

# Python & Turtle Graphics

## Tutorial

### GGC S3



# Programming Skills Objectives:

- Identify and program Python code in JES (Jython Environment for Students) and save into their file area with proper extension. (ProgramName.py)
- Use following commands to draw shapes and designs with Turtles.
- Use comments in program to describe what code will do.
- Troubleshoot and solve coding errors and problems.
- Run code from written programs and from program and command area.
- Define and use Python functions.
- Use the "for i in range(x)" commands to do repeats.

# Opening JES Software

- Click JES icon on the desktop to open the JES software.



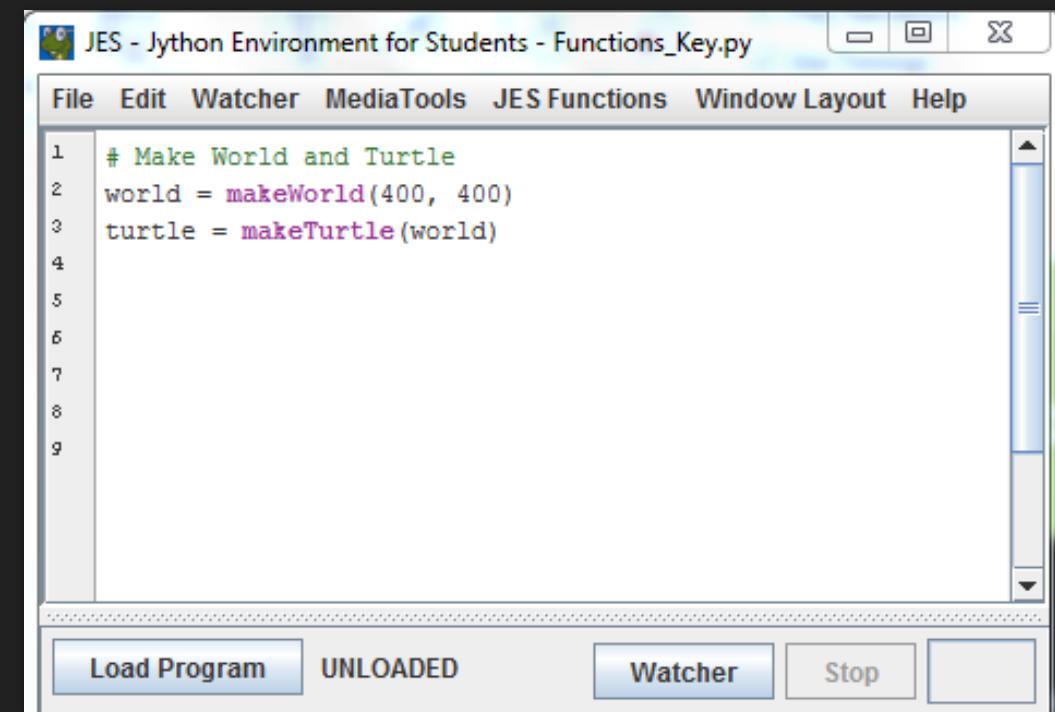
# Tutorial 1 Overview:

## Explore drawing a Square and a Rectangle with the Python Turtle Commands in JES

- In this tutorial, you will practice using the **Command Area** to test and draw a **Square** and a **Rectangle**.
- Use the up arrow key to show previous commands you made in the **Command Area** to make programming faster.
- Make sure you load the World and Turtle program you create **FIRST** before using the Command Area to draw your shapes.
- Modify any Turtle commands you typed in the **Program Area** in order to create the shape you want.
- Turtle Commands used in this Tutorial:
  - `turtle.forward()`
  - `turtle.turn()`
  - `turtle.penUp()`
  - `turtle.penDown()`
  - `turtle.moveTo(x, y)`
- Move on to the next slide to begin typing!

# Create World and Turtle

- First step is to create a World and a Turtle.
- In the **Program Area** of the JES editor, type in the following code to create a world and a turtle:
- `world = makeWorld(400, 400)`
- `turtle = makeTurtle(world)`
- Double check your code with the picture to the right.



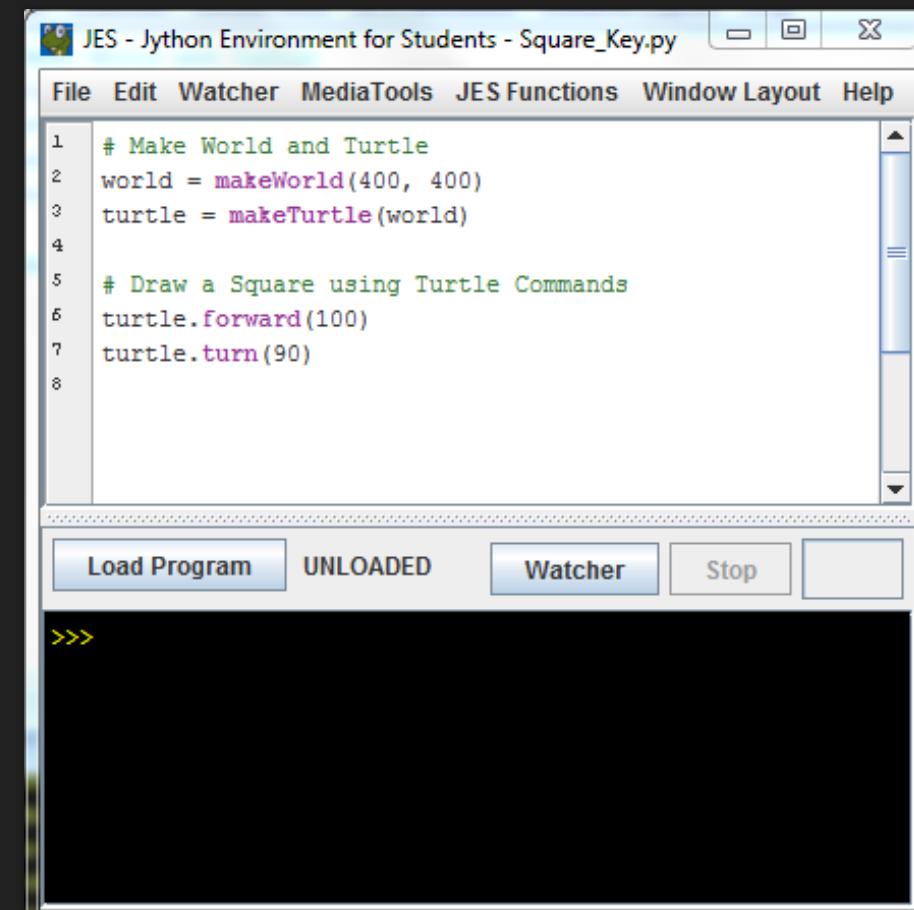
The screenshot shows the JES (Jython Environment for Students) application window. The title bar reads "JES - Jython Environment for Students - Functions\_Key.py". The menu bar includes File, Edit, Watcher, MediaTools, JES Functions, Window Layout, and Help. The main area, labeled "Program Area", contains the following Python code:

```
# Make World and Turtle
world = makeWorld(400, 400)
turtle = makeTurtle(world)
```

Below the code area, there are several buttons: "Load Program" (highlighted in blue), "UNLOADED", "Watcher", "Stop", and two empty input fields.

# Program commands to make the Turtle move

- Type the following commands below your World code to make your turtle move:
- `turtle.forward(100)`
- `turtle.turn(90)`
- Optional: Create a comment for your program.



The screenshot shows the JES (Jython Environment for Students) application window. The title bar reads "JES - Jython Environment for Students - Square\_Key.py". The menu bar includes File, Edit, Watcher, MediaTools, JES Functions, Window Layout, and Help. The main code editor area contains the following Python code:

