CSC 1051 - Lab 7  
Name:____________

Objectives:
Introduction to creating and using methods in Java.

Before you begin
In order to get an intuition for methods and their use in object oriented programming, you must complete the Alice exercise. Demonstrate your work to get a signature here:

Instructor/TA: ____________________________

Part A: The paintSnowman() method
Using your FancySnowman applet from Lab 5, (or the original Snowman.java from Chapter 2 of the text), you will restructure your code to create and use a method named paintSnowman.

Step 1: Preparation
The FancySnowman class contains a single method definition, for the paint() method. The heading of that method is:

```java
public void paint (Graphics page)
```

The heading is followed by the body of the method, i.e., by the Java statements that specify what the method should do. These statements are enclosed in matching braces {}. The method heading and the method body comprise the method definition.

a) We will be making changes to this file, so start by saving a copy it in a new folder, named Lab07 where all your work for this lab will be kept.

b) Right at the beginning of the method body, there is a comment to specify what the method does. It reads:

```java
//-----------------------------------------------
//  Draws a snowman.
//-----------------------------------------------
```

Change this comment to reflect what you plan to do next:

```java
//-----------------------------------------------
//  Draws a winter scene, including some snowmen.
//-----------------------------------------------
```

Methods should always have a comment to specify what they do.
Step 2: Creating a new method
We will now create another method definition and place it directly following the `paint()` method definition (i.e., after the closing “}” of the `paint()` method, but before the closing brace of the class – in other words, between the two closing braces that appear at the end of the file).

a) Create a heading for a new method, immediately after the `paint()` method definition:

```java
public void paintSnowman(Graphics page, int MID, int TOP)
```

b) Right above your heading, place the comment:

```java
//----------------------------------
// Draws a snowman at MID, TOP.
//----------------------------------
```

c) Based on the heading and comments that you just wrote, answer the following questions:

<table>
<thead>
<tr>
<th>Method name:</th>
<th>______________________________</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method return type:</td>
<td>______________________________</td>
</tr>
<tr>
<td>Required parameters for the method:</td>
<td>______________________________</td>
</tr>
<tr>
<td>Purpose of the method:</td>
<td>______________________________</td>
</tr>
</tbody>
</table>

d) Right below the heading, place the opening and closing braces that will enclose the statements of the method body.

e) Identify the Java code that creates the image of the snowman (head, torso, hat, etc). Note that some of the other code creates the sun, your fence etc – these are NOT part of the snowman. Cut the snowman code out of the `paint()` method and paste it as the body of the `paintSnowman()` method.

e) Verify that everything is properly indented and that you understand the overall structure of the code, which should be something like this:
f) Compile and test your applet. If the program no longer compiles, verify that you have the right structure and all the braces are in the right places. Note that when you run it, it should produce a picture with just the background – NO SNOWMAN!

**Step 3: Invoking the new method**

The reason that the snowman did not appear when you ran the applet is that, although you specified how to paint the snowman, you did not specify that the snowman painted in this scene. We do that by invoking the `paintSnowman()` method as part of the `paint()` method.

a) The `paintSnowman()` method is defined in the same class as the `paint` method, so it can be invoked simply by supplying some values for its parameters. As an example, add the following code to the end of the `paint` method definition:

```java
paintSnowman(page, MID, TOP);
paintSnowman(page, MID-40, TOP);
paintSnowman(page, MID+100, TOP+20);
```

You should now see not one, but three snowmen!
b) Note that you can use any value for the parameters. Add another snowman by using the paintSnowman method with numeric values:
   
   ```java
   paintSnowman (page, 0, 10);
   ```

c) That probably did not turn out too well (snowman is half out of the scene). Modify the above method invocation so you get a snowman all the way at the left of the applet (but not cut off) and whose hat touches the top of the applet. Write here the parameter values you used:

   ```java
   paintSnowman ( , , );
   ```

d) Sketch a picture of what your applet looks like now:
Part B: Adding more methods
Suppose you would like to add some more methods. For each of these proposed
methods, fill in the missing information, as show in this example.

EXAMPLE:

Method heading: 
public void paintSnowman(Graphics page, int MID, int TOP)

Method name: paintSnowman

Method return type: void

Required parameters for the method: 3 parameters: a Graphics object and
two integers

Purpose of the method (in your own words): 
Given a Graphics object page and two int parameters MID and TOP, 
paints an image of a snowman on page centered horizontally at position 
MID, with the top of the snowman’s head at position TOP.

1) Maybe it is springtime and we need some cloverleafs...

Method heading: 
public void paintClover(Graphics page, int x, int y, int size)

Method name: 

Method return type: 

Required parameters for the method: 

Purpose of the method: 
Given a Graphics object page and three int parameters x, y, and 
size, paints an image of a cloverleaf on page rooted at position x, y 
and of height determined by size.
2) A method for painting stars in random positions:

Method heading:
________________________________________________
Method name:_______________________________________
Method return type:_________________________________
Required parameters for the method: _______________________

Purpose of the method (in your own words):
Given a Graphics object page and an int parameter num, paints num stars in random positions along the top 300x200 portion of page.

3) A variation of the previous one, where you get to choose the dimensions of the “sky” (note: you can use the same method name, but need additional parameters):

Method heading:
________________________________________________
Method name:_______________________________________
Method return type:_________________________________
Required parameters for the method: _______________________

Purpose of the method:
   Given a Graphics object page and three int parameters num, width, height, paints num stars in random positions along the top width x height portion of page.
4) Do something similar for a method that paints the fence. You choose what kinds of parameters are needed here and write the description of the purpose accordingly.

<table>
<thead>
<tr>
<th>Method heading:</th>
</tr>
</thead>
<tbody>
<tr>
<td>_________________</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method name:</th>
</tr>
</thead>
<tbody>
<tr>
<td>_______________</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method return type:</th>
</tr>
</thead>
<tbody>
<tr>
<td>______________________</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Required parameters for the method:</th>
</tr>
</thead>
<tbody>
<tr>
<td>_________________________________</td>
</tr>
<tr>
<td>_________________________________</td>
</tr>
<tr>
<td>_________________________________</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Purpose of the method:</th>
</tr>
</thead>
<tbody>
<tr>
<td>______________________</td>
</tr>
</tbody>
</table>

---

**Part C: Implementing the methods from part B**

Implement at least one of the methods from Part B. Be sure to:
1) Include appropriate comments with the method definition  
2) Invoke the method more than once from the paint() method  
3) Test it to make sure it does what you want

Write down the names of the methods you implemented and demonstrate them in class:

<table>
<thead>
<tr>
<th>method name:</th>
<th>checked:</th>
</tr>
</thead>
<tbody>
<tr>
<td>_______________</td>
<td>______________</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>method name:</th>
<th>checked:</th>
</tr>
</thead>
<tbody>
<tr>
<td>_______________</td>
<td>______________</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>method name:</th>
<th>checked:</th>
</tr>
</thead>
<tbody>
<tr>
<td>_______________</td>
<td>______________</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>method name:</th>
<th>checked:</th>
</tr>
</thead>
<tbody>
<tr>
<td>_______________</td>
<td>______________</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>method name:</th>
<th>checked:</th>
</tr>
</thead>
<tbody>
<tr>
<td>_______________</td>
<td>______________</td>
</tr>
</tbody>
</table>