**Q1. Find out if the given number is an Armstrong number or not.**

A screenshot of a computer

Description automatically generated

**Q2. Find out all the Armstrong numbers falling in the range of 100-999**

A screenshot of a computer

Description automatically generated

**Q3. Find out the simple as well as the compound interest of supplied value**

A screenshot of a computer

Description automatically generated

**Q4. Supply marks of three subject and declare the result, result declaration is based on below conditions:**

**Condition 1: -All subjects marks is greater than 60 is Passed**

**Condition 2: -Any two subjects marks are greater than 60 is Promoted**

**Condition 3: -Any one subject mark is greater than 60 or all subjects’ marks less than 60 is failed.**

Text

Description automatically generated

A picture containing table

Description automatically generated

A screenshot of a computer

Description automatically generated

**Q6. Consider a CUI based application, where you are asking a user to enter his Login name and password, after entering the valid user-id and password it will print the message “Welcome” along with user name. As per the validation is concerned, the program should keep a track of login attempts. After three attempts a message should be flashed saying “Contact Admin” and the program should terminate.** A screenshot of a computer

Description automatically generated

**Q7. There is an Array which is of the size 15, which may or may not be sorted. You should write a program to accept a number and search if it in contained in the array**

Text

Description automatically generated

**Q8. Using the below table write method apply sorting using Bubble Sort.**

**Example:**

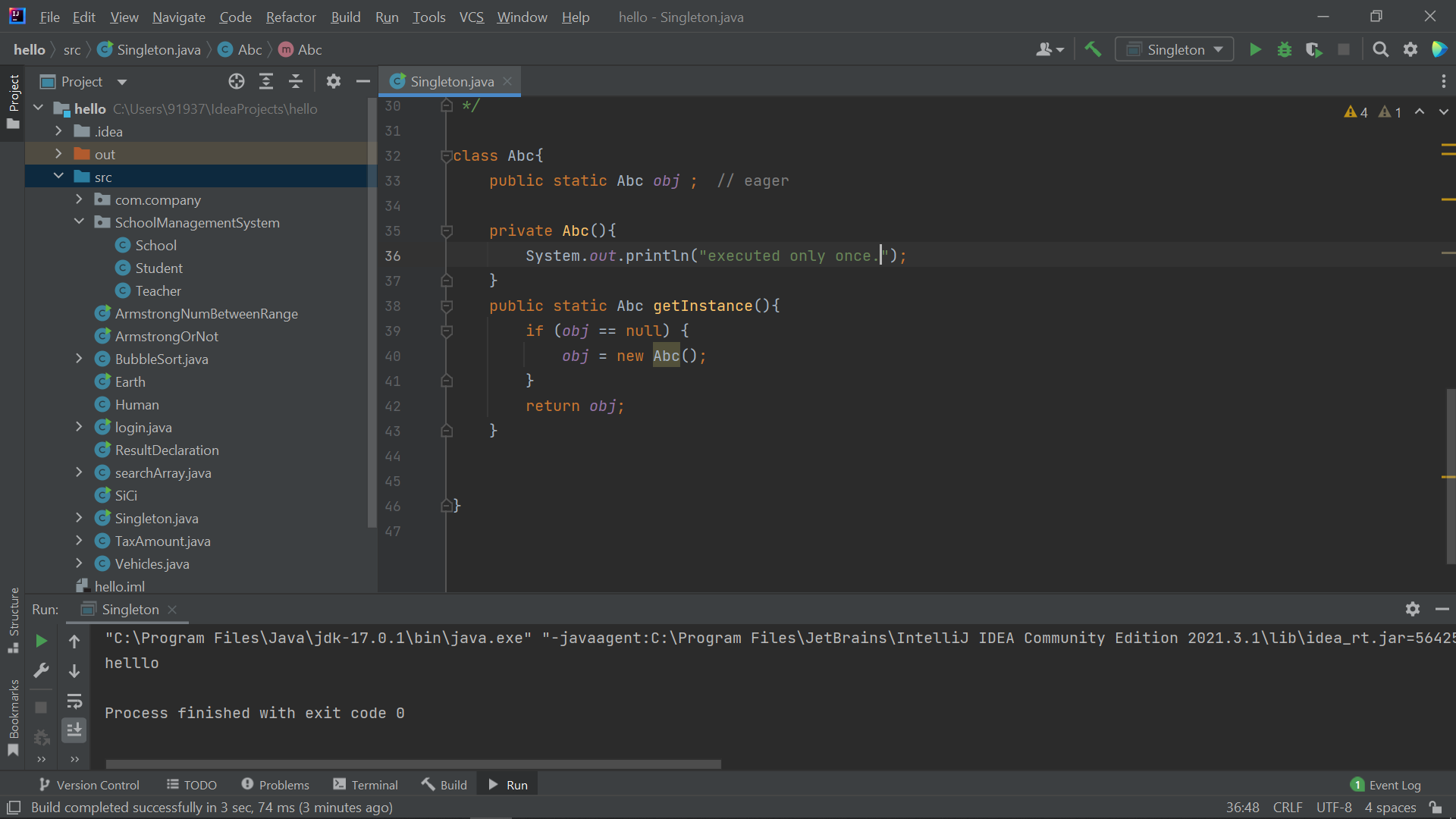
**5 12 14 6 78 19 1 23 26 35 37 7 52 86 47**

