CS 1110
Prelim I: Review Session

Exam Info
• Prelim 1: 7:30–9:00PM, Thursday, 8 October, Olin 255 (Last name starts with A-K) and Upson B17 (Last name starts with L-Z)
• Look at the previous Prelims
• Arrive early! Helps reducing stress
• Grades released the same evening (morning...)

What’s in the exam?
• Definitions of terms and key concepts
• Execution of assignment statements
• Evaluation / Execution of “new” expressions
• Evaluation of method calls
• Execute sequence of statements
• String functions
• Writing class definitions
• See “About the prelim” on course website

We have a lot to “cover”

What’s in the exam?
• Definitions of terms and key concepts
• Execution of assignment statements
• Evaluation / Execution of “new” expressions
• Evaluation of method calls
• Execute sequence of statements
• String functions
• Writing class definitions
• See “About the prelim” on course website

Definitions
• Usually the first question of the exam
• Short answers
• Questions ask for definitions or something to be done
• Let’s see 3 examples from Prelim I, spring’07.
Definitions
• (a) **5 pts.** What is an argument? A parameter?
  A parameter is a variable declared in the header of a method (within the parentheses). An argument is an expression that occurs in a call of a method.

• (b) **5 pts.** What is a local variable? What is its scope?
  A local variable is a variable that is declared in the body of a method. Its scope begins at its declaration and continues until the end of the block in which it is declared.

Definitions
• (a) **5 pts.** What is an argument? A parameter?

• (b) **5 pts.** What is a local variable? What is its scope?

What’s in the exam?
• Definitions of terms and key concepts
• Execution of assignment statements
• Evaluation / Execution of “new” expressions
• Evaluation of method calls
• Execute sequence of statements
• String functions
• Writing class definitions

See “About the prelim” on course website
Assignments

1. **Execution of an assignment statement** stores a value in a variable.

   - **To execute the assignment**
     
     ```
     <var> = <expr>;
     ```
     
     evaluate expression `<expr>` and store its value in variable `<var>`.

   - **Example**
     
     ```
     x = x + 1;
     ```
     Evaluate expression `x+1` and store its value in variable `x`.

2. **Define**

   - **Execute**
     
     ```
     <var> = <expr>;
     ```
     Evaluate expression `<expr>` and store its value in variable `<var>`.

```java
v1 = 5;
// v1: 5
v2 = 3;
// v1: 5
v1 = v1 + 1;
// v1: 6
v2 = v1 + 42;
// v1: 51
```

**What’s in the exam?**

- Definitions of terms and key concepts
- Execution of assignment statements
- Evaluation / Execution of “new” expressions
- Evaluation of method calls
- Execute sequence of statements
- String functions
- Writing class definitions
- See “About the prelim” on course website
  
  [Link](http://www.cs.cornell.edu/courses/cs1110/2009fa/exams/prelim1/aboutprelim1.pdf)

**Execute “new” statements**

```java
/**
   * An instance represents a Student
   */
public class Student {
    // the student's name
    private String name;
    // the student's number
    private String ne_d;
    /**
     * Constructor: a Person with name n and ne_d i
     */
    public Student(String n, String i) {
        name = n;
        ne_d = i;
    }
    /**
     * Set the Student's name to n
     */
    public void setName(String n) {
        name = n;
    }
    /**
     * =: this Student and s have the same ne_d
     */
    public boolean equals(Student s) {
        return ne_d.equals(s.ne_d);
    }
    /**
     * Get the Student's name
     */
    public String getName() {
        return name;
    }
    @Override
    public String toString() {
        return name + " | " + ne_d;
    }
}
```

**Question 4 (20 points)**. Below is a definition of class Student. Assume the following three assignment statements are executed:

- Student p1 = new Student("Bill", "bk12");
- Student p2 = new Student("Bill", "bk13");
- Student p3 = new Student("William", "bk12");

(a) What is the value of each of the following four expressions?

   - p1.equals(p2)
   - p1.equals(p3)
   - p1 == p2
   - p1 == p3

(b) Now consider these statements:

   ```
   p1 = new Student("Bill", "bk12");
   p2 = new Student("Bill", "bk13");
   p3 = p2;
   p3.setName("Jack");
   ```

   Below, first draw all three variables. Then execute the four statements — of course, draw any objects that are created during execution.

**A message from Profs. Gries and Lee**

- You won’t get these questions correct unless you draw the variables, objects, and execute their assignments!

