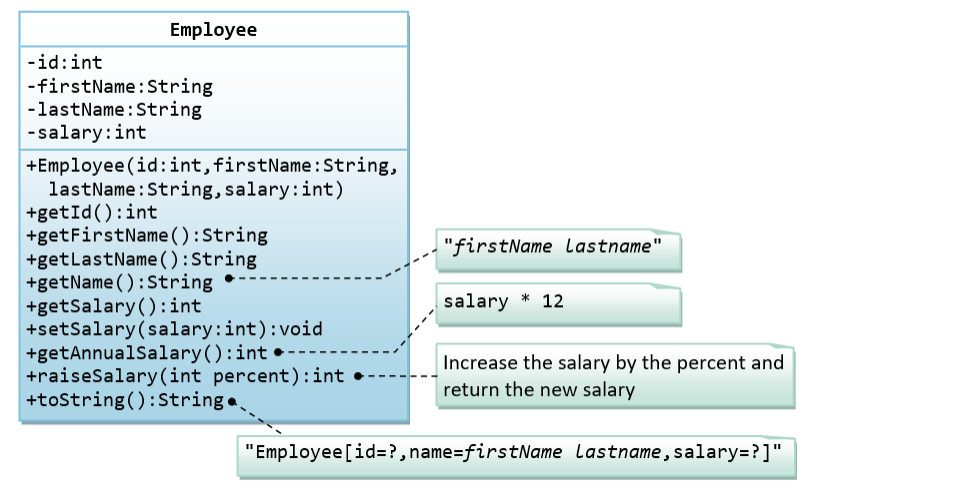
Assignment (BCAC391)

(Class composition, different type of constructors, and multiple package concept)

1. A class called Employee, which models an employee with an ID, name and salary, is designed as shown in the following class diagram. The method raiseSalary(percent) increases the salary by the given percentage. Write the Employee class.



Below is a test driver to test the Employee class:

public class TestMain {

public static void main(String[] args) {

Employee e1 = new Employee(8, "Peter", "Tan", 2500);

System.out.println(e1); //toString();

// Test Setters and Getters

e1.setSalary(999);

System.out.println(e1); // toString();

System.out.println("id is: " + e1.getId());

System.out.println("firstname is: " + e1.getFirstName());

System.out.println("lastname is: " + e1.getLastName());

System.out.println("salary is: " + e1.getSalary());

System.out.println("name is: " + e1.getName());

System.out.println("annual salary is: " + e1.getAnnualSalary());

System.out.println(e1.raiseSalary(10));

System.out.println(e1);

}

}

The expected out is:

Employee[id=8,name=Peter Tan,salary=2500]

Employee[id=8,name=Peter Tan,salary=999]

id is: 8

firstname is: Peter

lastname is: Tan

salary is: 999

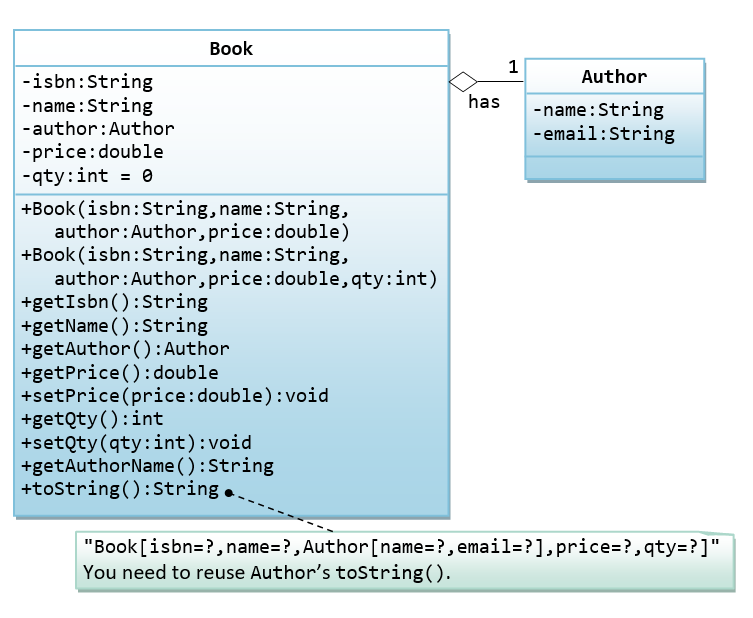
name is: Peter Tan

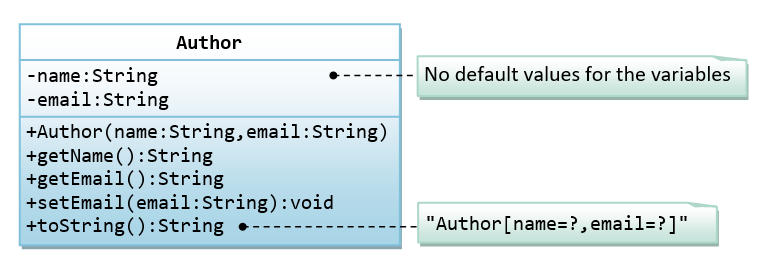
annual salary is: 11988

1098

Employee[id=8,name=Peter Tan,salary=1098]

1. A class called Author, which models an author of a book, is designed as shown in the class diagram. A class called Book, which models a book written by ONE author and composes an instance of Author as its instance variable, is also shown. Write the Author and Book classes.





