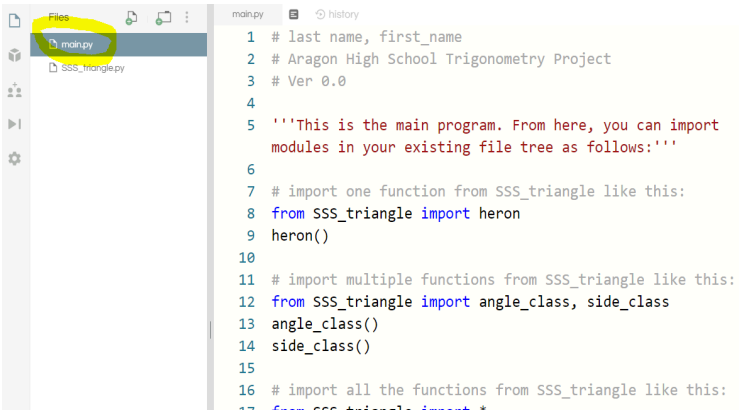
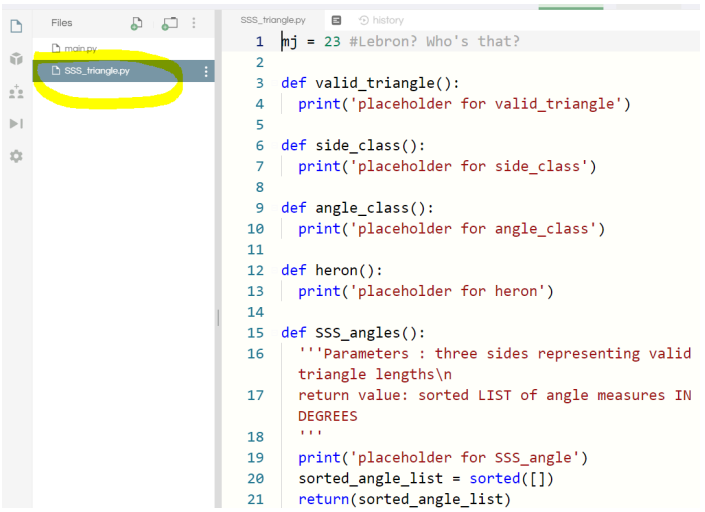
Part I: Creating a `main.py` program to use functions from a created python module.

By this time I've shared **`project 3`** with you on replit.com. When you open it you will find a repl with two files, `main.py and SSS_triangle.py .`

When you click `main.py` , you should see:	When you click `SSS_triangle.py`, :
	

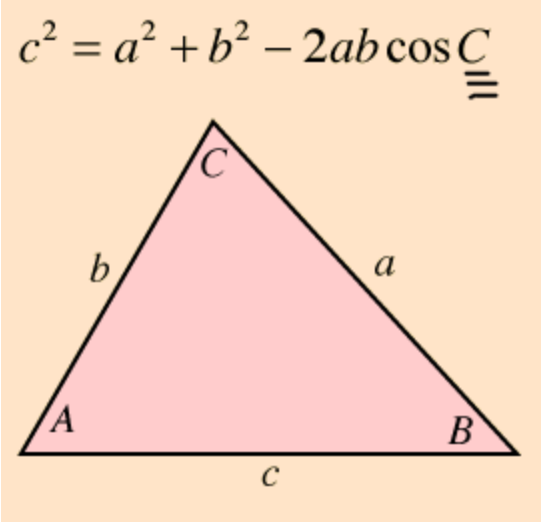
Notice that inside of main, I have included instructions on how to import functions from SSS_triangle.py. This will be essential when you write your program. Specifications are below:

1) In the file tree, click **`SSS_triangle.py`**. Finish the stubs of the functions inside that file. You may copy ***your*** completed code from previous functions here.
2) In the file tree, click on **`main.py`**
    a) This is where you should be writing **your** main program. This program should behave in the following manner:
    b) You should ask the user for three side lengths. You may assume that these are numbers and not strings or other date types for now. You may **not** assume that they are given in order from smallest to largest order. Store these variables in three different variables.
    c) After asking for your three sides, use conditionals to test for validity and return a series of statements about the classification of the triangle by sides, angles, as well as the area of the triangle. I have included a few examples of how the program is expected to behave below.

Example of Valid Triangle Output	Example of Invalid Triangle Output
Given information possibilities: 1) SSS 2) SAS 3) ASA 4) AAS Which given information? 1 You have chosen SSS. 1st Side? 5 2nd Side? 4 3rd Side? 3 This triangle is right and scalene. The triangle has an area of 6.0 units squared. The triangle has a perimeter of 12.0 units.	Given information possibilities: 1) SSS 2) SAS 3) ASA 4) AAS Which given information? 1 You have chosen SSS. 1st Side? 1 2nd Side? 1 3rd Side? 2 Those are not valid side lengths!