**A message from Profs. Gries and Lee**

- You won’t get these questions correct unless you draw the variables, objects, and execute their assignments!
(a) What is the value of each of the following four expressions?
   - p1.equals(p2)
   - p1.equals(p3)
   - p1 == p2
   - p1 == p3

   false  true  false  false

(b) Now consider these statements:
   - p3 = p2;
   - p3.setName("Jack");

   Below, first draw all three variables. Then execute the two statements.

   n1  p1  n2  p2
   n3  p3

What's in the exam?
- Definitions of terms and key concepts
- Execution of assignment statements
- Evaluation / Execution of "new" expressions
- Evaluation of method calls
- Execute sequence of statements
- String functions
- Writing class definitions

See "About the prelim" on course website
The frame (the box) for a method call

Remember: Every method is in a folder (object) or in a file-drawer.

- Method name: instruction counter
- Scope box
- Parameters
- Local variables (don’t deal with these now)
- Draw the parameters as variables.
- Number of the statement of the method body to execute next. Helps you keep track of what statement to execute next. Start off with 1.
- Scope box contains the name of entity that contains the method—a file-drawer or object.

To execute the call `x.setAmt(50);`

1. Draw a frame for the call.
2. Assign the value of the argument to the parameter (in the frame).
3. Execute the method body. (Look for variables in the frame; if not there, look in the place given by the scope box.)
4. Erase the frame for the call.
To execute the call `x.setAmt(10);

1. Draw a frame for the call.
2. Assign the value of the argument to the parameter (in the frame).
3. Execute the method body. [look for variables in the frame; if not there, look in the place given by the scope box.]
4. Erase the frame for the call. But not in the exam! Leave it there for us to see!

```
Account
```

```
setAmt(int newAmt) { amt = newAmt; }
getAmt() {...}
```

To execute the call
```
x.setAmt(50);
```

1. Draw a frame for the call.
2. Assign the value of the argument to the parameter (in the frame).
3. Execute the method body. (Look for variables in the frame; if not there, look in the place given by the scope box.)
4. Erase the frame for the call.

But not in the exam! Leave it there for us to see!

What’s in the exam?

- Definitions of terms and key concepts
- Execution of assignment statements
- Evaluation / Execution of “new” expressions
- Evaluation of method calls
- Execute sequence of statements
- String functions
- Writing class definitions

See “About the prelim” on course website

(d) 10 pts. Draw a frame for the call `m(2+3, 6)` of the following procedure `m`. We want to see what the frame for the call looks like after the argument values are assigned to the parameters but before the method body is executed.

```
public void m(int x, double y) {
    int z;
    z = x + y;
}
```

```
void Account
```

```
setAmt(int newAmt) { amt = newAmt; }
getAmt() {...}
```

```
x Account
```

Question 5: (20 pts) Write function `fix`, which is specified below. You may use the following methods (you may not need them all). This might help you: when you break a string up into pieces, store the pieces in local variables and then use these pieces.

```
/** = Date s in a more suitable form.
Precondition: s contains a date in the form month/day/year, with each part separated by "/". Examples are: 4/26/39 and 04/005/1939.
The output should contain the year.month.day. Examples are: 39.26.4 and 1939.04.005.
Each of day, month, and year may be any length. They appear in exactly the same form in the input and output; just their order and the separator are changed. */
```

```
public static String fix(String s) {
    int k = s.indexOf("/");
    // index of first "/" in rest of string is second occurrence of "/".
    String month = s.substring(0, k);
    String rest = s.substring(k + 1);
    k = rest.indexOf("/");
    // index in rest of string of the only "/":
    String day = rest.substring(0, k);
    String year = rest.substring(k + 1);
    return year + "." + month + "." + day;
```

```
/** = Date s in a more suitable form.
Precondition: s contains a date in the form month/day/year, with each part separated by "/". Examples are: 4/26/39 and 04/005/1939.
The output should contain the year.month.day. Examples are: 39.26.4 and 1939.04.005.
Each of day, month, and year may be any length. They appear in exactly the same form in the input and output; just their order and the separator are changed. */
```

```
public static String fix(String s) {
    int k = s.indexOf("/");
    // index of first "/" in rest of string is second occurrence of "/".
    String month = s.substring(0, k);
    String rest = s.substring(k + 1);
    k = rest.indexOf("/");
    // index in rest of string of the only "/":
    String day = rest.substring(0, k);
    String year = rest.substring(k + 1);
    return year + "." + month + "." + day;
```