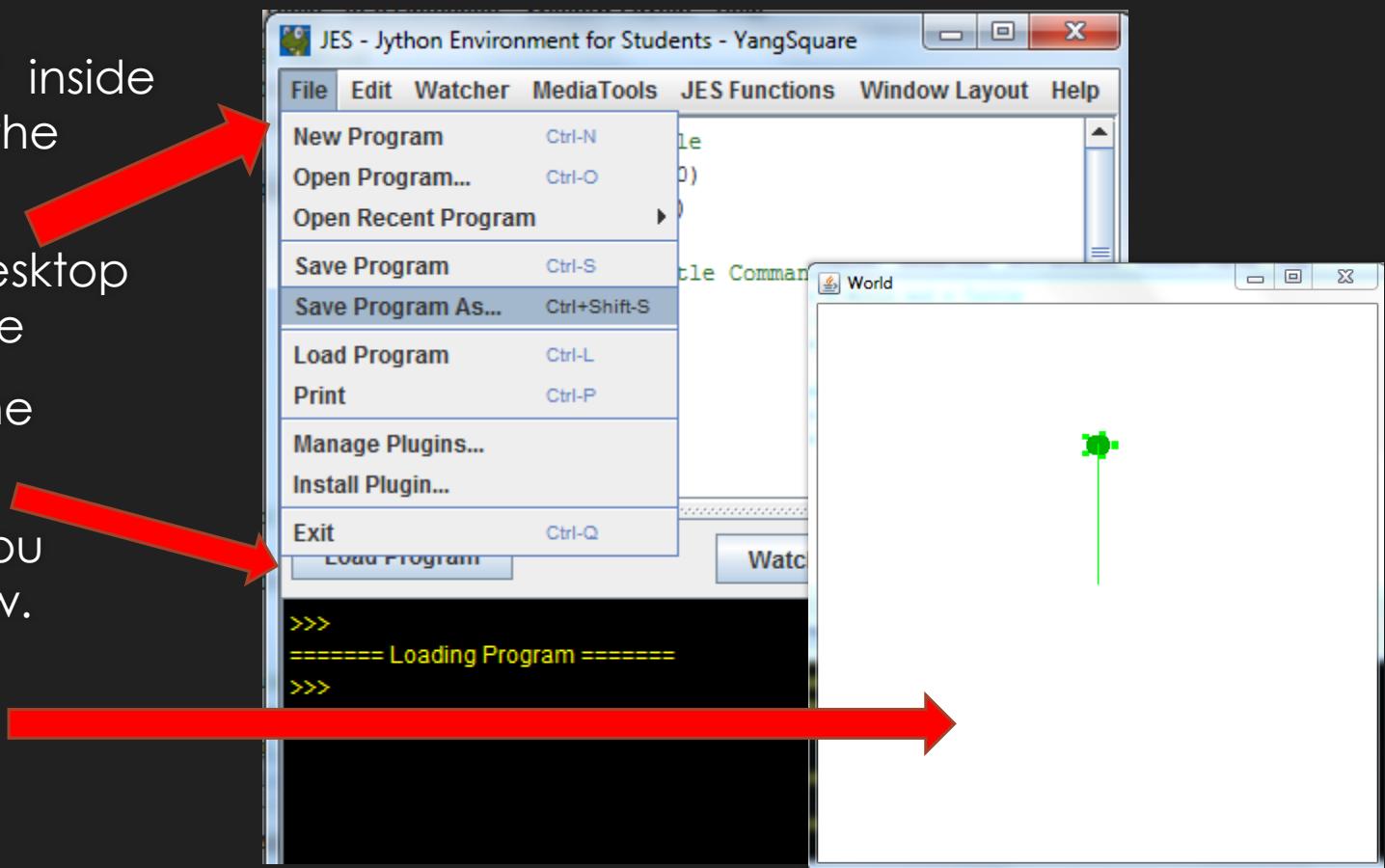
```
# Make World and Turtle
world = makeWorld(400, 400)
turtle = makeTurtle(world)

# Draw a Square using Turtle Commands
turtle.forward(100)
turtle.turn(90)
```

Below the code editor are several buttons: "Load Program" (disabled), "UNLOADED", "Watcher", "Stop", and an empty button. A command prompt window at the bottom shows the text ">>>".

# Save and Load Program

- Save file as “Lastname\_Square.py” inside the Python Turtles folder that is on the desktop.
- Click File → Save Program AS → Desktop → Python & Turtle Graphics → Save
- Click “Load Program” to create the World.
- A new window will open up and you should have a turtle in your window.



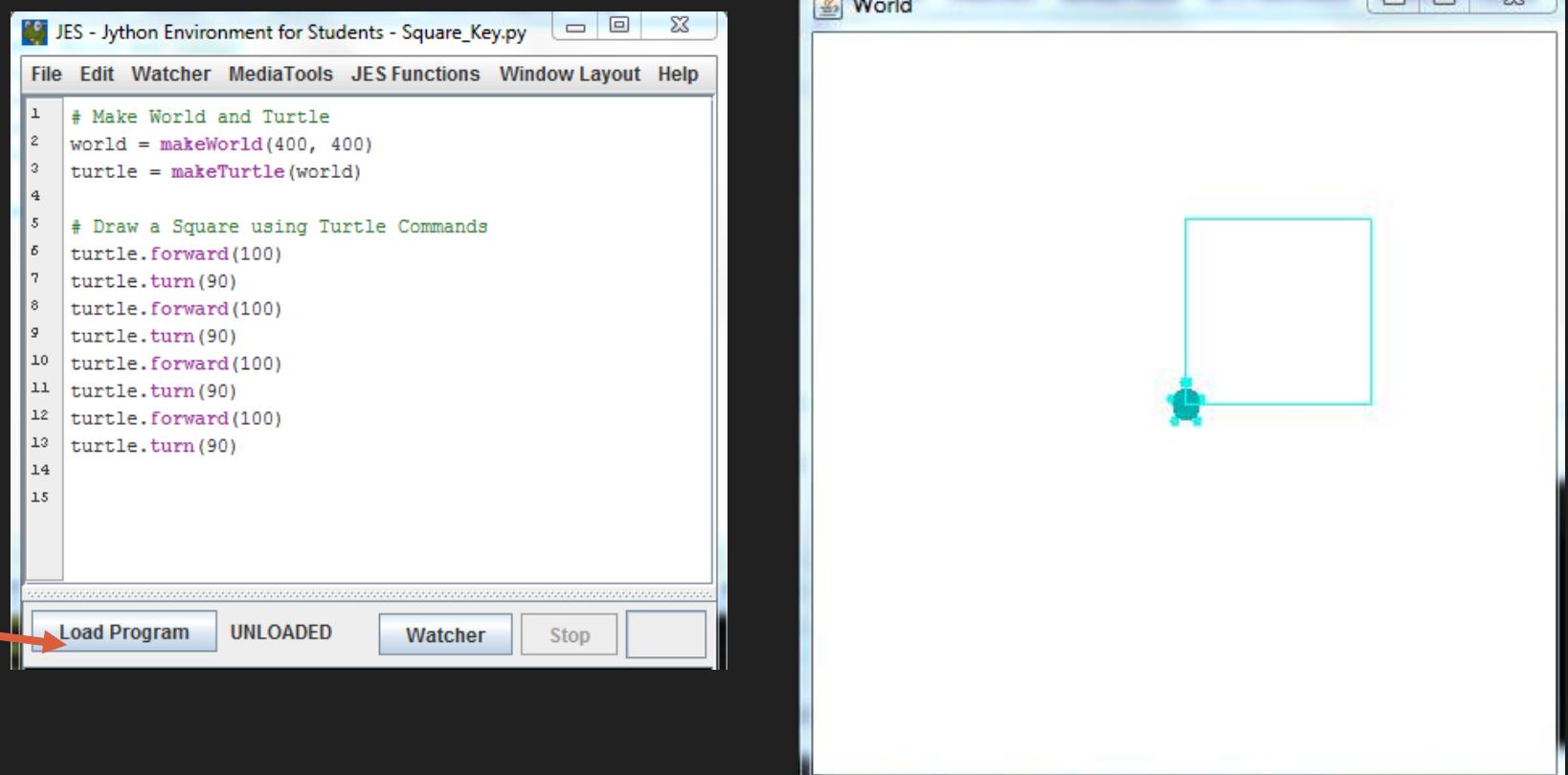
# Using Command Area to test and draw your square

- The JES Command Area allows you to test your codes right away.
- Use the Command Area to test and find a correct path for your turtle to draw your square.
- Only use the following two turtle commands:
  - `turtle.forward(100)`
  - `turtle.turn(90)`
- Having Trouble?: Hit the space bar to reveal the remaining commands your turtle needs to draw a square.



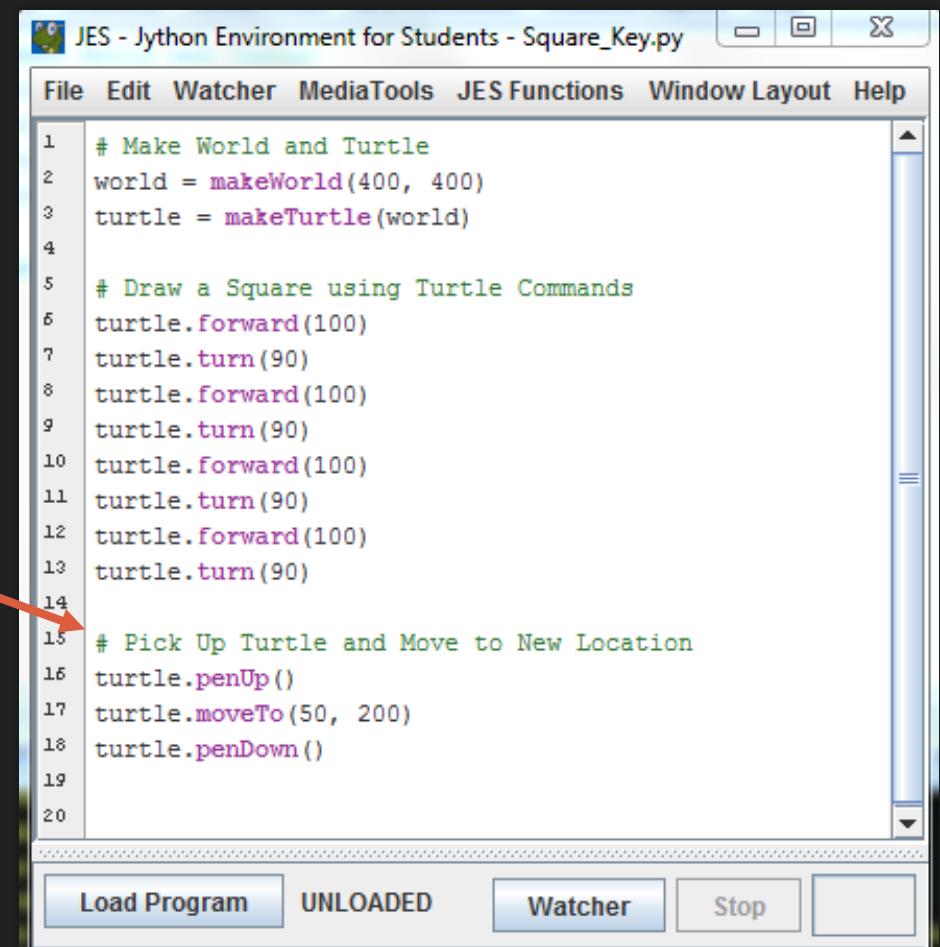
# Turtle Square

- Now that you have figured out the path to create your square, type the correct code into the Program Area.
- Click “Load Program” and you should now have a Square!