A screenshot of a computer

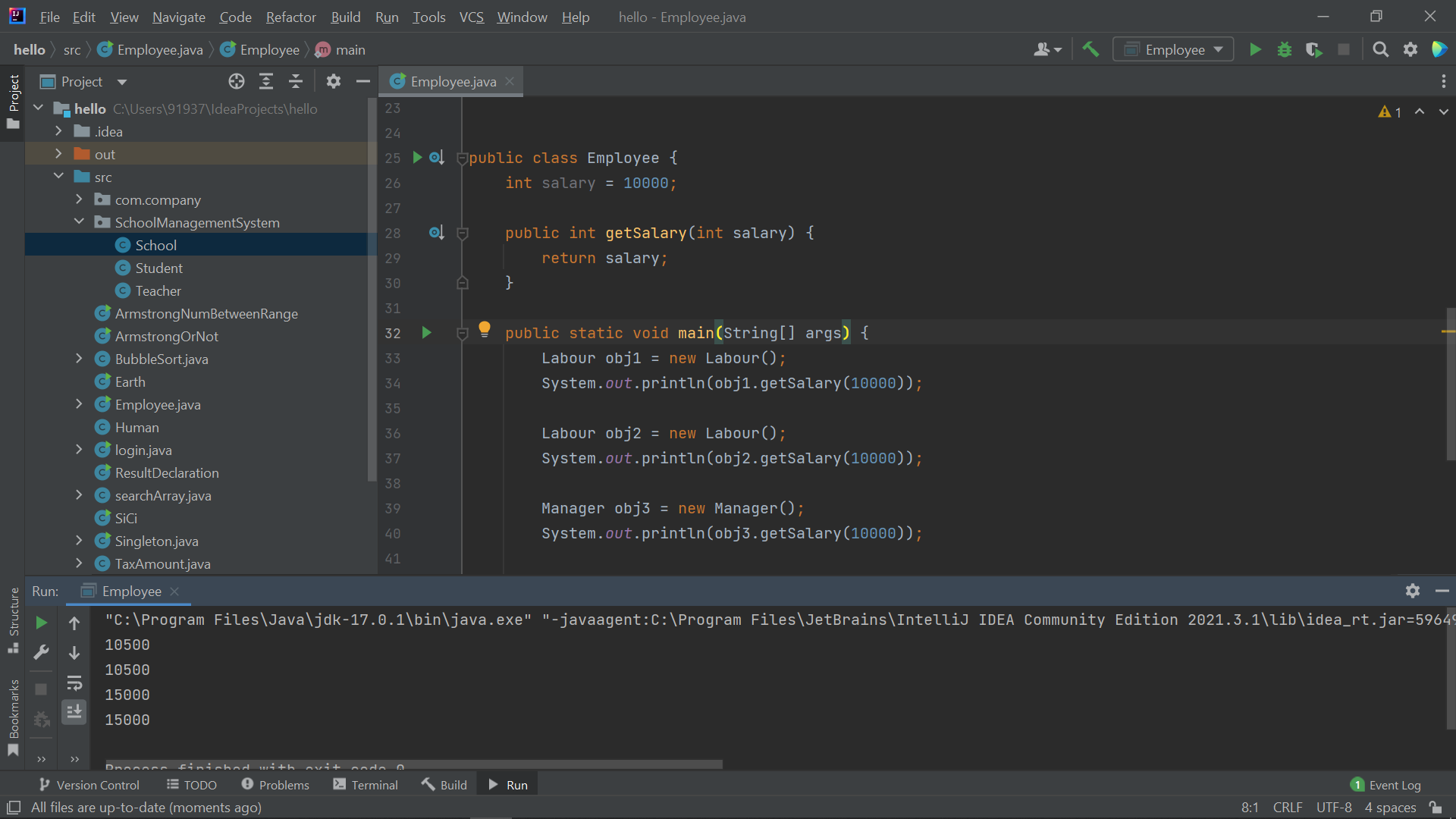
Description automatically generated with medium confidence