Below is a test driver:

public class TestMain {

public static void main(String[] args) {

// Test Author class

Author a1 = new Author("Tan Ah Teck", "ahteck@nowhere.com");

System.out.println(a1);

a1.setEmail("ahteck@somewhere.com");

System.out.println(a1);

System.out.println("name is: " + a1.getName());

System.out.println("email is: " + a1.getEmail());

// Test Book class

Book b1 = new Book("12345", "Java for dummies", a1, 8.8, 88);

System.out.println(b1);

b1.setPrice(9.9);

b1.setQty(99);

System.out.println(b1);

System.out.println("isbn is: " + b1.getIsbn());

System.out.println("name is: " + b1.getName());

System.out.println("price is: " + b1.getPrice());

System.out.println("qty is: " + b1.getQty());

System.out.println("author is: " + b1.getAuthor()); // Author's toString()

System.out.println("author's name: " + b1.getAuthorName());

System.out.println("author's name: " + b1.getAuthor().getName());

System.out.println("author's email: " + b1.getAuthor().getEmail());

}

}

The expected output is:

Author[name=Tan Ah Teck,email=ahteck@nowhere.com]

Author[name=Tan Ah Teck,email=ahteck@somewhere.com]

name is: Tan Ah Teck

email is: ahteck@somewhere.com

Book[isbn=12345,name=Java for dummies,Author[name=Tan Ah Teck,email=ahteck@somewhere.com],price=8.8,qtt=88]

Book[isbn=12345,name=Java for dummies,Author[name=Tan Ah Teck,email=ahteck@somewhere.com],price=9.9,qtt=99]

isbn is: 12345

name is: Java for dummies

price is: 9.9

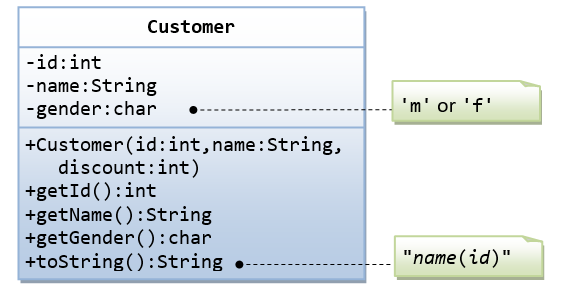
qty is: 99

author is: Author[name=Tan Ah Teck,email=ahteck@somewhere.com]

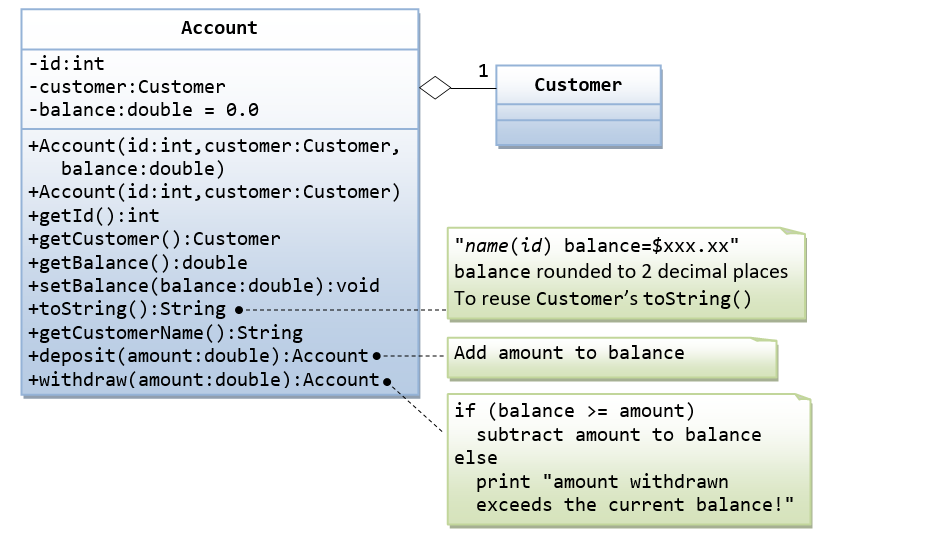
author's name: Tan Ah Teck

author's name: Tan Ah Teck

author's email: [ahteck@somewhere.com](mailto:ahteck@somewhere.com)



The Customer class models a customer is design as shown in the class diagram. Write the codes for the Customer class and a test driver to test all the public methods.



The Account class models a bank account, design as shown in the class diagram, composes a Customer instance (written earlier) as its member. Write the codes for the Account class and a test driver.

