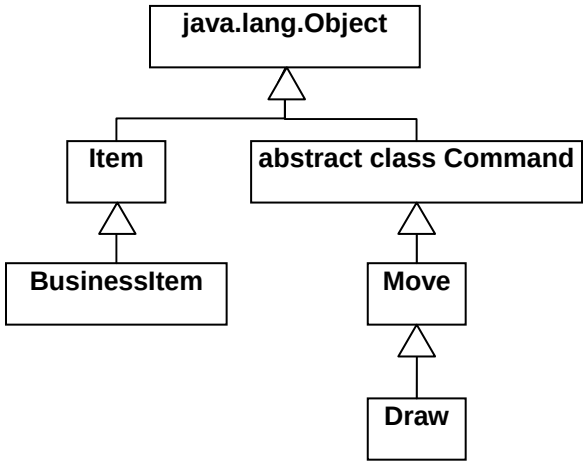
CS2102 B2006 Exam #2 NAME _____

Q1. [12 pts.] Given a base class *B*, a derived class *C*, and a method **m** of the derived class *C*:

a) Method **m** is an overloaded method. Explain what this means in relation to either class B or C.

b) Method **m** overrides another method. Explain what this means in relation to either class B or C.

Q2. [25 pts.] You are given the following definitions (all classes belong to the same package). Assume each class has a no-argument constructor. Assume you already have constructed the following two objects:




```
BusinessItem bi = new BusinessItem ();
Draw d = new Draw();
```

Which of the following five statements are legal? For the **illegal statements only**, explain why the statement is illegal:

```
S1.  BusinessItem x = new Item ();
S2.  boolean b = d.equals(bi);
S3.  Object o = new Command();
S4.  Command c = new Draw();
S5.  Command y = new Item();
```

Q3. [25 pts] Design a *RationalNumber* class that represents fraction values **n**/**d** where **n** and **d** are integers > 0. You must define the following constructor and four instance methods. **No Documentation is Necessary.** Each of the following methods is **5 pts**.

- A *RationalNumber* **constructor** that takes **int n** and **int d**, both numbers > 0
- An **equals (Object o)** instance method to compare if another *RationalNumber* object is "equal to" this one
- A **toString()** instance method that produces String representations of the form "n/d"
- An **add (RationalNumber rn)** instance method that adds the given *RationalNumber* rn to update this object's value. Note rn is unchanged. *You don't need to reduce the resulting fraction to its simplest form. Thus 2/3 + 1/6 can result in 15/18*
- A **multiply (RationalNumber rn)** instance method that multiplies this object by the given *RationalNumber* rn. Note rn is unchanged. *You don't have to reduce the resulting fraction to its simplest form. Thus 2/3 * 3/5 can result in 6/15*

```
/** Represents a Rational Fraction n/d where n and d are integers > 0. */
public class RationalNumber {
  private int n;      /** The numerator. */
  private int d;      /** The denominator. */
```



```
}
```

Q4. [22 pts.] Given the following *NumberNode* and *NumberList* classes and an **add** (**int** i) method in *NumberList* that **prepends** numbers to the linked list start with **head**.

```
/** Node class representing an integer. */
public class NumberNode {
  int          value;    /* node value. */
  NumberNode   next;     /* next one. */

  public NumberNode (int i) {
    value = i;
    next = null;
  }
}
```

```
/** List of numbers. */
public class NumberList {
  NumberNode head;  /* first one. */

  /** Prepend NumberNode with i to list. */
  public void add (int i) { … }

}
```

Write an instance method **collapse** for *NumberList* that removes all sequential repetitions of numbers in the list starting with head. For example, consider the *NumberList* resulting from: **add(5); add(1); add(1); add(1); add(4); add(2); add(2),** represented by {2, 2, 4, 1, 1, 1, 5}. After calling **collapse** on this *NumberList* object, the list becomes {2, 4, 1, 5}.

(a) [**16 pts**.] Implement the **collapse** method. **Provide No Documentation.**

(b) [**6 pts.**] Explain why the **collapse** method should in *NumberList* and not *NumberNode*.

```
/** Node class representing an integer. */
public class NumberNode {
  int          value;    /* node value. */
  NumberNode   next;     /* next one. */

  public NumberNode (int i) {
    value = i;
    next = null;
  }
}
```

```
/** List of numbers. */
public class NumberList {
  NumberNode head;  /* first one. */

  /** Prepend NumberNode with i to list. */
  public void add (int i) { … }

}
```

Q5. [14 pts] These questions use the *NumberNode* and *NumberList* classes, shown again above.

(a) [6 pts.] If the **value** and **next** instance variables in *NumberNode* were changed to be **private** instance variables, explain the changes you would need to make to the *NumberNode* class.

(b) [8 pts.] The following method **void strangeMethod()** exists in the *NumberList* class. After it is invoked on a *NumberList* object, what is the resulting linked list maintained by that object?

```
public void strangeMethod() {
   head = new NumberNode(3);
   NumberNode node = head.next;
   node = new NumberNode(5);
   node.next = head;
}
```

Q6. [2 pts.] Fill in the Dilbert cartoon. If my eyes burn from the radiance of your joke, you get extra points. Or, write the name of **one** of the musical selections from the daily objectives.