class Wombat {
    int age;
    String name;
    Wombat(String nm) {
        age = 1;
        name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay();
        bob.showBDay();
        sue.showBDay();
    }
}
```java
class Wombat {
    int age;
    String name;

    Wombat(String nm) {
        age = 1;
        name = nm;
    }

    void showBDay() {
        age = age + 1;
        out.println(name + " is now " + age);
    }

    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay();
        bob.showBDay();
        sue.showBDay();
    }
}
```
```java
class Wombat {
   int age;
   String name;

   Wombat(String nm) {
      age = 1;
      name = nm;
   }

   void showBDay() {
      age = age + 1;
      out.println(name + " is now " + age);
   }

   public static void main(String[] a) {
      Wombat sue = new Wombat("jane");

      sue.showBDay();
      sue.showBDay();
   }
}
```

Really means:

- `int id# of Wombat`
- `Creates a Wombat Object`
- `Calls procedure named Wombat !!!`

More Notes

Thursday, April 05, 2012
12:04 PM
OO means "Object Oriented"

The style of paper check will be a combination of both kinds of procedure activation chart.

The Local Contexts will be kept in outline form along with calls being indents and returns being outdents.

However, the Object Contexts will be kept in a separate column and referred to by reference numbers - in the same style that was previously used with Local Contexts.

Object Contexts are just called "Objects" for short.
The main differences between Local Contexts and Object Contexts are the following:

(1L) A Local Context is created by entry to its procedure.
(1O) An Object Context is created by the `new` operator.

(2L) A Local Context is destroyed by exit from its procedure activation.
(2O) An Object Context is only destroyed when the program has no way to find it. That is known as "garbage collection".

(3L) A Local Context "BELONGS" to its procedure.
(3O) An Object Context "BELONGS" to the entire program.

(4L) The design of a Local Context comes from the variables declared in its procedure - both the parameter variables and the working variables.
(4O) The design of an Object Context comes from the `member variables` declared in its `CLASS`.

(5L) A Local Context simply has a reference number held by its procedure.
(5O) An Object Context has `both` a reference number - which can be held anywhere in the program or in another Object Context.
(5O+) An Object Context also has a reference to its `CLASS`. 
Both Object Contexts and Local Contexts have the following features in common:

(1) Both contain named variables - represented by a small table with two columns. The left column names a variable. The right column provides the value for that variable.

(2) Both have a reference number - which by itself is completely sufficient to identify that particular Context.
```java
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay(); bob.showBDay();
        sue.showBDay();
    }
}
```

**Console**

```
jane is now 2  
cathy is now 2  
jane is now 3  
```
```
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay(); bob.showBDay();
        sue.showBDay();
    }
}
```

ASSUMPTION: Object reference numbers are 1000,2000,3000,... in order of creation.

Program creates new object of class Wombat - its reference number is 1000.
Its variables are not yet initialized.
class Wombat {
    int age;  String name;
    Wombat(String nm) {  
        this = 1000;  nm = "jane";
        age = 1;  name = nm;
    }
    void showBDay() {  
        age = age + 1;
        out.println(name+" is now "+age);
    }  
    public static void main(String[] a) {  
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay();  bob.showBDay();
        sue.showBDay();
    }  
}

main(---) {
    1000.Wombat("jane")
    this = 1000
    nm = "jane"

main calls constructor procedure named Wombat with owner 1000 and parameter value "jane".

The variable this always refers to the owner.
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name + " is now " + age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay();  bob.showBDay();
        sue.showBDay();
    }
}

Procedure Wombat executes two assignment statements.
One copies constant 1 into Object #1000's variable named age.
The second copies local variable nm's value into Object #1000's variable named name.
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay(); bob.showBDay();
        sue.showBDay();
    }
}

main(...) {
    1000.Wombat("jane")
    this = 1000
    nm = "jane"
class Wombat {
    int age; String name;
    Wombat(String nm) {
        age = 1; name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay(); bob.showBDay();
    }
}
Notice that using the operator `new` requires three steps. The command looks like this:

```java
Wombat sue = new Wombat("jane");
```

But the word `Wombat` has three related but different meanings:

First the `new` operation creates a new `Wombat` - where "Wombat" means the `class Wombat`.

```java
Wombat sue = new Wombat("jane");
```

Second, the `Wombat` constructor procedure is called - where "Wombat" means `constructor procedure`.

```java
Wombat sue = new Wombat("jane");
```

Finally, the `Wombat` variable `sue` is assigned an integer value - the reference number of the new `Wombat` - where "Wombat" means `reference number`.

```java
Wombat sue = new Wombat("jane");
```
The three steps of `new` are SO IMPORTANT - you need to MEMORIZE them.