# Pick Up and Move Turtle

- Next step is to have the Turtle draw a Rectangle in the same World.
- But before you do that, the turtle needs to be moved to a new location.
- Use the penUp(), moveTo(), and penDown() to accomplish this task.
- Type in the code you see in the image to the right under your Square Turtle code.
- Option: Use Comments to make your program easy to read and for you to understand.



The screenshot shows a window titled "JES - Jython Environment for Students - Square\_Key.py". The code in the editor is as follows:

```
# Make World and Turtle
world = makeWorld(400, 400)
turtle = makeTurtle(world)

# Draw a Square using Turtle Commands
turtle.forward(100)
turtle.turn(90)
turtle.forward(100)
turtle.turn(90)
turtle.forward(100)
turtle.turn(90)
turtle.forward(100)
turtle.turn(90)

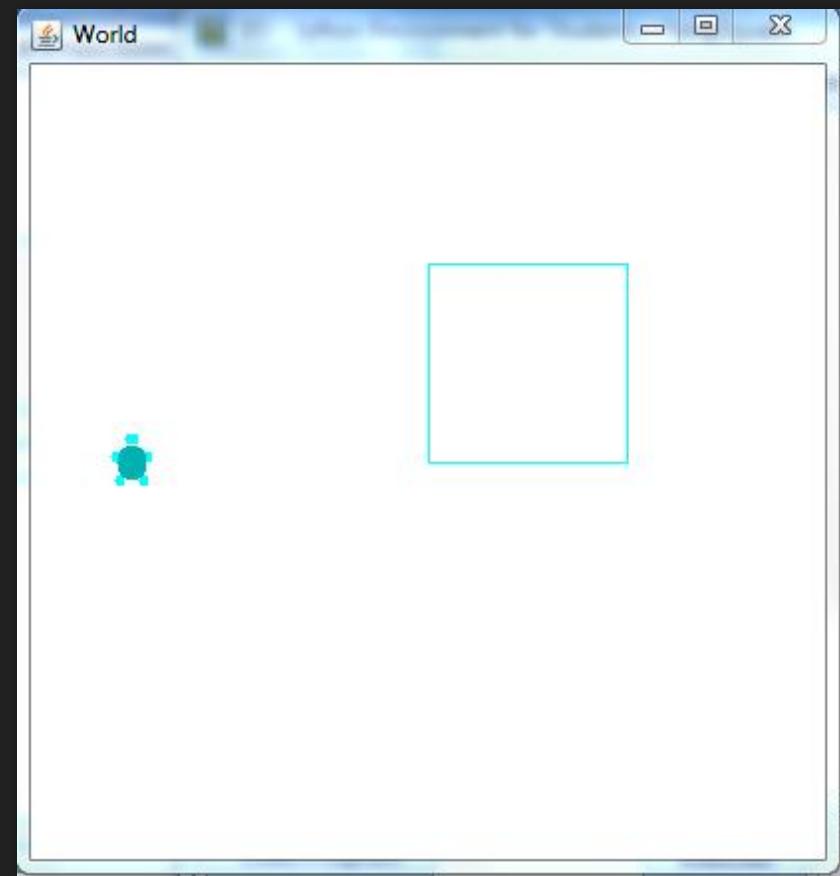
# Pick Up Turtle and Move to New Location
turtle.penUp()
turtle.moveTo(50, 200)
turtle.penDown()

Load Program UNLOADED Watcher Stop
```

A red arrow points from the text "Option: Use Comments to make your program easy to read and for you to understand." in the list above to the line "# Pick Up Turtle and Move to New Location" in the code.

# Pick Up and Move Turtle

- Save and then click “Load Program”.
- Your Turtle should now be in a new location.
- **Note: Multiple JES windows will open when you click “Load Program”. Be sure to close the windows when you are finish with each tutorial. This avoids slowing down the computer.**



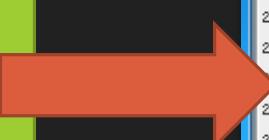
# STOP! Time to Switch Roles

- If you need help, ask for assistance.
- Be sure SWITCH ROLES before moving on to the next tutorial!



# Draw a Rectangle

- Squares and Rectangles are very similar.
- What can we modify or use in the Square Turtle commands to draw a Rectangle?
- Be sure to switch positions and have the other partner draw the Rectangle.
- Use the Command Area to test your code out, then type in the correct code into the Program Area.
- 



JES - Jython Environment for Students - GYSquare.py

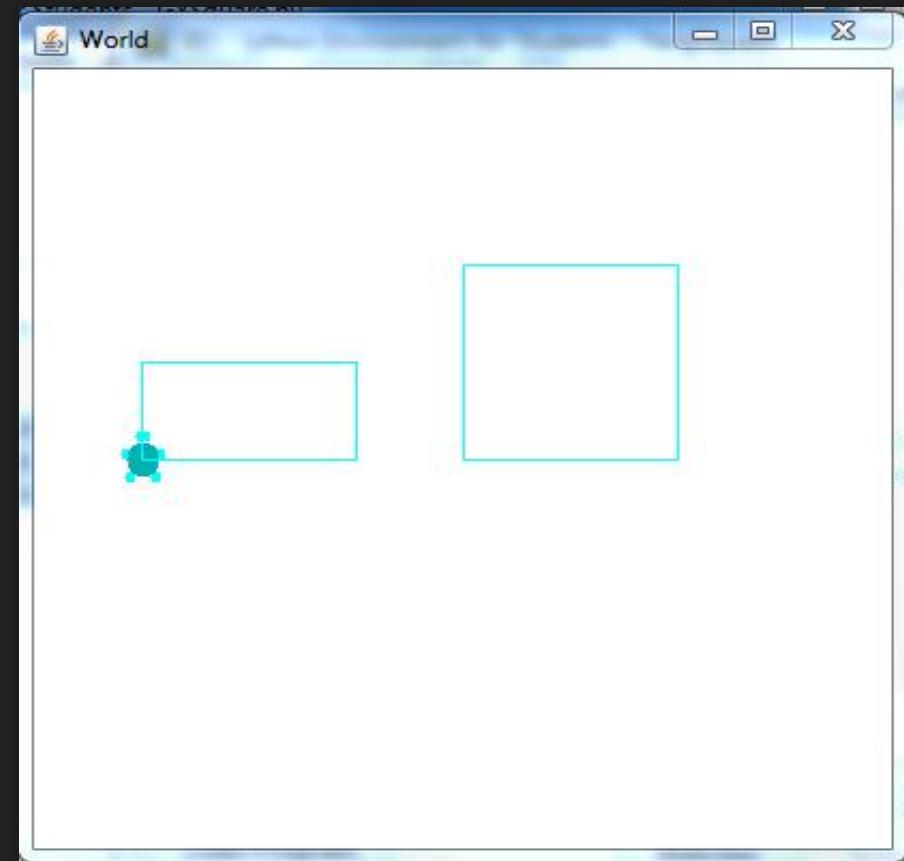
```
File Edit Watcher MediaTools JES Functions Window Layout Help
10 turtle.forward(100)
11 turtle.turn(90)
12 turtle.forward(100)
13 turtle.turn(90)
14
15 # Pick Up Turtle and Move to New Location
16 turtle.penUp()
17 turtle.moveTo(50, 200)
18 turtle.penDown()

20
21
22
23
24
25
26
27
28
29
30
31
32
```

Load Program UNLOADED Watcher

# Draw a Rectangle

- Save and then click “Load Program”.
- You should now have a Rectangle and a Square in your World.