Question 5: (20 pts) Write function `fix`, which is specified below. You may use the following methods (you may not need them all). This might help you: when you break a string up into pieces, store the pieces in local variables and then use these pieces.

```
/** = Date s in a more suitable form.
Precondition: s contains a date in the form month/day/year, with each part separated by "/". Examples are: 4/26/39 and 04/005/1939.
The output should contain the year.month.day. Examples are: 39.26.4 and 1939.04.005.
Each of day, month, and year may be any length. They appear in exactly the same form in the input and output; just their order and the separator are changed. */
```

```
public static String fix(String s) {
    int k = s.indexOf("/");
    // index of first "/" in rest of string is second occurrence of "/".
    String month = s.substring(0, k);
    String rest = s.substring(k + 1);
    k = rest.indexOf("/");
    // index in rest of string of the only "/":
    String day = rest.substring(0, k);
    String year = rest.substring(k + 1);
    return year + "." + month + "." + day;
```
What’s in the exam?

- Definitions of terms and key concepts
- Execution of assignment statements
- Evaluation / Execution of “new” expressions
- Evaluation of method calls
- Execute sequence of statements
- String functions
- Writing class definitions

See “About the prelim” on course website
http://www.cs.cornell.edu/courses/cs1110/2009fa/exams/prelim1/

Q. 2 (20 points) Write a class definition for a class PercussionInstrument that
- Is a subclass of MusicInstrument;
- Has suitable specifications on methods and definitions on fields;
- Has a field numDrums, which is the number of drums in this percussion instrument;
- Has a constructor with the name of the instrument and the number of drums as parameters;
- Overrides function play() to return the number of “drums”, similar to the way function play in class StringInstrument works.

```java
/** An instance represents a percussion instrument */
public class PercussionInstrument extends MusicInstrument {
    // number of drums in this instrument
    private int numDrums;
    public PercussionInstrument(String s) {
        name = s;
    }
    public PercussionInstrument() {
        name = "";
    }
    /** = sound this instrument makes */
    public String play() {
        return "drums " + numDrums + "tris";
    }
    /** = a repr of this instrument */
    public String toString() {
        return "Instrument: " + name;
    }
}
```
3. Has a constructor with the name of the instrument and the number of drums as parameters;

```java
/** An instance represents a percussion instrument. */
public class PercussionInstrument extends MusicInstrument {
    // number of drums in this instrument
    private int numDrums;
    /** Constructor: an instance name s with n drums */
    public PercussionInstrument(String s, int n) {
        super(s);
        numDrums= n;
    }
}
```

4. Overrides function play() to return the number of “druuums”, similar to the way function play in class StringInstrument works.

```java
/** An instance represents a percussion instrument. */
public class PercussionInstrument extends MusicInstrument {
    // number of drums in this instrument
    private int numDrums;
    /** Constructor: an instance name s with n drums */
    public PercussionInstrument(String s, int n) {
        super(s);
        numDrums= n;
    }
    public String play() {
        return numDrums = "druuums";
    }
}
```

Purpose of super and this

This refers to the name of the object in which it appears.

```java
/** = String representation of this Employee. */
public String toString() { return this.getName() + ", year " +
getStartDate() + ", salary " + salary; }
ok, but unnecessary
/** = toString value from superclass */
public String toString() { return super.toString(); }
```
/** An instance represents a string instrument with no name */

```java
public class StringInstrument extends MusicInstrument {
    // number of strings on this instrument
    private int numStrings;

    public StringInstrument() {
        name = "";
    }

    public String play() {
        return super.toString() + numStrings + "triiings";
    }
}
```

5. Overrides function `play()` to return the number of “druuums”, similar to the way function play in class `StringInstrument` works.

```java
public class PercussionInstrument extends MusicInstrument {
    // number of drums in this instrument
    private int numDrums;

    public PercussionInstrument(String name, int n) {
        super(name);
        numDrums = n;
    }

    public String play() {
        return super.toString() + numDrums + "druuums";
    }
}
```

6. Has suitable specifications on methods and definitions on fields

```java
/** An instance represents a percussion instrument */
public class PercussionInstrument extends MusicInstrument {
    // number of drums in this instrument
    private int numDrums;

    public PercussionInstrument(String name, int n) {
        super(name);
        numDrums = n;
    }

    public String play() {
        return super.toString() + numDrums + "druuums";
    }
}
```

What’s in the exam?

- Definitions of terms and key concepts
- Execution of assignment statements
- Evaluation / Execution of “new” expressions
- Evaluation of method calls
- Execute sequence of statements
- String functions
- Writing class definitions
- See “About the prelim” on course website

Good Luck!