

For the questions on this review sheet, you should feel free to refer to the listing of methods in the Actor and World classes at the end of the review sheet and use them if needed to answer the questions. The same tables will be provided on the exam.

```

public class Foo {
    private int _num;
    private Bar _foo;

    public Foo() {
        _num = 0;
    }

    public void fooBar() {
        _foo = new Bar();
        moveForward(25);
    }
}
    
```

Handwritten annotations:

- Red boxes around `_num;` and `_foo;` with arrows pointing to 'C' and 'D (either or)'.
- Red box around `public Foo() {` with arrow pointing to 'A'.
- Red box around `_num = 0;` with arrow pointing to 'E'.
- Red box around `public void fooBar() {` with arrow pointing to 'E'.
- Red box around `_foo = new Bar();` with arrow pointing to 'B'.
- Red box around `moveForward(25);` with arrow pointing to 'E'.
- Green box around the entire `fooBar()` method body with arrow pointing to 'method definition'.
- Red arrow pointing to the `fooBar()` header with arrow pointing to 'header + body'.
- Green arrow pointing to `moveForward(25);` with arrow pointing to 'method call'.

1. Use the class definition above to circle and identify the parts of code from the list given.

- a) Constructor definition
- b) Code that creates an object
- c) Instance variable name
- d) Instance variable declaration
- e) Assignment statement
- f) Method call
- g) Method definition

2. What is the purpose of the constructor in code? When is a constructor called/executed?

Handwritten answer:

To set up the initial state of an object
 Whenever we create an object

For questions 3 - 7, fill assume that the method calls will go in the space indicated in the code sample given.

```
public class Forrest extends World {  
    public Forrest() {  
        //Code for Questions 3-7 would be written here  
    }  
}
```

3. Write the method call to add a Leaf to the world at location (45, 36).

```
addObject(new Leaf(), 45, 36);
```

4. Write the method call to add a Leaf to the world at a random location.

```
addObject(new Leaf(), Greenfoot.getRandomNumber(getWidth()), Greenfoot.getRandomNumber(getHeight()));
```

5. Write the method call to add a Leaf to the world at the lower right hand corner of the world.

```
addObject(new Leaf(), getWidth(), getHeight());
```

6. Write the code that adds 5 Leaf objects to the world at random locations (using a loop).

```
int count = 0;  
while(count < 5) {  
    addObject(new Leaf(), Greenfoot.getRandomNumber(getWidth()), Greenfoot.getRandomNumber(getHeight()));  
    count = count + 1;  
}
```

The following questions are not related to the above example.

7. Write the code to create a Boy object.

```
new Boy();
```

8. Write the code to declare an instance variable of type Boy named `_boy`.

```
private Boy _boy;
```

9. Write the code that assigns the value 45 to a variable named `temp`.

```
temp = 45;
```

10. Write the code that declares a variable whose type is an array that holds integers. The name of the variable should be `nums`.

```
int[] nums;
```

11. Write the code that declares the array `nums` and inserts the even numbers from 0 to 14 into it.

```
int[] nums = {0,2,4,6,8,10,12,14};
```

Other topics to study:

- if/else statements
- boolean logical operators (and, or, not)
- Loops
- How we made multiple keys for the piano out of one class definition (abstraction)

Actor Method Summary

void	act () The act method is called by the greenfoot framework to give objects a chance to perform some action.
protected void	addedToWorld (World world) This method is called by the Greenfoot system when the object has been inserted into the world.
GreenfootImage	getImage () Returns the image used to represent this Actor.
protected	getIntersectingObjects (java.lang.Class cls)

java.util.List	Return all the objects that intersect this object.
protected java.util.List	<u>getNeighbours</u> (int distance, boolean diagonal, java.lang.Class cls) Return the neighbours to this object within a given distance.
protected java.util.List	<u>getObjectsAtOffset</u> (int dx, int dy, java.lang.Class cls) Return all objects that intersect the given location (relative to this object's location).
protected java.util.List	<u>getObjectsInRange</u> (int radius, java.lang.Class cls) Return all objects within range 'radius' around this object.
protected <u>Actor</u>	<u>getOneIntersectingObject</u> (java.lang.Class cls) Return an object that intersects this object.
protected <u>Actor</u>	<u>getOneObjectAtOffset</u> (int dx, int dy, java.lang.Class cls) Return one object that is located at the specified cell (relative to this objects location).
int	<u>getRotation</u> () Return the current rotation of the object.
<u>World</u>	<u>getWorld</u> () Return the world that this object lives in.
int	<u>getX</u> () Return the x-coordinate of the object's current location.
int	<u>getY</u> () Return the y-coordinate of the object's current location.
protected boolean	<u>intersects</u> (<u>Actor</u> other) Check whether this object intersects with another given object.
void	<u>setImage</u> (<u>GreenfootImage</u> image) Set the image for this object to the specified image.
void	<u>setImage</u> (java.lang.String filename) Set an image for this object from an image file.
void	<u>setLocation</u> (int x, int y) Assign a new location for this object.
void	<u>setRotation</u> (int rotation) Set the rotation of the object.

World Method Summary

void	<u>act</u> () Act method for world.
void	<u>addObject</u> (<u>Actor</u> object, int x, int y) Add an Actor to the world.
<u>GreenfootImage</u>	<u>getBackground</u> () Return the world's background image.

int	<u>getCellSize</u> () Return the size of a cell (in pixels).
java.awt.Color	<u>getColorAt</u> (int x, int y) Return the color at the centre of the cell.
int	<u>getHeight</u> () Return the height of the world (in number of cells).
java.util.List	<u>getObjects</u> (java.lang.Class cls) Get all the objects in the world, or all the objects of a particular class.
java.util.List	<u>getObjectsAt</u> (int x, int y, java.lang.Class cls) Return all objects at a given cell.
int	<u>getWidth</u> () Return the width of the world (in number of cells).
int	<u>numberOfObjects</u> () Get the number of actors currently in the world.
void	<u>removeObject</u> (<u>Actor</u> object) Remove an object from the world.
void	<u>removeObjects</u> (java.util.Collection objects) Remove a list of objects from the world.
void	<u>repaint</u> () Repaints the world.
void	<u>setActOrder</u> (java.lang.Class... classes) Set the act order of objects in the world.
void	<u>setBackground</u> (<u>GreenfootImage</u> image) Set a background image for the world.
void	<u>setBackground</u> (java.lang.String filename) Set a background image for the world from an image file.
void	<u>setPaintOrder</u> (java.lang.Class... classes) Set the paint order of objects in the world.
void	<u>started</u> () This method is called by the Greenfoot system when the execution has started.
void	<u>stopped</u> () This method is called by the Greenfoot system when the execution has stopped.