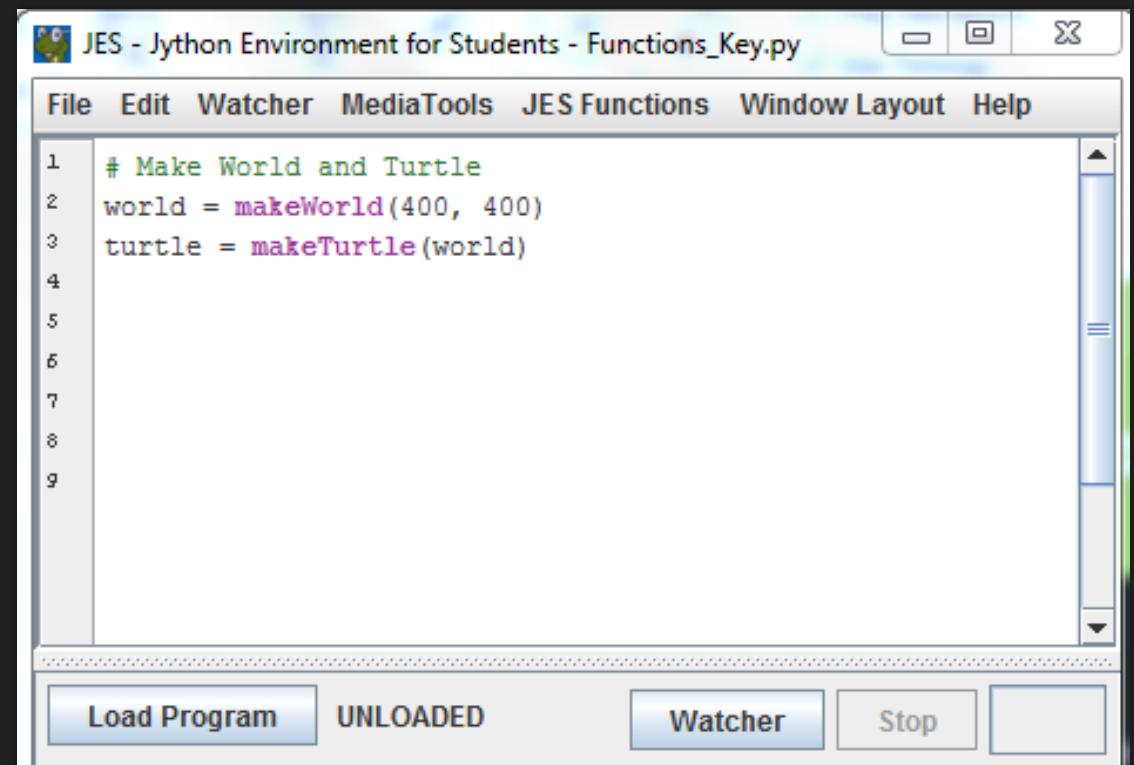


# Tutorial 2: Creating Python Functions for Shapes and Designs in JES

- Remember using the Command Area is good for short experiments.
- Longer programs need to be coded and then saved in the Program Area.
- You can achieve this by placing your Turtle commands in Functions.
- Save your “LastName\_Square.py” program. We will create a new JES program file to use for Tutorial 2 through 4.
- Go to File → New Program → Save Program As → Desktop → Python & Turtle Graphics → Save File Name as “LastName\_Functions.py”.
- **Reminder: Multiple JES windows will open when you click “Load Program”. Be sure to close the windows when you are finish with each tutorial. This avoids slowing down the computer.**

# Create a World and a Turtle

- Begin by creating your World and Turtle In the **Program Area** of the JES editor, type in the following code below:
- `world = makeWorld(400, 400)`
- `turtle = makeTurtle(world)`
- Double check your code with the picture to the right.



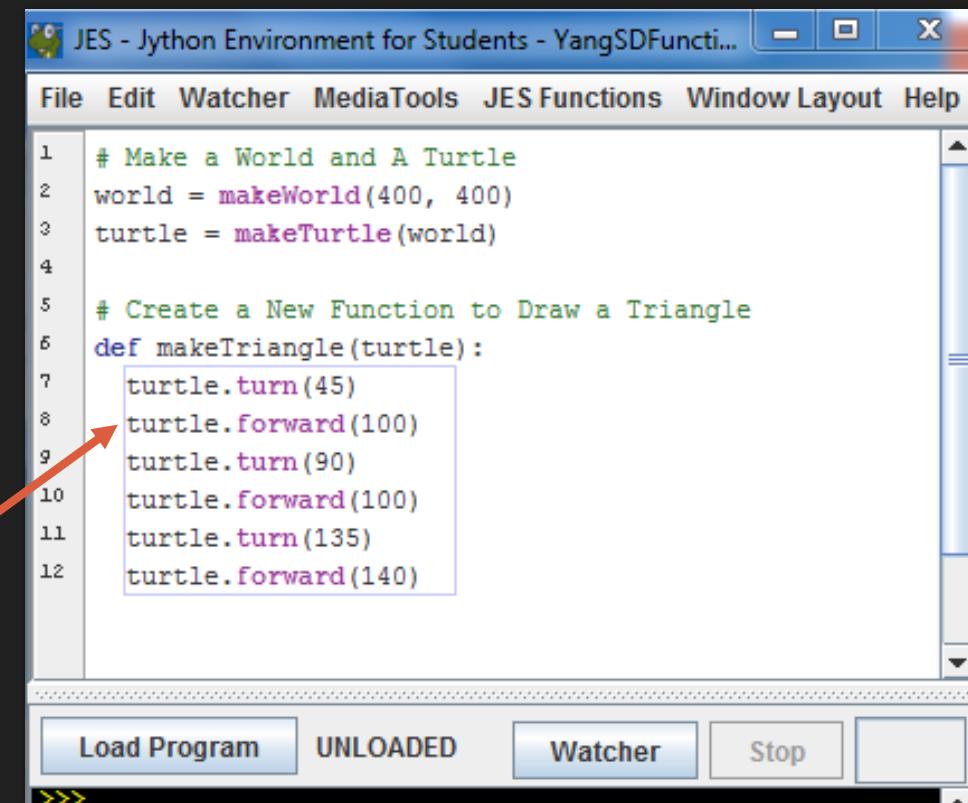
The screenshot shows the JES (Jython Environment for Students) interface. The title bar reads "JES - Jython Environment for Students - Functions\_Key.py". The menu bar includes File, Edit, Watcher, MediaTools, JES Functions, Window Layout, and Help. The main program area contains the following Python code:

```
1 # Make World and Turtle
2 world = makeWorld(400, 400)
3 turtle = makeTurtle(world)
```

Below the code area, there are several buttons: "Load Program" (disabled), "UNLOADED", "Watcher", "Stop", and two empty buttons.

# Create a Function to Draw a Triangle

- Begin the function with **def**
  - **def** keyword means you are creating a new command for your turtle.
- Then, type the name of the function, and the input “turtle” between parentheses:  
**makeTriangle(turtle)**
- End the line with a colon ( “ : ” )
- Next, type in the following turtle commands you see in the image to right to create a Triangle. These commands makes up the *body* of the function.
- The *body* of the function needs to be indented (Hint: Use two spaces).



The screenshot shows the JES (Jython Environment for Students) application window. The menu bar includes File, Edit, Watcher, MediaTools, JES Functions, Window Layout, and Help. The main code editor area contains the following Python script:

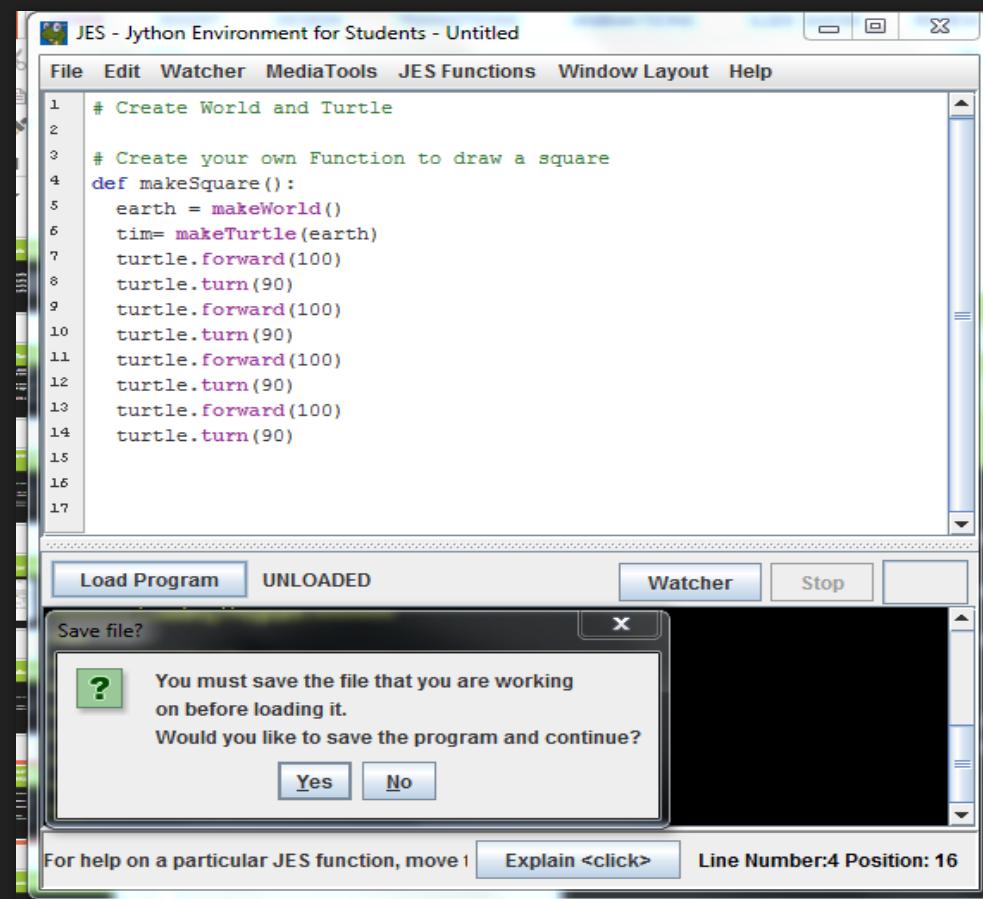
```
# Make a World and A Turtle
world = makeWorld(400, 400)
turtle = makeTurtle(world)

# Create a New Function to Draw a Triangle
def makeTriangle(turtle):
    turtle.turn(45)
    turtle.forward(100)
    turtle.turn(90)
    turtle.forward(100)
    turtle.turn(135)
    turtle.forward(140)
```

A red arrow points from the fifth bullet point in the list on the left to the body of the `makeTriangle` function in the code editor. The status bar at the bottom of the window shows "Load Program", "UNLOADED", "Watcher", "Stop", and a "">>>>" button.