Below is test Driver:

import com.account.pkg.Account;

import com.customer.pkg.Customer;

public class testDrive {

public static void main(String[] args) {

// TODO code application logic here

Customer customer1=new Customer(1001,"Susovan Kumar Pan",'m');

System.out.println(customer1);

Account account=new Account(1001,customer1,500);

account.deposite(100);

account.withdraw(50);

System.out.println(account);

account.withdraw(600);

//System.out.println(account);

}

}

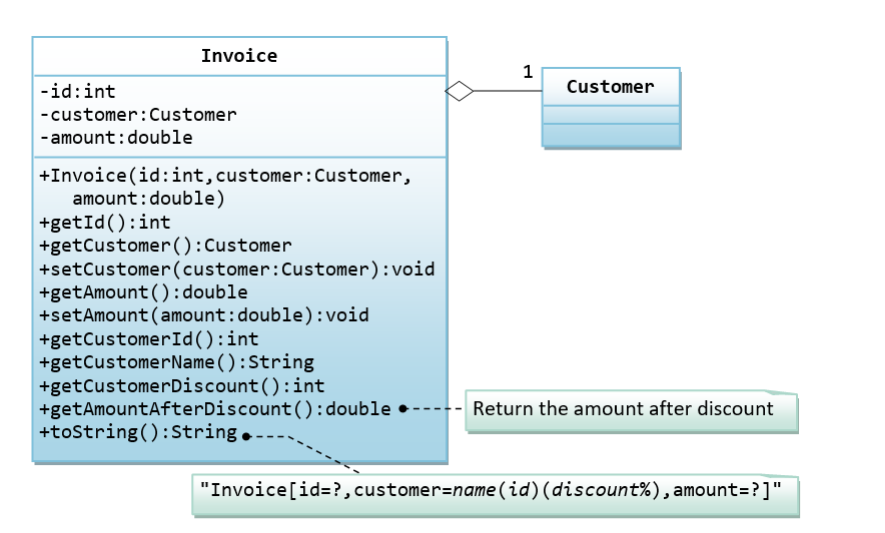
The expected output is:

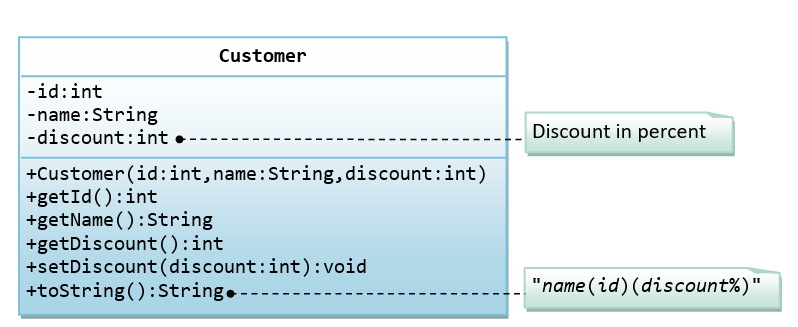
Susovan Kumar Pan(1001)

Susovan Kumar Pan(1001) balance=$550.0

amount withdrawn exceeds the current balance!

1. A class called Customer, which models a customer in a transaction, is designed as shown in the class diagram. A class called Invoice, which models an invoice for a particular customer and composes an instance of Customer as its instance variable, is also shown. Write the Customer and Invoice classes.





Below is a test driver:

public class TestMain {

public static void main(String[] args) {

// Test Customer class

Customer c1 = new Customer(88, "Tan Ah Teck", 10);

System.out.println(c1); // Customer's toString()

c1.setDiscount(8);

System.out.println(c1);

System.out.println("id is: " + c1.getId());

System.out.println("name is: " + c1.getName());

System.out.println("discount is: " + c1.getDiscount());

// Test Invoice class

Invoice inv1 = new Invoice(101, c1, 888.8);

System.out.println(inv1);

inv1.setAmount(999.9);

System.out.println(inv1);

System.out.println("id is: " + inv1.getId());

System.out.println("customer is: " + inv1.getCustomer()); // Customer's toString()

System.out.println("amount is: " + inv1.getAmount());

System.out.println("customer's id is: " + inv1.getCustomerId());

System.out.println("customer's name is: " + inv1.getCustomerName());

System.out.println("customer's discount is: " + inv1.getCustomerDiscount());

System.out.printf("amount after discount is: %.2f%n", inv1.getAmountAfterDiscount());

}

}

The expected output is:

Tan Ah Teck(88)(10%)

Tan Ah Teck(88)(8%)

id is: 88

name is: Tan Ah Teck

discount is: 8

Invoice[id=101,customer=Tan Ah Teck(88)(8%),amount=888.8]

Invoice[id=101,customer=Tan Ah Teck(88)(8%),amount=999.9]

id is: 101

customer is: Tan Ah Teck(88)(8%)

amount is: 999.9

customer's id is: 88

customer's name is: Tan Ah Teck

customer's discount is: 8

amount after discount is: 919.91