CS2102, B12

Exam 2

Name:

You have 50 minutes to complete the problems on the following pages. There should be sufficient space provided for your answers.

If a problem asks you to create an interface, you should provide a complete interface, including method headers and argument types.

If a problem asks you to create a class:

- Include **implements** and **extends** statements
- Include field names and types
- Omit constructors
- Omit methods unless a problem asks otherwise

Omit the `Examples` class (examples of data and test cases) unless a question asks otherwise.

If a problem asks you to choose a data structure, provide the type of content as well as the name of the data structure (for example, a LinkedList<String>, a HashMap<String,Dillo>, a graph with People as nodes). You may indicate the content types either in code or in prose.

---

1. **Topic: Creating and Encapsulating Data**

Recall the banking system we worked with earlier in the term: each `Account` contained a `Customer` and each `Customer` contained a list of his `Accounts`. The classes appear below, this time with constructors.

```
class Customer {
  String name;
  int password;
  LinkedList<Account> accounts;

  Customer(String name, LinkedList<Account> accounts) {
    this.name = name;
    this.accounts = accounts;
    this.password = genPassword();
  }
}

class Account {
  int number;
  Customer owner;
  double balance;

  Account(int number, Customer owner) {
    this.number = number;
    this.owner = owner;
    this.balance = 0;
  }
}
```

addAccount adds an Account to the accounts list in a Customer

Someone named Alice wants to register as a new customer and simultaneously open a new account. Write code to do this using calls to the constructors provided. If you need code beyond calls to the constructors, (a) explain why, and (b) show the code. For encapsulation points, write the extra code using appropriate new methods and state the purposes of those methods (you do not need to write the methods themselves, just purposes).

```
Customer A = new Customer ("Alice",
                           new LinkedList<Account>())
A.addAccount(new Account(123, A))
```

we need to use Add Account because the data is cyclic, and hence must be created in stages.

(exam continues next page)