# REMEMBER: The Most Common JES Issue is Forgetting to Load

- Your function does **NOT** exist for JES until you *load* it
  - Before you load it, the program is just a bunch of characters.
  - Loading encodes it as an executable function
- Save and Save As
  - You must Save before Loading
  - You must Load before you can use your function

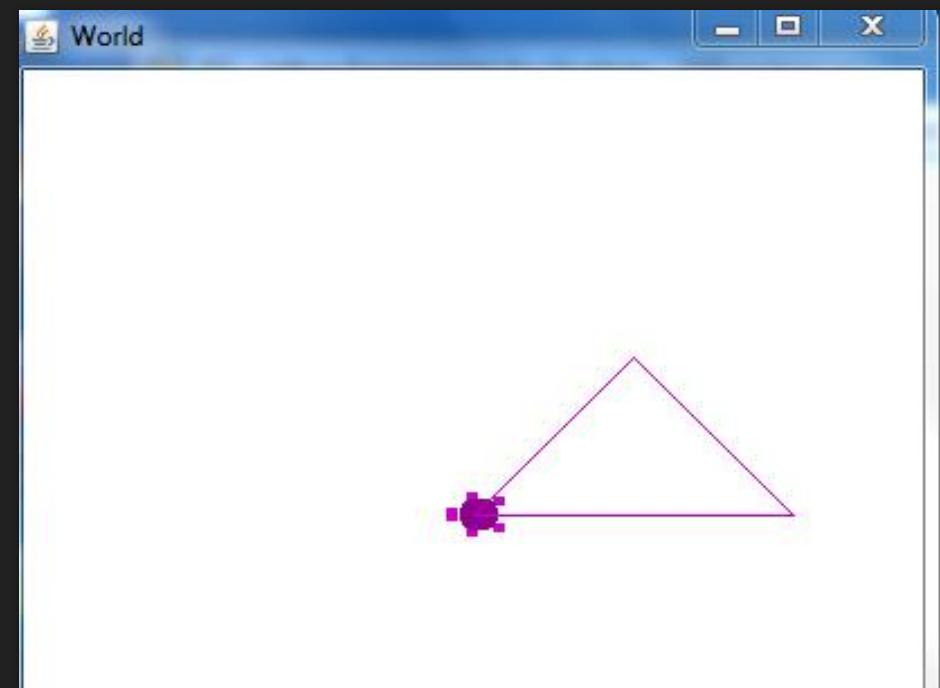


An “Unloaded” function doesn’t exist yet.

# Call Triangle Function

- Save and then click “Load Program”.
- A World window with a Turtle will open.
- Now, call your **makeTriangle(turtle)** function by typing the function name in the Command Area.
- Then hit the enter key.
- This calling will allow the Turtle to draw a Triangle in the World.

```
>>>  
===== Loading Program =====  
>>> makeTriangle(turtle)  
>>>
```



Congrats! You have successfully created a Python Function!

# STOP! Time to Switch Roles

- If you need help, ask for assistance.
- Be sure SWITCH ROLES before moving on to the next tutorial!
- **Reminder: Multiple JES windows will open when you click “Load Program”. Be sure to close the windows when you are finish with each tutorial. This avoids slowing down the computer.**



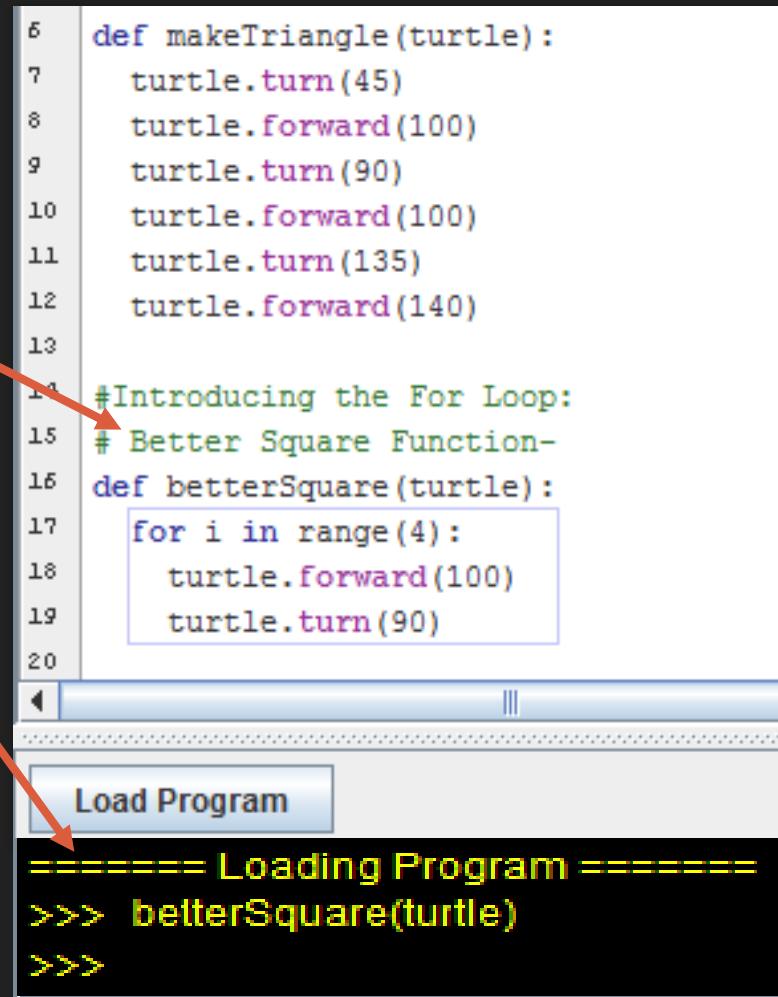
# Tutorial 3: Introducing the For Loop- Using a For Loop in a Function

- Loops are simply parts of a program that run over and over again to reduce the amount of times you type the same command.
- The phrase “**for i in range**” sets the number of times you want to run a command or a series of commands.
- You specify how many times you want to run the turtle commands to repeat inside the parenthesis of the range function.
- TIP: In JES, a For Loop must end with a colon (:) and be indented by two spaces in a Function.**

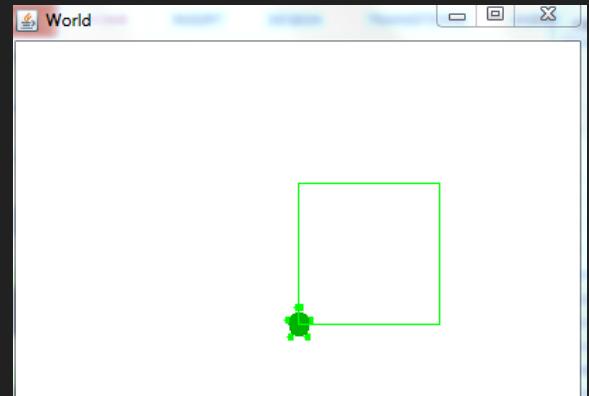
```
def forLoopExample(turtle):  
    for i in range(4):  
        turtle.forward(100)
```

# betterSquare(turtle) Function

- Begin by defining the **betterSquare(turtle)** function under your `makeTriangle(turtle)` function.
- Then type in the for loop and turtle commands you see in the image.
- Be sure to indent the for loop and your turtle commands.
- Next, save and click “**Load Program**”
- Now call your function by typing **betterSquare(turtle)** into the Command Area.
- Be sure to hit the enter key to complete the call of your function.

