

CS401 MPP Midterm

Corazza

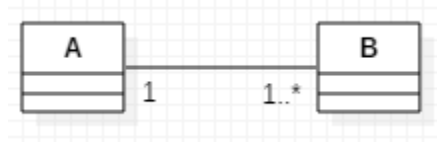
Name: _____

StudentId: _____

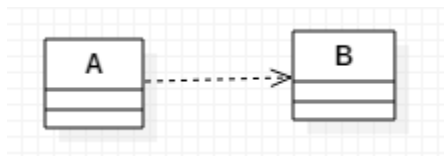
Part I (24)	Part II.1 (8)	Part II.2 (8)	Part II.3 (12)	Part III SCI (3)

Part I: Short Answer (3 points each)

- ___ 1. (T/F) To implement the class diagram below in code, a list of type A objects must be placed inside the class B.



- ___ 2. (T/F) If the class diagram below has been implemented in code, the following must be true at runtime: When an instance of class A is created, it keeps a reference to class B.



- ___ 3. A Sequence Diagram shows the flow of communication between the running objects of the system, driven by the use cases of the system.

- ___ 4. What happens when the main method in the following code is executed?

```
public class Base extends Extension {
    public static void main(String[] args) {
        Extension s = new Base();
        s.print();
    }
    public void print() {
        System.out.println("From Base");
    }
}

public class Extension {
    void print() {
        System.out.println("From Extension");
    }
}
```

- A. There is a compiler error.
- B. There is a runtime error.
- C. "From Base" is printed to the console.
- D. "From Extension" is printed to the console.

___ 5. What happens when the main method in the following code is executed?

```
public class Base extends Extension {  
    public static void main(String[] args) {  
        Extension s = new Base();  
        s.print();  
    }  
    public static void print() {  
        System.out.println("From Base");  
    }  
}  
  
public class Extension {  
    static void print() {  
        System.out.println("From Extension");  
    }  
}
```

- A. There is a compiler error.
 - B. There is a runtime error.
 - C. "From Base" is printed to the console.
 - D. "From Extension" is printed to the console.
6. Four Use Cases that were described in the Library System project for this class were Login, Add New Member, Add Book Copy, Checkout Book. Draw a single Use Case Diagram for the Library System using just these four use cases. Show all actors.

7. Consider the following Customer class. It contains instance variables of type Account, LocalDate, and List<Double>. (The Account class is also shown.)

```
public class Customer {  
    private Account checkingAccount;  
    private LocalDate birthdate;  
    private List<Double> thisYearsSalaries;  
    public Customer(Account checking, LocalDate bdate,  
        List<Double> salaries) {  
        this.checkingAccount = checking;  
        this.birthdate = bdate;  
        this.thisYearsSalaries = salaries;  
    }  
}  
  
public class Account {  
    private double balance;  
}
```

If this Customer class is modeled in a class diagram:

- a. Which instance variables should be modeled as *attributes*?
- b. Which instance variables should be modeled as *associations*?

Explain your answer.

8. Draw the Object Creation Factory diagram in the space below. Then answer the following:

The following lines of code make use of one of the factory methods in Collections. Compare these with the Object Creation Factory pattern diagram, and answer the following:

1. Which class plays the role of Factory?
2. Which class plays the role of Client?
3. Which interface plays the role of Product?
4. Which class plays the role of Concrete Product?

```
public class MyClass {  
    public static void main(String[] args) {  
        List<String> list = Arrays.asList("Joe", "Bill", "Tom");  
        List<String> unList = Collections.unmodifiableList(list);  
    }  
}
```

1

Part II: Skill Questions

1. [8 pts] A file system for an operating system on a computer is set up so that files can be organized into directories that have a hierarchical structure. The file system satisfies the following requirements:
 - A directory may contain zero or more other directories and zero or more files
 - There is a top-level directory that is not a subdirectory of any other directory
 - Every file belongs to exactly one directory and every directory, other than the top-level directory, belongs to exactly one other directory.

Draw a class diagram that models this kind of file system. Your diagram should consist of two classes together with the correct type of association(s) and multiplicities.

//Your solution goes here

2. [8 pts] The diagram below shows that (for a particular application) there is a one-one bidirectional association between a UI class and a Controller class.



In the space provided below, write Java code that implements this diagram. Assume that UI and Controller are the only classes in a particular package. Your code must meet the following requirements:

- The UI class owns the relationship, so it should not be possible to create an instance of Controller independently of an already existing UI class
- The code must show relevant instance variables, constructor implementations, and methods, sufficient to implement this model. (Show only those instance variables that are implied by the diagram. Getters for instance variables should be provided.)
- All classes, properties, methods, and constructors must be given appropriate visibility qualifiers (private, protected, public, or package level).

```
public class UI {
```

```
}
```

```
public class Controller {
```

```
}
```

3. [12 pts] In a company, employees may have multiple bank accounts: zero or more savings accounts and zero or more checking accounts. Each checking account has an account id, a balance, and a monthly fee. Each savings account has an account id, a balance, and an interest rate associated with the particular type of savings account. It is possible to read the current balance in any of these accounts, but it is also possible to determine the balance after interest or monthly fee is applied by calling the `computeUpdatedBalance` method on the account.