**INTERMEDIATE OOPS ASSIGNMENT**

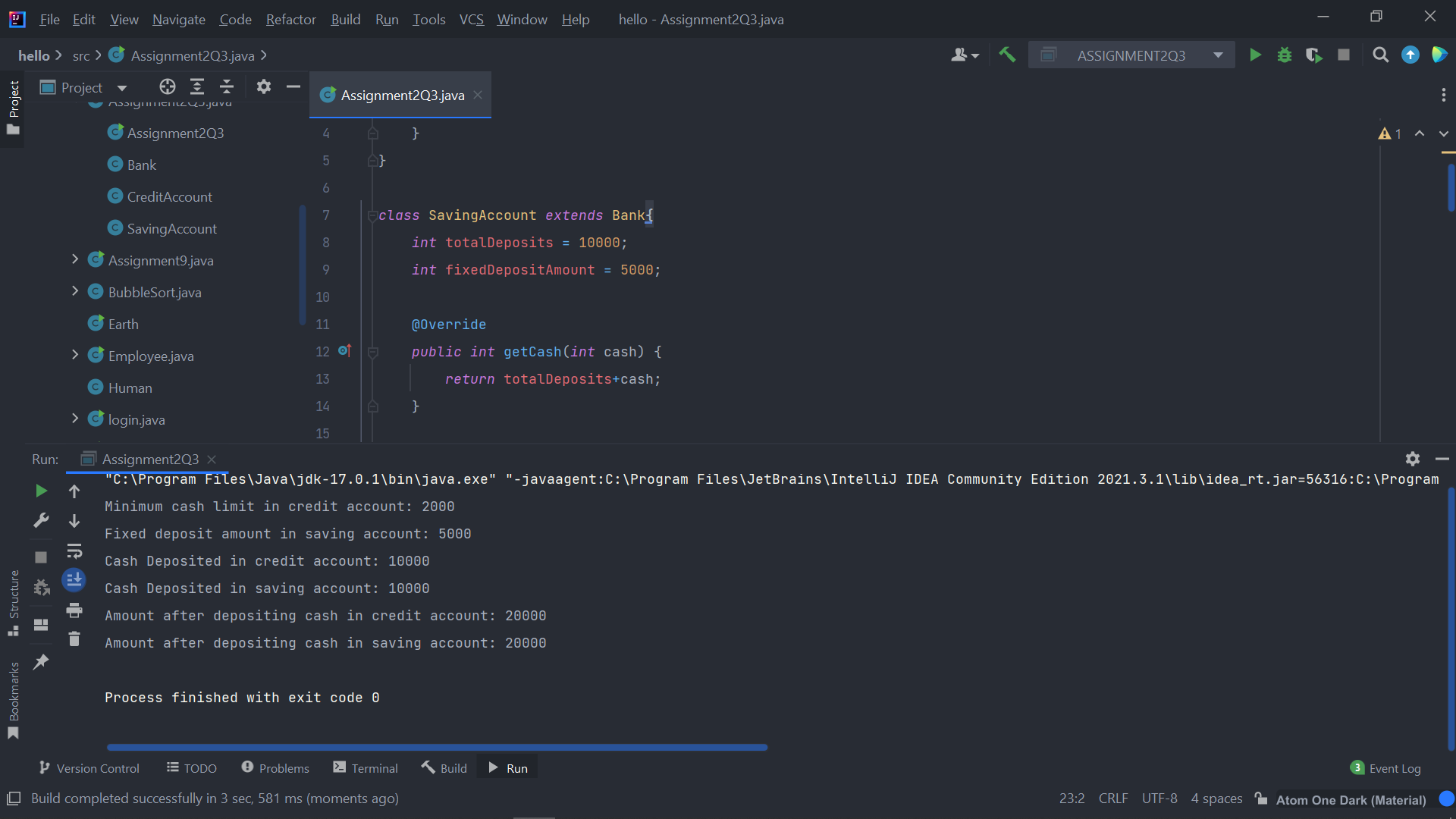
[**Q1**](https://adapt.in.capgemini.com/mod/vpl/view.php?id=2149)**. Write a singleton class. Confirm that singleton class cannot be inherited.**