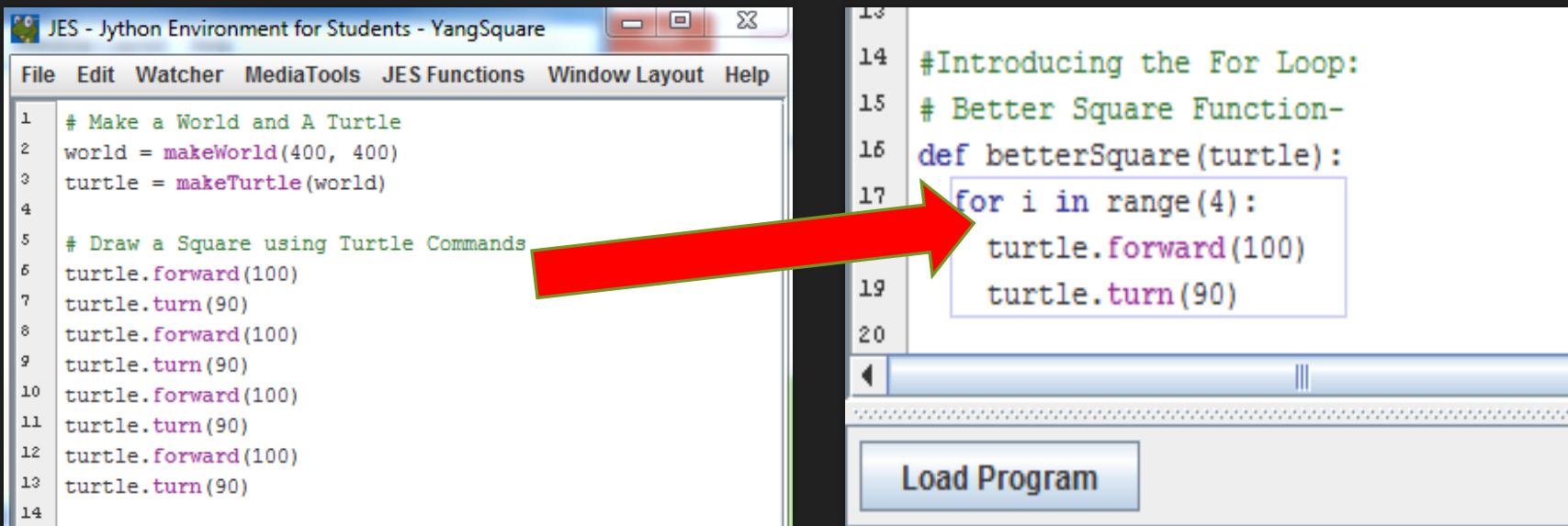


```
5     def makeTriangle(turtle):
6         turtle.turn(45)
7         turtle.forward(100)
8         turtle.turn(90)
9         turtle.forward(100)
10        turtle.turn(135)
11        turtle.forward(140)
12
13
14 #Introducing the For Loop:
15 # Better Square Function-
16 def betterSquare(turtle):
17     for i in range(4):
18         turtle.forward(100)
19         turtle.turn(90)
20
21
22
23 Load Program
===== Loading Program =====
>>> betterSquare(turtle)
>>>
```



# betterSquare(turtle) Function Review

- Remember the two commands to draw a square
  - turtle.forward(100)
  - Turtle.turn(90)
- See how you can use a for loop to minimize the lines of code needed to draw a square.
- Notice how the for loop tells the turtle to make 4 rounds of the forward(100) and turn(90) command to draw the square.



A screenshot of the JES (Jython Environment for Students) software interface. The window title is "JES - Jython Environment for Students - YangSquare". The menu bar includes File, Edit, Watcher, MediaTools, JES Functions, Window Layout, and Help. The code editor contains the following Python code:

```
File Edit Watcher MediaTools JES Functions Window Layout Help
1 # Make a World and A Turtle
2 world = makeWorld(400, 400)
3 turtle = makeTurtle(world)
4
5 # Draw a Square using Turtle Commands
6 turtle.forward(100)
7 turtle.turn(90)
8 turtle.forward(100)
9 turtle.turn(90)
10 turtle.forward(100)
11 turtle.turn(90)
12 turtle.forward(100)
13 turtle.turn(90)
14
15 #Introducing the For Loop:
16 # Better Square Function-
17 def betterSquare(turtle):
18     for i in range(4):
19         turtle.forward(100)
20         turtle.turn(90)
21
22
23
24 Load Program
```

A large red arrow points from the explanatory text above to the for loop definition in the code editor.

# makeCircle(turtle) Function

- Begin by defining the **makeCircle(turtle)** function.
- Then type in the for loop and turtle commands you see in the image.
- Be sure to indent the for loop and your turtle commands.
- Next, save and click “**Load Program**”.
- Now call your function by typing **makeCircle(turtle)** into the **Command Area**.
- This will allow the Turtle to draw a Circle.
- Notice how the for loop tells the Turtle to repeat the commands `forward(20)` and `turn(20)` 18 times to complete the Circle.

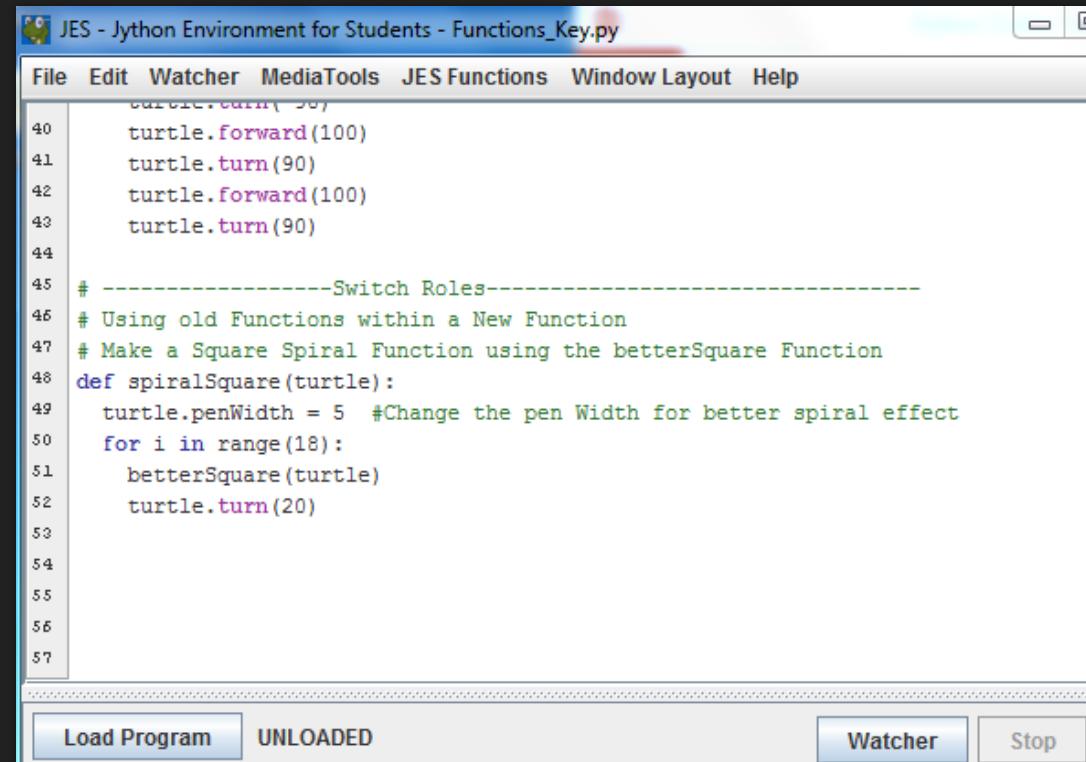
The image shows the JES (Jython Environment for Students) interface. On the left, the code editor window displays Python code for drawing shapes. A red arrow points from the first bullet point in the list to the `makeCircle` function definition. Another red arrow points from the last bullet point to the for loop within the `makeCircle` function. On the right, the 'World' window shows a green circle drawn by the turtle, and the command window at the bottom shows the text "===== Loading Program =====" followed by three green '>>>' prompts.

```
JES - Jython Environment for Students - Functions_Key.py
File Edit Watcher MediaTools JES Functions Window
8     turtle.forward(100)
9     turtle.turn(90)
10    turtle.forward(100)
11    turtle.turn(135)
12    turtle.forward(140)
13
14 #Introducing the For Loop:
15 # Better Square Function-
16 def betterSquare(turtle):
17     for i in range(4):
18         turtle.forward(100)
19         turtle.turn(90)
20
21 # Make a Circle using For Loop
22 def makeCircle(turtle):
23     turtle.penWidth = 3
24     for i in range(18):
25         turtle.forward(20)
26         turtle.turn(20)

=====
>>> makeCircle(turtle)
>>>
```

# Tutorial 4: Using an Old Function within a New Function (Square)

- Create a `spiralSquare(turtle)` Function by using the `betterSquare(turtle)` function.
- Re-using an old shape function within a new function allows your Python program to create cool designs!
- Type in the code you see in the image.
- Save and Load program!
- Be sure to call you **`spiralSquare(turtle)`** function in Command Area!



