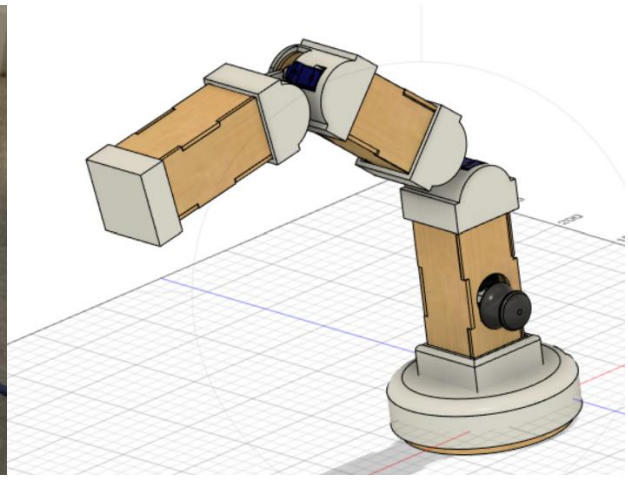
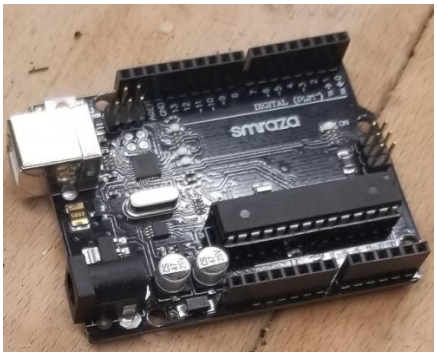
How to make a simple robotic arm



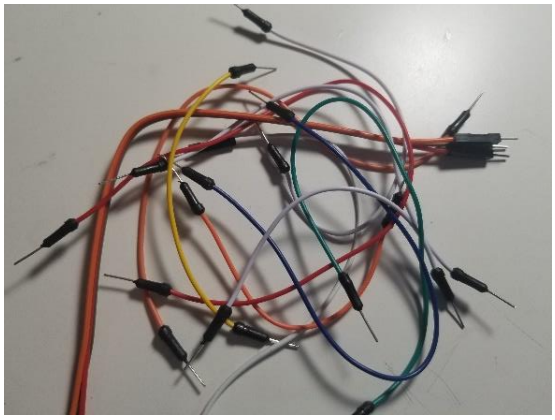
Hi! These instructions will show you how to make a simple robotic arm in the Invention Studio. Before proceeding, it's worth noting that you will need the following that you will need the following things that aren't available in the invention studio:

-arduino uno microcontroller or off-brand equivalent



-power/code usb cord

-wires (depending on what you can find in the miscellaneous bin in the invention studio and how much you want to solder)



Alright, with that covered, let's get started!

This project has 2 main parts- building and programming (I've done all the designing already :)).

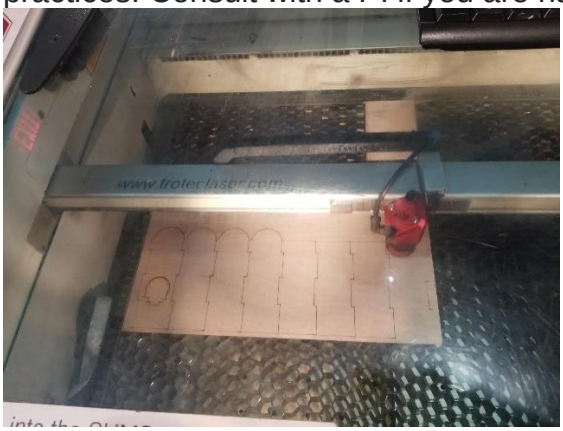
Building:

Step 1: Laser-cutting

A: Plywood Sheet Parts

The file "Robotic Arm Flat Plywood Lasercut.svg" is the inkscape file that has the outlines of every piece that you will need to laser cut. These outlines are close together, but not as close as they should be to maximize space. This is so that you can grab the outlines and move them around to fit whatever shape you need to for the wood you have.

This is all laser cut (everything cut, no engravings unless you are adding them of your own volition) out of ~2.5mm thick plywood sheet(s) (the same used in the woodroom/lasercutter/paint booth tests, so there should be some that you can easily find in the scrap bins) following standard laser-cutting practices. Consult with a PI if you are having any problems with this.