An administrator has access to all employee records and from time to time computes the total balance across all employee-owned accounts; for each account, the balance that is needed in this computation is the *updated* balance. This computation is performed in the static method `computeUpdatedBalanceSum`

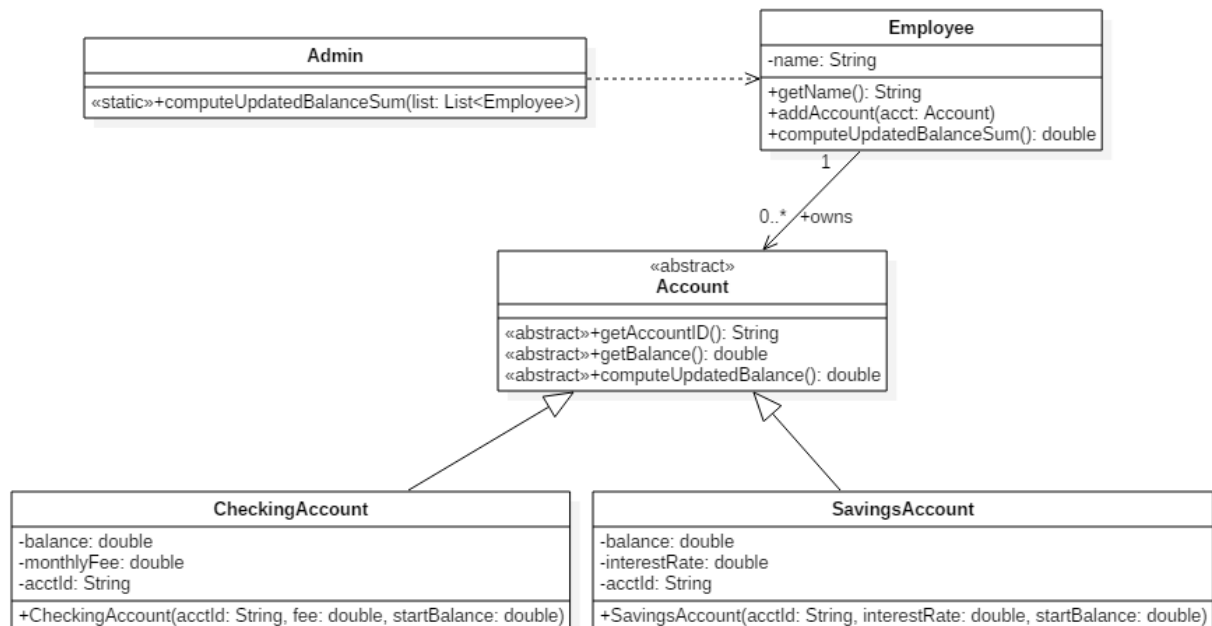
in the `Admin` class.

Below is a class diagram showing the classes involved and relationships between them. A sequence diagram for the operation `computeUpdatedBalanceSum` is also provided. Your task in this problem is to write Java code that implements the classes and relationships shown in the diagram. Shells for `Admin` and `Employee` have been provided for you (below).

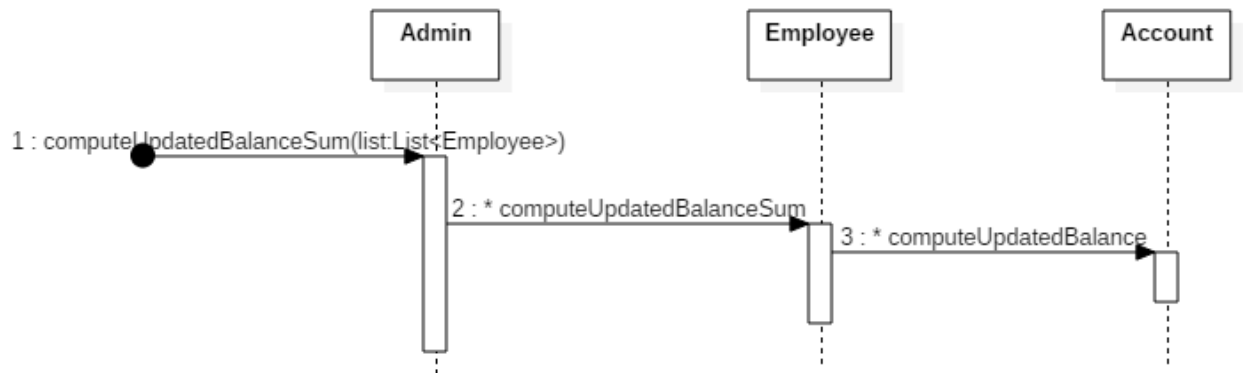
The method `computeUpdatedBalance` in `CheckingAccount` does the following computation to obtain the return value:

$$\text{balance} - \text{monthlyFee}.$$

The method `computeUpdatedBalance` in `SavingsAccount` performs the following computation to obtain the return value:

$$\text{balance} + (\text{interestRate} * \text{balance}).$$


interaction Sequence Diagram for computeUpdatedBalanceSum



//Your code for Problem 3 should begin here

```
public class Admin {
    public static double computeUpdatedBalanceSum(List<Employee> list) {
```

```
    }
}
```

```
public class Employee {
    public double computeUpdatedBalanceSum() {
```

```
    }
}
```


Part III: SCI. [3 pts] In an ancient text, one reads the following:

Know that by which all this is known.

1. What does this expression mean? What is it saying? Is it some kind of SCI point?
2. Does this expression illuminate any aspect of the software engineering discipline discussed in class? Explain.