`The triangle has angle measures of 36.86989765, 53.13010235, and 90.0 degrees.`	

3) Notice that inside `SSS_triangle.py` there is a new function `SSS_angles.` This takes 3 sides of a triangle and reports a **sorted** list of the triangle's angle measures in degrees. Example of output is above. You won't be able to do Part II until you finish this part. The Law of cosines should be a huge help here:



In the equation at left, lower case a, b, and c are side lengths.

UPPER CASE A, B, C are angle measures.

a) Use algebra to solve the equation above to solve for C. I will help you if you get stuck with this. When you are done, check in with me to make sure you have the right equation.

b) You now have an equation that will solve for the measure of C. Python has built in functions to use inverse sine, cosine, etc. You will need to `import math` and use the function `acos()`. This function is python's way of using inverse cosine (also known as arc cosine, hence the a in front of it) .

c) Finally, note that all the answers given from `acos()` will be in terms of radians. You will need to convert these to degrees. The math module can do this easily with `degrees(x)`, which is found in the math module. In this expression, x is some answer in radians that you want to convert to degrees.

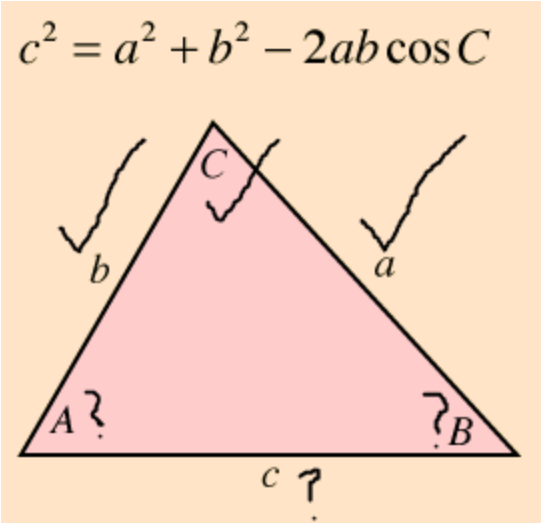
Trigonometric Form	Inverse Trigonometric Form	Inverse Trigonometric Form using Python	Converting from radians to degrees in Python

$cos(60^0) = 0.5$	$60^0 = cos^{-1}(0.5)$	`math.acos(0)`
		`# returns`
		`# 1.570796327`
			`x = 1.570796327`
			`math.degrees(x)`
			`# returns`
			`# 90`

$cos(C) = 1$	$C = cos^{-1}(1)$		

Part II: Creating a new python module.

Create another python file called SAS_triangle.py. The idea behind this file is that it can be used when two sides and their included angle are known. Note: As before, capital letters matter when you define variables. See the picture below. You will know the quantities with check marks and will need to find the quantities with question marks.



a) Inside **`SAS_triangle.py`** , create a function called **`SAS(b,angle_C,a)`** .
   - This triangle will take in two sides (parameters `b` and `a)` as well as an included angle (parameter `angle_C`) as shown above. It will return a sorted **list** of all three sides of the triangle in order from smallest to largest.
   - Use the law of cosines again , but this time use it to solve for c, the missing side across from angle C. Check in with us to make sure your equation is correct.
   - Program this into python so that SAS calculates the side you didn't know previously.
   - Store your sides in a list, sort them, and use the SSS function to return the list.

b) If you did this correctly, you now have a function that will return a list of all the sides of the triangle. This reduces it to the SSS case your previously completed.

Part III: Finish up your main program.

a) Add code in your program to ask the user if they have a SSS case or SAS case.
b) The program should return all parts of the triangle (angles and sides) as well as the triangle's area and perimeter. Of course, if the user gives you impossible values for triangle parts, you should tell her this.
c) Your main program should be as clear and concise as you can make it. Guidelines:

- You should try to make it as readable as possible.
- You should eliminate redundancy as much as possible.

Part IV: Bonus: Add an AAS and ASA component to your program **See Next page.**

Create two additional python files, **`AAS_triangle.py`** and **`ASA_triangle.py`**
You need to use the law of sines for these two cases.

a) For AAS triangle, your python file should take two angle measures (in degrees) of a triangle , angle_A and angle_B, as well as a non-included side, a. It should return a list of two lists:
   i) List of sides in ascending order
   ii) List of angle measures (in degrees) in ascending order

b) For ASA triangle, your python file should take two angle measures (in degrees) of a triangle , angle_A and angle_B, as well as an **included** side, c. It should return a list of two lists:
   i) List of sides in ascending order
   ii) List of angle measures (in degrees) in ascending order
c) Add code to the main program so that you can have an AAS and ASA mode. Make sure all triangle parts are printed out, as well as its area and perimeter.