**Q2. Write a program that describes the hierarchy of an organization. Here we need to write 3 classes Employee, Manager & Labour where Manager & Labour are the sub classes of the Employee. Manager has incentive & Labour has over time. Add the functionality to calculate total salary of all the employees. Use polymorphism i.e. method overriding.**



**Q3. Write a program to consider saving & current account in the bank. Saving account holder has ‘Fixed Deposits’ whereas Current account holder has cash credit. Apply polymorphism to find out total cash in the bank.**



**Q.4. Test the following principles of an abstract class:**

**• If any class has any of its method abstract then you must declare entire class abstract.**

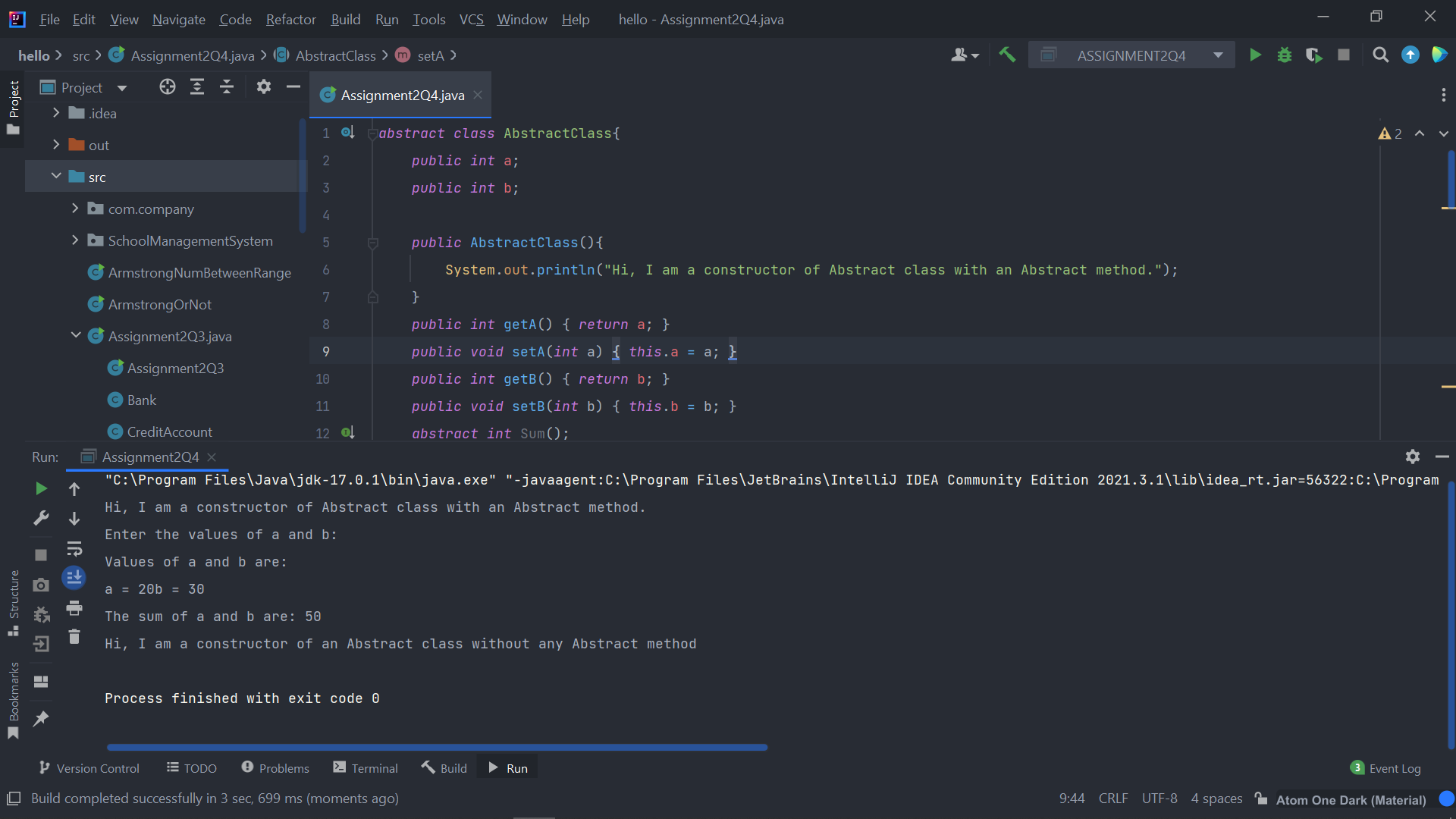
**• Abstract class cannot be instantiated.**

**• When we extend an abstract class, we must either override all the abstract methods in sub class or declare subclass as abstract.**