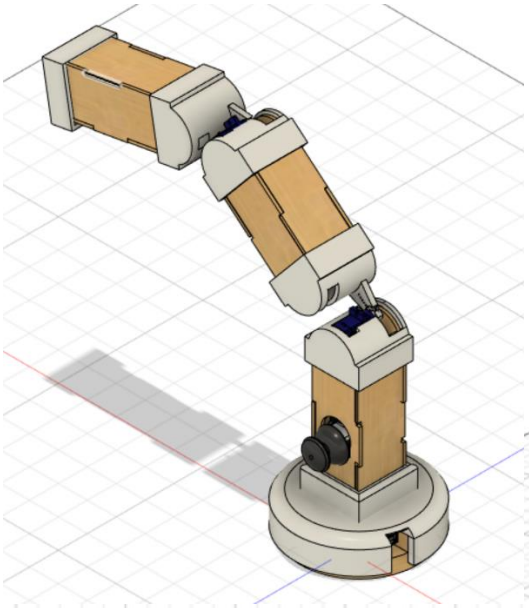
B: Wood Base

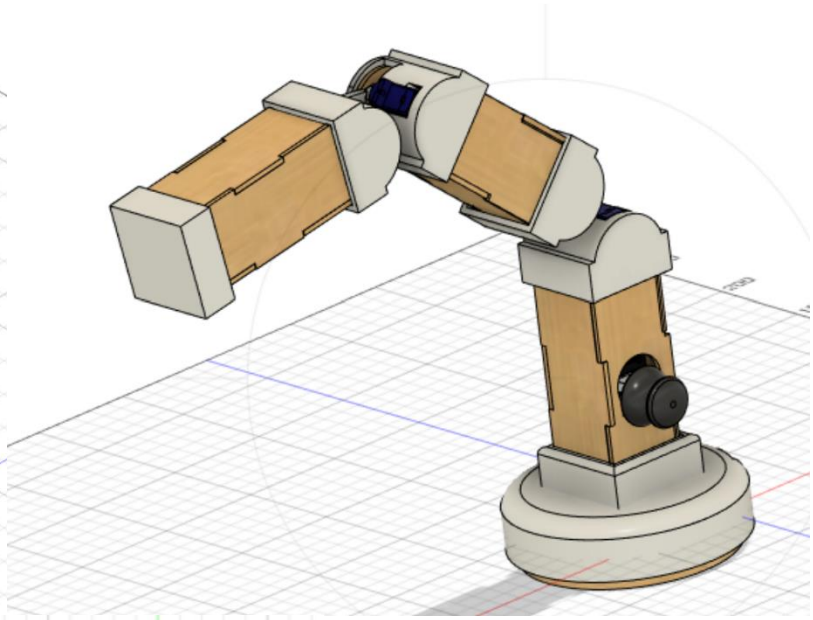
(ignore the velcro. that comes up later)

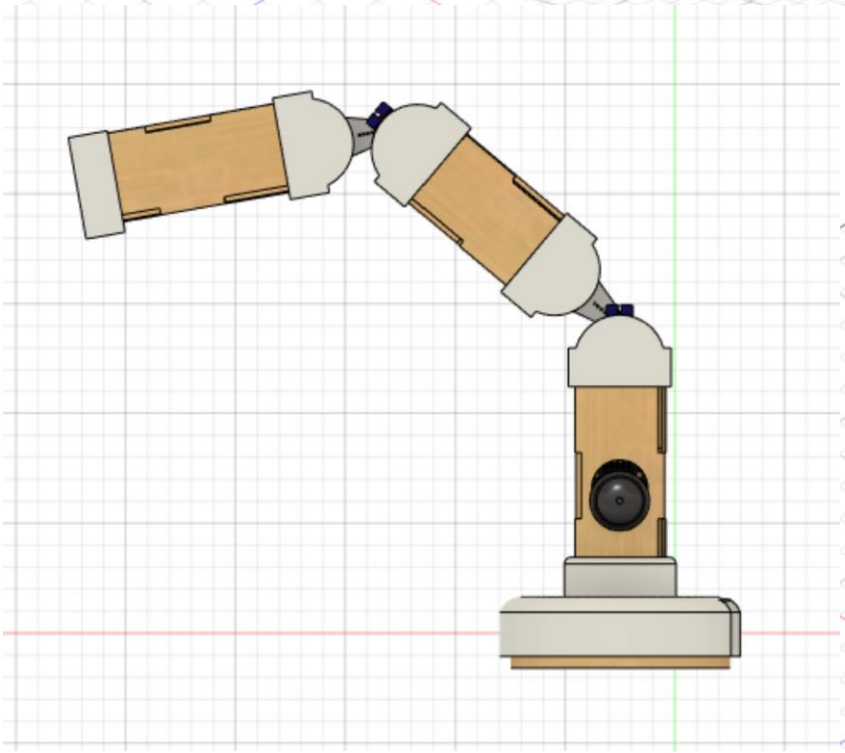
You will need the files "Arduino Uno Circle.svg" and "Full Circle.svg".
I cut one of each of the files out of half inch wood (found in scrap bins), but you could also do this out of four layers of 1/4 inch wood (which would probably be better for the laser cutter). You may have to get creative here, but just be aware of the rough end goal- 1/2 inch full circle and 1/2 inch arduino uno circle.
Once again, please follow proper laser cutting procedures and ask a PI if you need help.