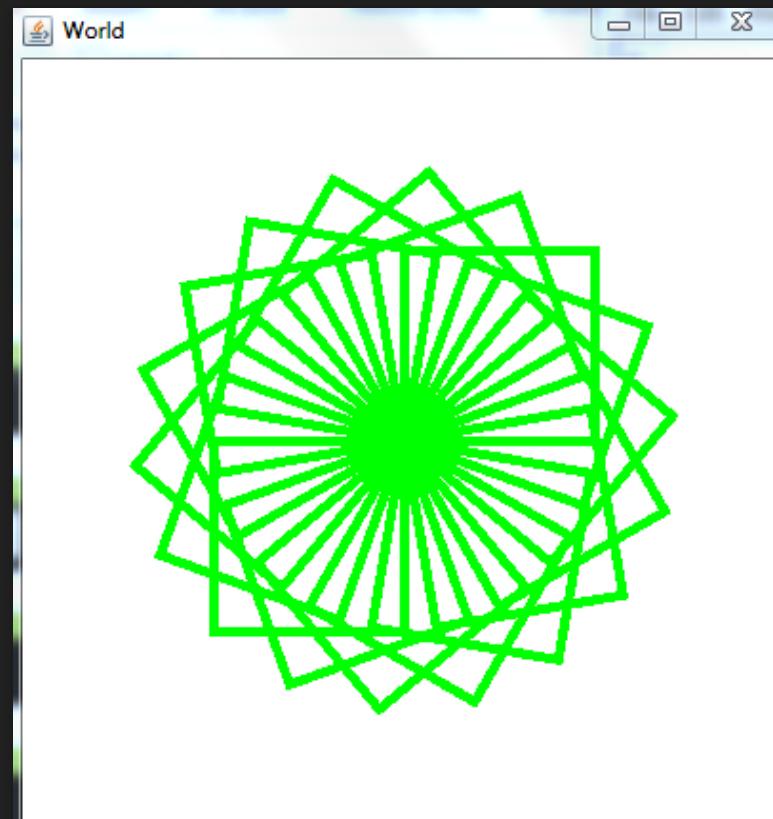
The screenshot shows the JES (Jython Environment for Students) interface. The title bar reads "JES - Jython Environment for Students - Functions\_Key.py". The menu bar includes File, Edit, Watcher, MediaTools, JES Functions, Window Layout, and Help. The main code area contains the following Python code:

```
39     turtle.turn(50)
40     turtle.forward(100)
41     turtle.turn(90)
42     turtle.forward(100)
43     turtle.turn(90)
44
45     # -----Switch Roles-----
46     # Using old Functions within a New Function
47     # Make a Square Spiral Function using the betterSquare Function
48     def spiralSquare(turtle):
49         turtle.penWidth = 5  #Change the pen Width for better spiral effect
50         for i in range(18):
51             betterSquare(turtle)
52             turtle.turn(20)
53
54
55
56
57
```

At the bottom, there are buttons for "Load Program" (which is highlighted), "UNLOADED", "Watcher", and "Stop".

# YOUR RESULTS SHOULD LOOK LIKE THIS!

spiralSquare(turtle)



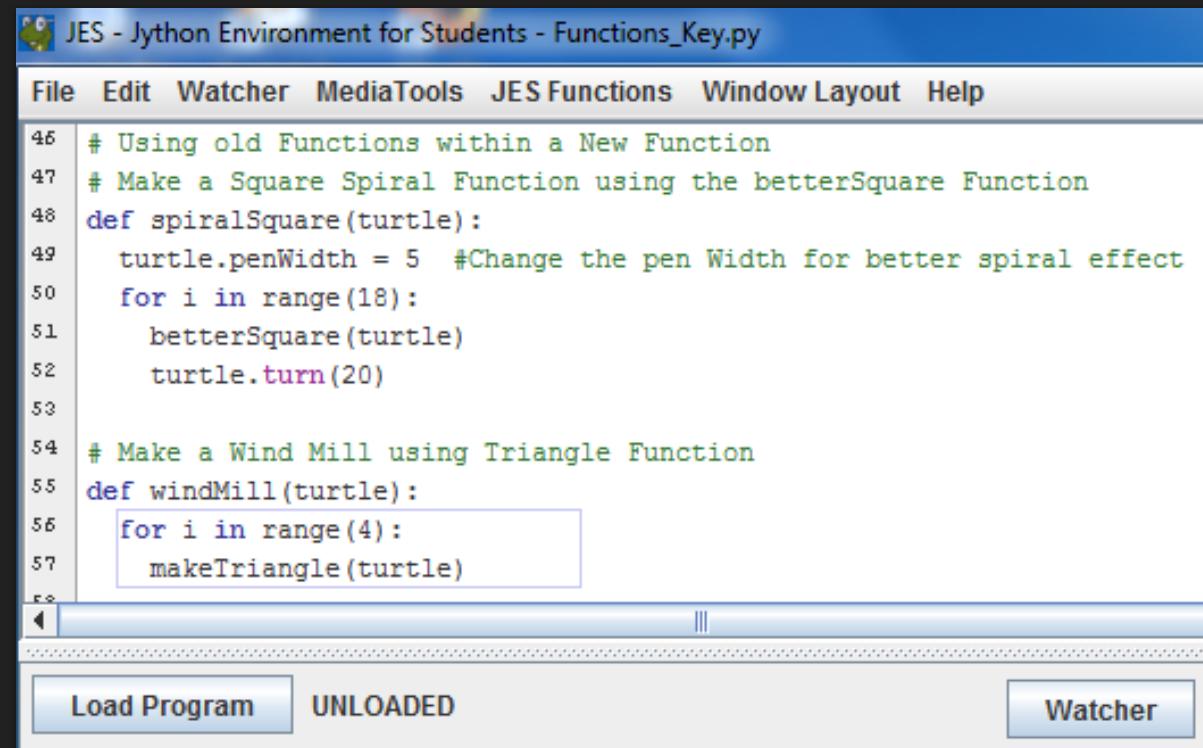
# STOP! Time to Switch Roles

- If you need help, ask for assistance.
- Be sure SWITCH ROLES before moving on to the next tutorial!
- **Reminder: Multiple JES windows will open when you click “Load Program”. Be sure to close the windows when you are finish with each tutorial. This avoids slowing down the computer.**



# Tutorial 4: Using an Old Function within a New Function (Triangle)

- Create a `windMill(turtle)` Function by using the `makeTriangle(turtle)` function.
- Re-using an old shape function within a new function allows your Python program to create cool designs!
- Type in the code you see in the image.
- Save and Load program!
- Be sure to call your **windMill(turtle)** function in Command Area!

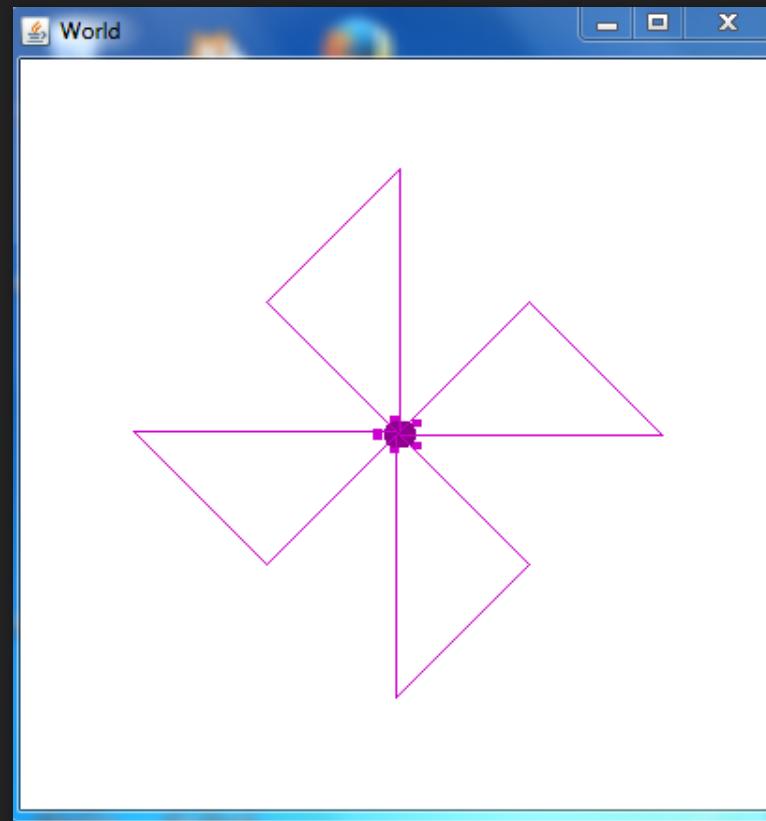


```
JES - Jython Environment for Students - Functions_Key.py
File Edit Watcher MediaTools JES Functions Window Layout Help
46 # Using old Functions within a New Function
47 # Make a Square Spiral Function using the betterSquare Function
48 def spiralSquare(turtle):
49     turtle.penWidth = 5 #Change the pen Width for better spiral effect
50     for i in range(18):
51         betterSquare(turtle)
52         turtle.turn(20)
53
54 # Make a Wind Mill using Triangle Function
55 def windMill(turtle):
56     for i in range(4):
57         makeTriangle(turtle)

Load Program UNLOADED Watcher
```