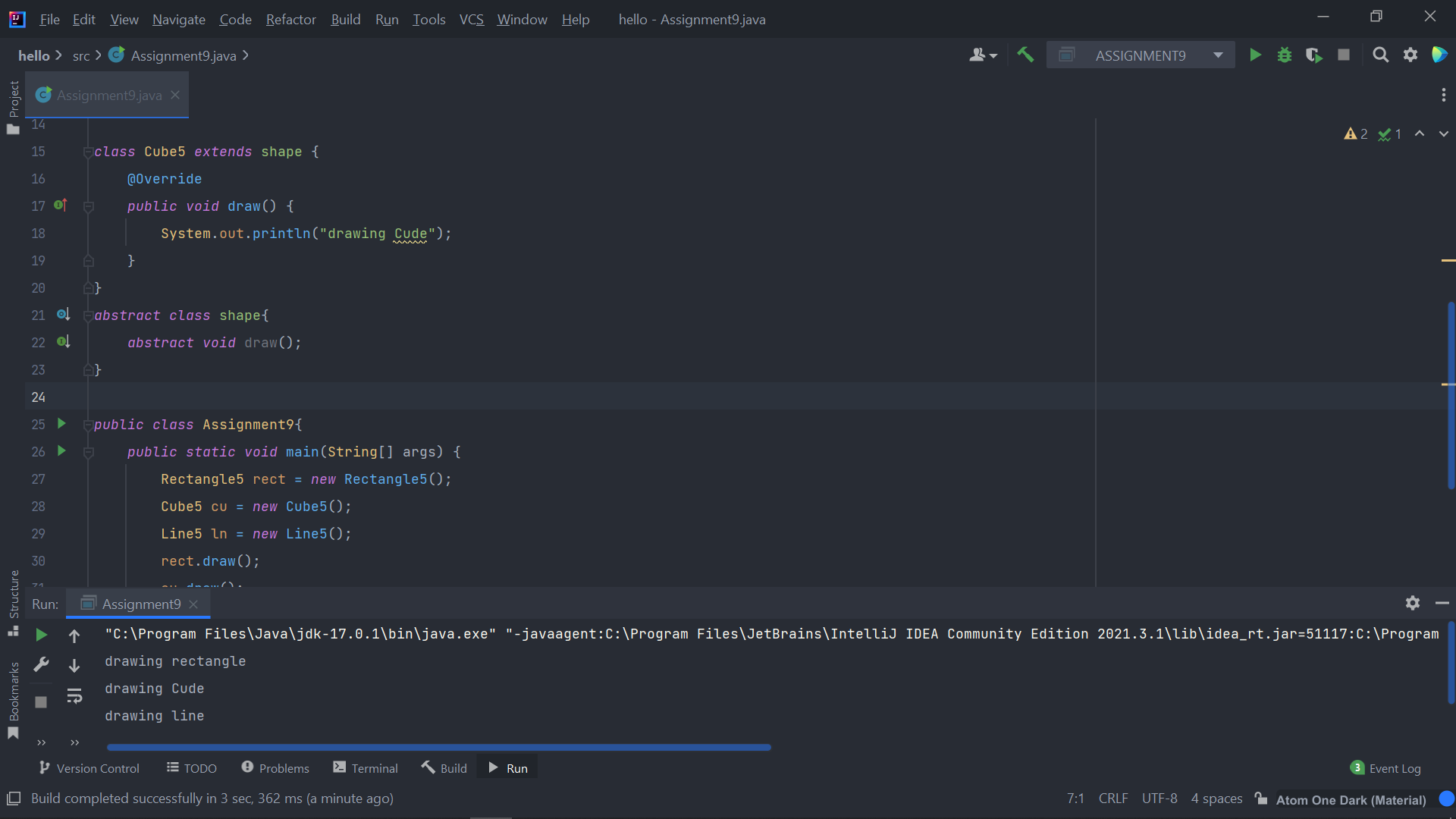
**• Abstract class cannot be private.**

**• Abstract class cannot be final.**

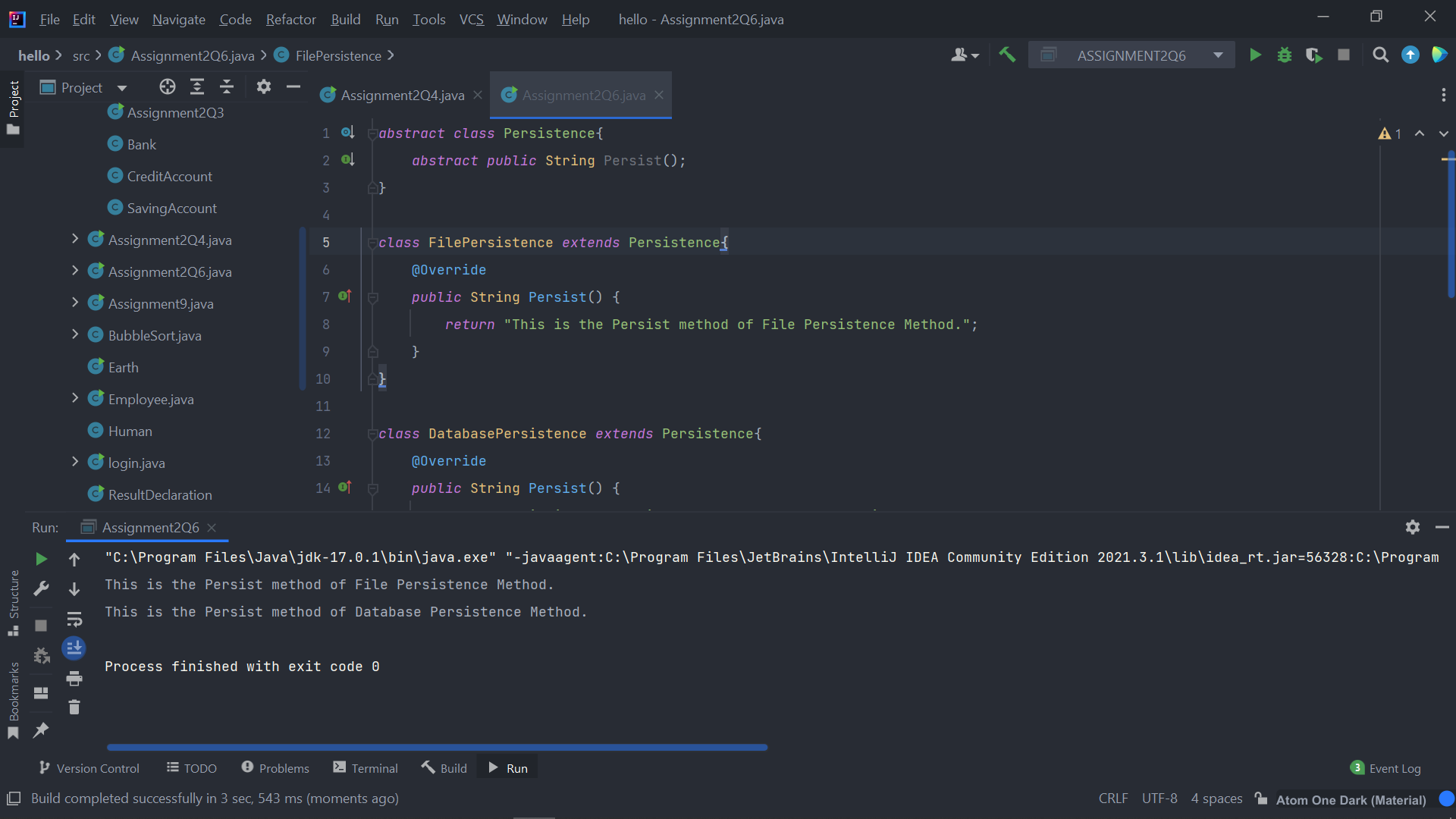
**• You can declare a class abstract without having any abstract method.**



**Q5. Write the classes Line, Rectangle, Cube etc. & make the Shape as their base class. Add an abstract draw() method in the class Shape & draw all shapes.**



**Q6. Write an abstract class ‘Persistence’ along with two sub classes ‘FilePersistence’ & ‘DatabasePersistence’. The base class with have an abstract method persist() which will be overridden by its sub classes. Write a client who gets the Persistence object at runtime & invokes persist() method on it without knowing whether data is being saved in File or in Database.**



[**Q7**](https://adapt.in.capgemini.com/mod/vpl/view.php?id=2169)**. Develop an application for Dessert shop. The application should allow owner to add items like Candy, Cookie or Ice Cream in the shop storage. Also, customers should be able to place an order.**

**Dessert Item is an abstract class having an abstract method getCost(). Every dessert item has tax associated. Candy item is sold in dollar currency, Cookie in Euro currency & Ice Cream in Rupees currency. The sub classes are supposed to override these methods. When we run the application, it should ask us our role i.e., owner or customer. If role is owner, we should be able to add dessert items in our storage. If role is customer, then we should be able to place an order. The currency conversion rates are:**

**1 dollar = 60 rupees.**

**1 euro = 70 rupees**