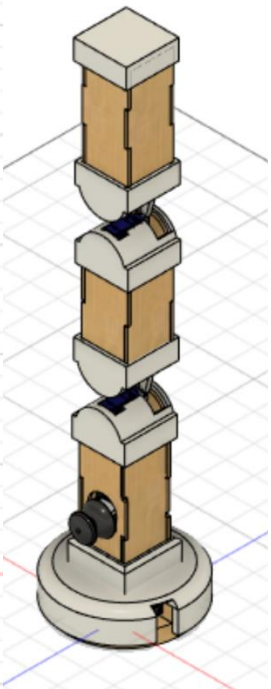
Step 2: 3D-printing

There are 6 STL files attached: Robot Arm Cap, Servo Arm Attachment 1, Servo Arm Attachment 2, Robot Arm Base Cover, Servo Holder Attachment (**print 2 of the Servo Holder Attachment**)
There's nothing of particular note for these pieces. This is what the end design looks like (the white parts are 3D printed):



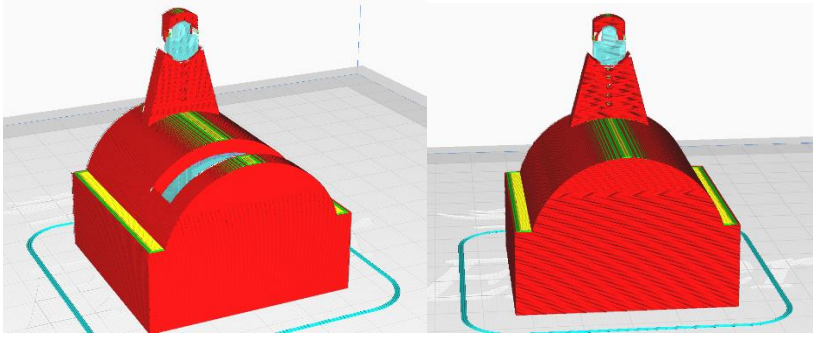






You can also view this directly in fusion from the attached fusion file.
I would suggest using at most 20% infill and being cautious how you orient pieces such that you can successfully remove supports without breaking fragile parts.

Particularly, the servo holder pieces should probably be printed like this:



You can print these pieces in any color and even paint the laser-cut sections as desired.

Step 3: Assembling

A: Materials

You will need the following for the final assembly:

-previously laser cut pieces
-previously 3D printed pieces
-2 small pieces of velcro (one from each side of the velcro).. can be found in craftland
-wood glue.. can be found in the wood room
-arduino uno or equivalent (as noted in the beginning)
-usb cord for arduino uno (as noted in the beginning)
-wires (as noted in the beginning)... may be able to be found in the electronics area
-joystick... found in electronics drawers section
-2 micro servos... found in the electronics drawers section (note: don't forget to get the servo arms and screws, typically in sets in little bags)
-duck tape, as desired... found in craftland
-paperclips
-glue, miscellaneous and as desired

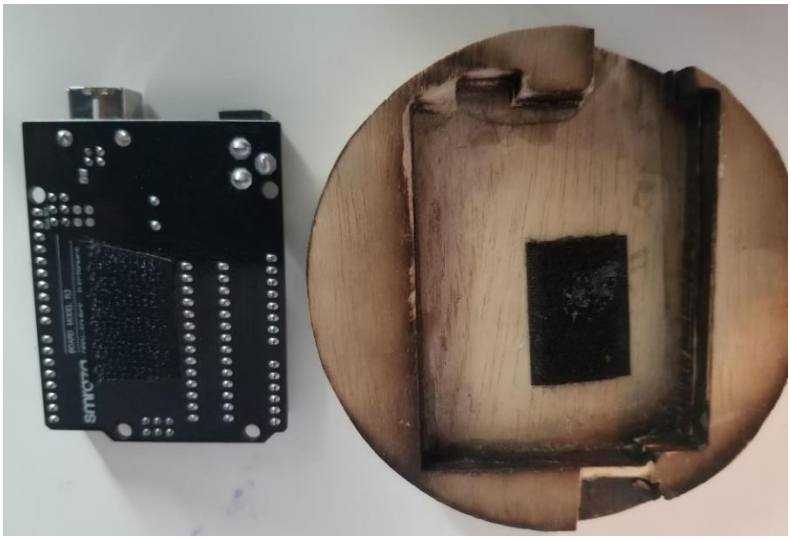
You will not need to gather all of this now, but a master list seems like it would be useful.

