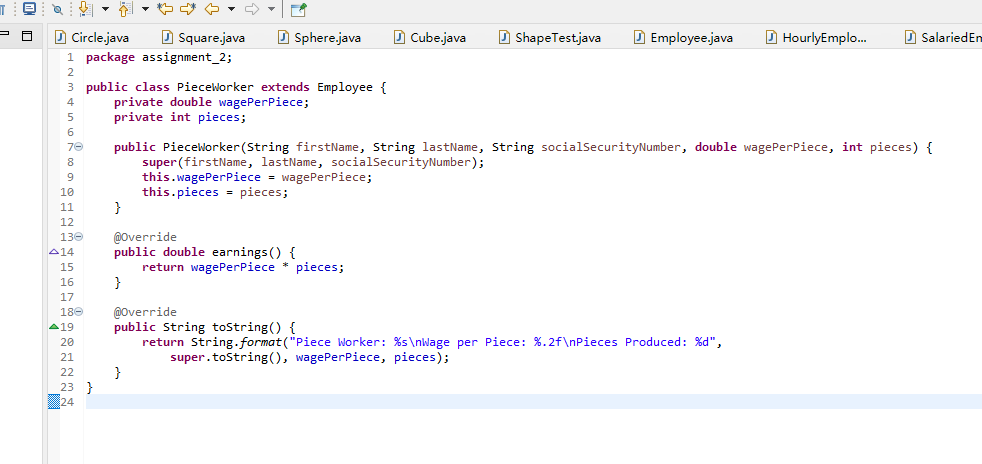
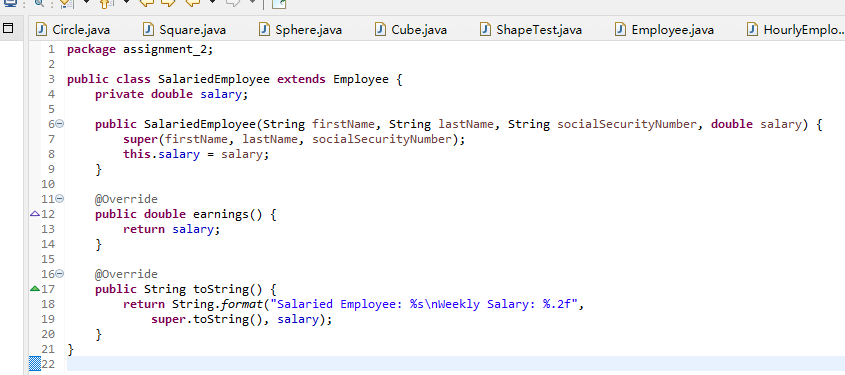
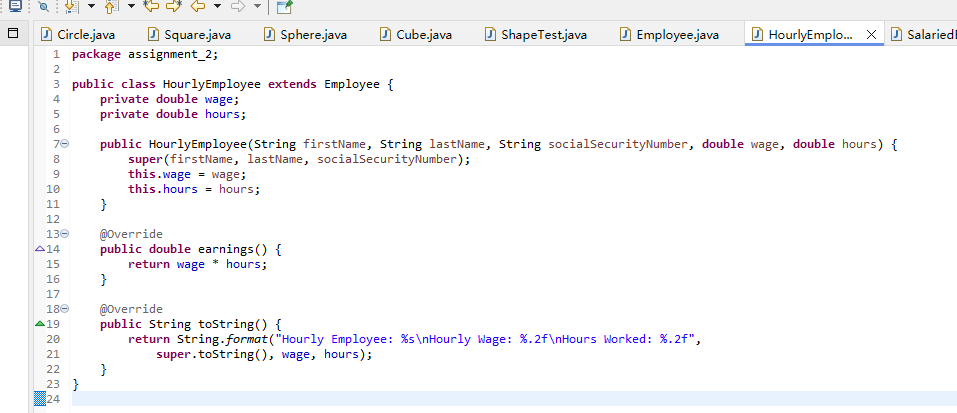
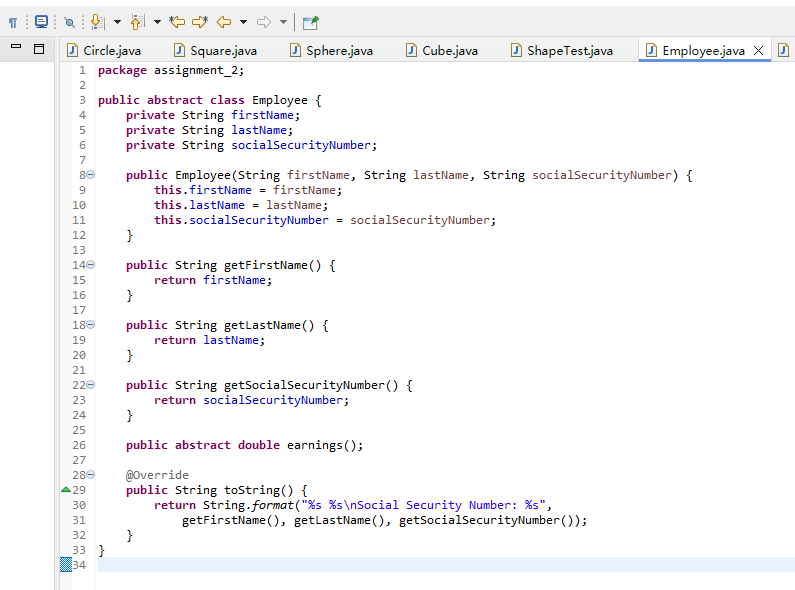
Adding different types of employee objects

Iterating through the array

Printing Employee Description

Print Employee Earnings

End



Q8:

Pseudo-Code

Getting Started

Creating the Prime Class

Write the calculatePrime method

Parameters are integers

throws an exception if the argument is negative or exceeds 10000

Returns a list of prime numbers

Create PrimeTester class

In the main method

prompts the user for an integer

Catch and handle exceptions

Call the calculatePrime method

Outputs a list of prime numbers

End