Here they are again in summary:

<table>
<thead>
<tr>
<th>Step</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create new Wombat</td>
<td><code>Wombat sue = new Wombat(&quot;jane&quot;);</code></td>
</tr>
<tr>
<td>Call Wombat Constructor</td>
<td><code>Wombat sue = new Wombat(&quot;jane&quot;);</code></td>
</tr>
<tr>
<td>Assign Wombat reference to variable</td>
<td><code>Wombat sue = new Wombat(&quot;jane&quot;);</code></td>
</tr>
</tbody>
</table>
Program creates new object of class Wombat - its reference number is 2000. Its variables are not yet initialized.
```java
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay();  bob.showBDay();
    }
}
```
Wombat
#1000
age 1
name "jane"

Wombat
#2000
age 1
name "cathy"

Console

Procedure **Wombat** executes two assignment statements.
One copies constant 1 into Object #2000's variable named age.
The second copies local variable nm's value into Object #2000's variable named name.
```java
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay();  bob.showBDay();
        sue.showBDay();
    }
}
```

Assign new to bob

Assign value of new operation - namely 2000 - to the local variable bob belonging to main.
The three steps of `new` are SO IMPORTANT - you need to MEMORIZE them.

Here they are again in summary for the second `new` example:

<table>
<thead>
<tr>
<th>Step</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create new Wombat</td>
<td><code>Wombat bob = new Wombat(&quot;cathy&quot;);</code></td>
</tr>
<tr>
<td>Call Wombat Constructor</td>
<td><code>Wombat bob = new Wombat(&quot;cathy&quot;);</code></td>
</tr>
<tr>
<td>Assign Wombat reference to variable</td>
<td><code>Wombat bob = new Wombat(&quot;cathy&quot;);</code></td>
</tr>
</tbody>
</table>
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay(); bob.showBDay();
    }
}
```java
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay(); bob.showBDay();
        sue.showBDay();
    }
}
```
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay();  bob.showBDay();
        sue.showBDay();
    }
}
class Wombat {
    int age; String name;
    Wombat(String nm) {
        age = 1; name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay(); bob.showBDay();
        sue.showBDay();
    }
}
class Wombat {
    int age; String name;
    Wombat(String nm) {
        age = 1; name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name + " is now " + age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay(); bob.showBDay();
        sue.showBDay();
    }
}
```java
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name + " is now " + age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay(); bob.showBDay();
    }
}
```

```
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name + " is now " + age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay(); bob.showBDay();
    }
}
```
```java
class Wombat {
    int age; String name;
    Wombat(String nm) {
        age = 1; name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name + " is now " + age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay(); bob.showBDay();
        sue.showBDay();
    }
}
```
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay();  bob.showBDay();
        sue.showBDay();
    }
}

main(-->

jane is now 2
cathy is now 2
```java
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay(); bob.showBDay();
        sue.showBDay();
    }
}
```

```
main(...) {
    1000.Wombat("jane")
    this = 1000
    nm = "jane"
    sue = 1000
    2000.Wombat("cathy")
    this = 2000
    nm = "cathy"
    bob = 2000
    1000.showBDay()
    this = 1000
    2000.showBDay()
    this = 2000
    1000.showBDay()
    this = 1000
}
```

**Wombat #1000**
- age: 1
- name: "jane"

**Wombat #2000**
- age: 2
- name: "cathy"

**Console**
- jane is now 2
- cathy is now 2
- jane is now 3
```java
class Wombat {
    int age;  String name;
    Wombat(String nm) {
        age = 1;  name = nm;
    }
    void showBDay() {
        age = age + 1;
        out.println(name+" is now "+age);
    }
    public static void main(String[] a) {
        Wombat sue = new Wombat("jane");
        Wombat bob = new Wombat("cathy");
        sue.showBDay(); bob.showBDay();
        sue.showBDay();
    }
}
```

```
main(---) {
    1000.Wombat("jane")
        this = 1000
        nm = "jane"
    sue = 1000
    2000.Wombat("cathy")
        this = 2000
        nm = "cathy"
    bob = 2000
    1000.showBDay()
        this = 1000
    2000.showBDay()
        this = 2000
    1000.showBDay()
        this = 1000
}
```

---

**Console**

- Jane is now 2
- Cathy is now 2
- Jane is now 3