B: Base

i) Starting out in the wood room, locate wood glue and clamps. Clamp together the laser-cut pieces of wood for the base (from Step 1-B) with an appropriate amount of wood glue in between them. Be sure to properly align the pieces.

ii) (optional) Once dry, use the belt sander to sand off the edges of the base to remove char marks. Sand out any excess wood glue using sand paper.

iii) Using the velcro pieces, attach a small piece of velcro to the bottom of the arduino uno and the inside of the base. Make sure to line up the pieces properly.



iv) Place the 3D printed base cover on top of the base and make sure that the USB cord can fit through.

C: Arm segment assemblies

Using the rest of the 3D printed pieces and laser cut pieces, put together the arm assemblies. This part is fairly straightforward, but you can also refer to the fusion files if this section is unclear. Use tape and glue as necessary to get pieces to stay together.

For now, this is as far as the assembly goes. The last part, wiring, comes in the final steps of the Programming section, which you can move onto now.

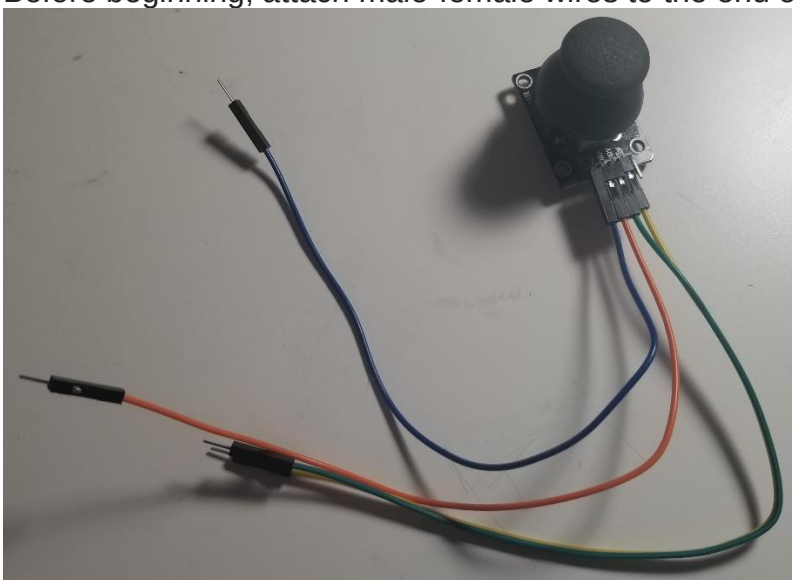
Programming:

Step 1: Configuring

First, download the arduino ide software onto your computer from here: https://www.arduino.cc/en/software. This guide is not intended to teach you how to use this software or troubleshoot any connection issues that will incur. However, I will add that the most common problems come from not connecting the microcontroller properly to the computer, not selecting the right port, or not selecting the right microcontroller. Please refer to the internet for the rest of your arduino based problems. You won't be the only one doing so anyway.