# YOUR RESULTS SHOULD LOOK LIKE THIS!

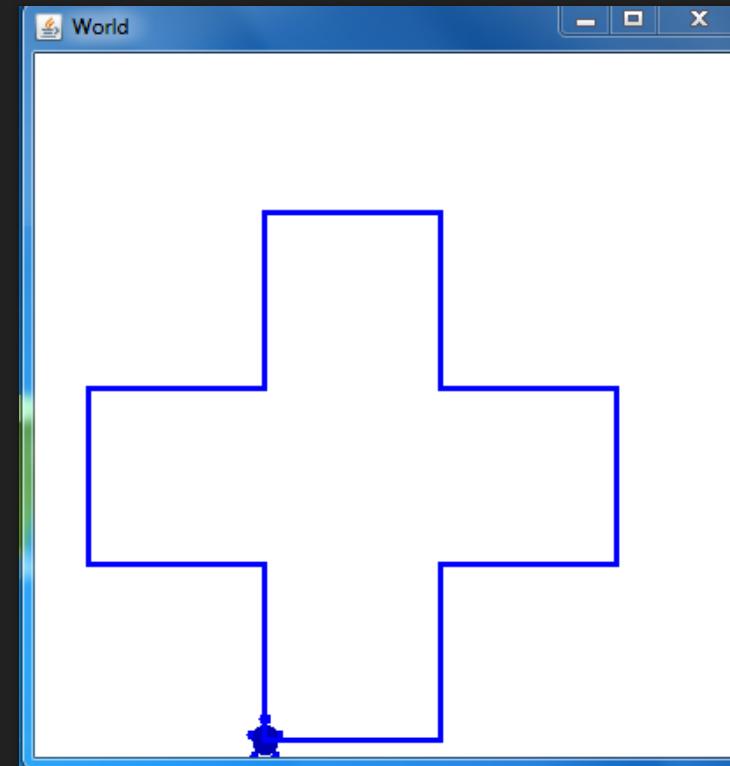
windmill(turtle)



# Tutorial 5: Putting it All Together

## Create a Plus Sign

- Do It Yourself!
- Combine all the Turtle Commands you've learned and create a Plus sign!
- Use Paper and Pencil if needed.
- Raise your hand if you need assistance!
- Make sure you save and "Load Program"
- Then call your function in Command Area.
- Try completing the task, before looking at the Hint.



# Tutorial 5: Putting it All Together

## Create a Plus Sign

Use any of the following turtle commands

- **turtle.forward(distance)**- command to move the turtle in the directions it's facing. Default distance is 100 pixels.
- **turtle.turn(degrees)**- turns the turtle in 90 degrees depending on the direction its facing.
- **turtle.moveTo(x, y)**- command to place the turtle in a new location in the world.
- **turtle.penUp()**- command to pick the pen up.
- **turtle.penDown()**- command to put the pen down again to draw.
- **turtle.penWidth = (1-10)** - allows you to change the size of the pen
- **turtle.color = red (blue, green . . .) -or- (255, 255, 255) RGB notation**- allows you to change the color of the pen and turtle.

# makePlus(turtle) Function Hint Example



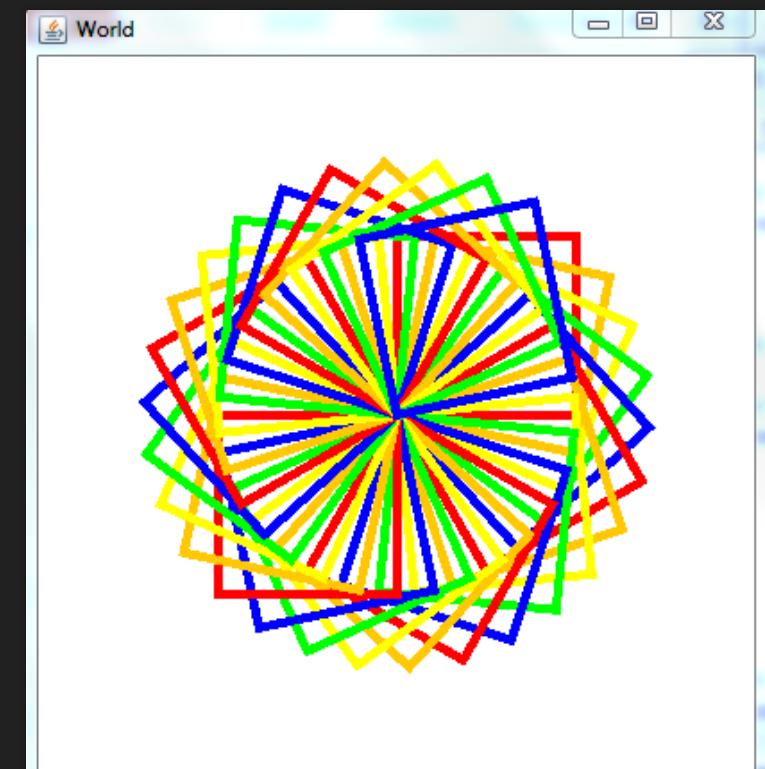
# CONGRATULATIONS!

- You have successfully completed the Python Turtles Tutorial.
- BONUS: Do more with your spiral and windmill designs by adding colors!
- CHALLENGE: Create your initials using the JES and Turtle commands!

# BONUS: Doing More with Turtle Colors!

JES - Jython Environment for Students - Functions\_Key.py

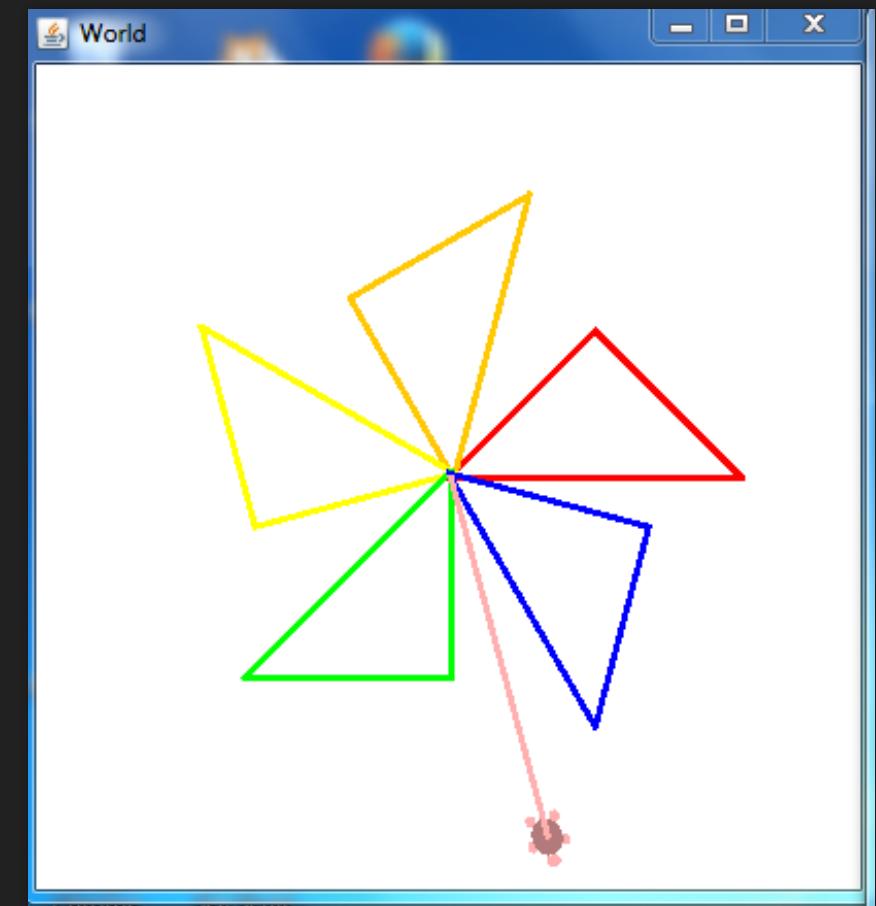
```
File Edit Watcher MediaTools JES Functions Window Layout Help  
58  
59 # Combining it All together! Doing More with Turtle Colors!  
60  
61 # RainbowSpiral  
62 def rainbowSpiral(turtle):  
63     colors = [red, orange, yellow, green, blue]  
64     turtle.penWidth = 5  
65     for i in range(30):  
66         turtle.color = colors[i % 5]  
67         betterSquare(turtle)  
68         turtle.turn(12)  
69
```



# BONUS: Doing More with Turtle Colors!

JES - Jython Environment for Students - Functions\_Key.py

```
File Edit Watcher MediaTools JES Functions Window Layout Help  
53 colors = [red, orange, yellow, green, blue]  
54 turtle.penWidth = 5  
55 for i in range(30):  
56     turtle.color = colors[i % 5]  
57     betterSquare(turtle)  
58     turtle.turn(12)  
59  
60 # Make Rainbow Wind Mill with a Pink Stick  
61 def rainbowWindMill(turtle):  
62     colors = [red, orange, yellow, green, blue]  
63     turtle.penWidth = 3  
64     for i in range(5):  
65         turtle.color = colors[i % 5]  
66         makeTriangle(turtle)  
67         turtle.turn(15)  
68         turtle.color = pink  
69         turtle.turn(180)  
70         turtle.forward(180)
```



# Challenge: Use all the turtle commands you have learned and create your initials!

- **turtle.forward(distance)**- command to move the turtle in the directions it's facing.  
Default distance is 100 pixels.
- **turtle.turn(degrees)**- turns the turtle in 90 degrees depending on the direction its facing.
- **turtle.moveTo(x, y)**- command to place the turtle in a new location in the world.
- **turtle.penUp()**- command to pick the pen up.
- **turtle.penDown()**- command to put the pen down again to draw.
- **turtle.penWidth = (1-10)** - allows you to change the size of the pen
- **turtle.color = red (blue, green . . .) -or- (255, 255, 255) RGB notation**- allows you to change the color of the pen and turtle.

# Survey

- Copy and paste the URL below into your browser to take the survey!
- Thanks, we appreciate your feedback!
- <http://tinyurl.com/ggctech>