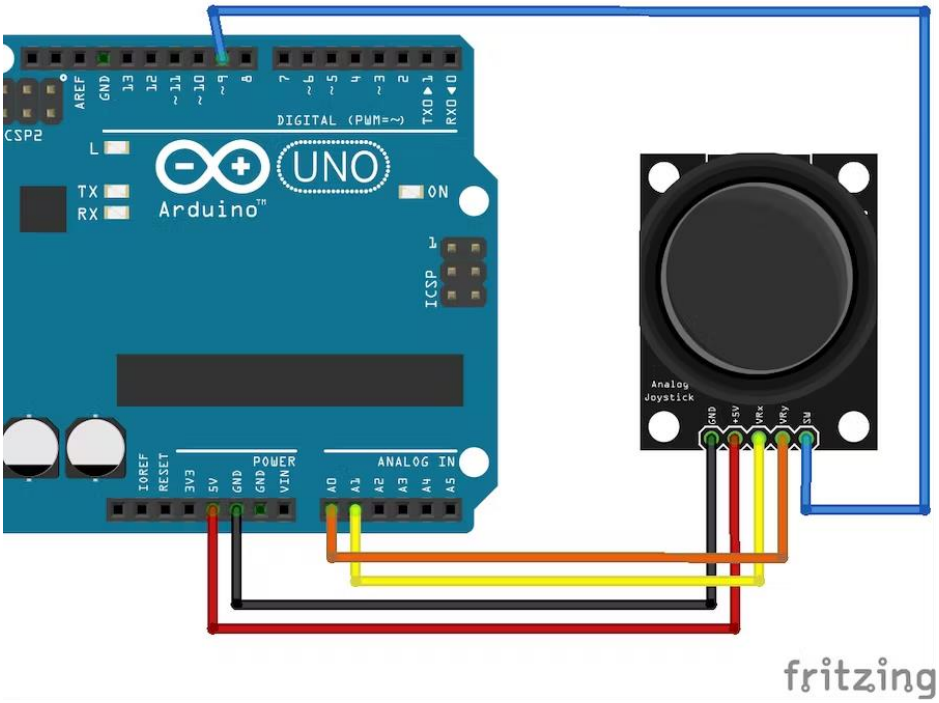
A: Joysticks

Before beginning, attach male-female wires to the end of the joystick like this:



All of this testing is done because individual joysticks vary and that needs to be properly accounted for.

Firstly, wire the joystick as shown in this diagram: (**except for the blue wire. you do not need the blue wire)**



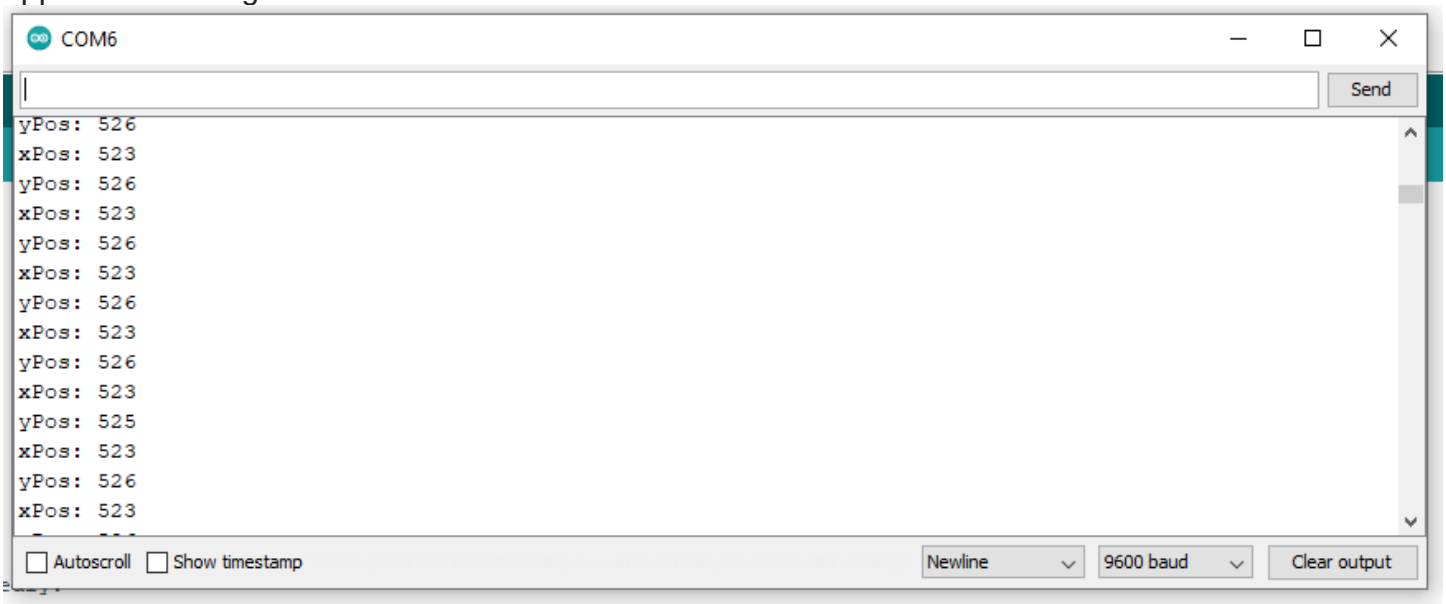
Secondly, run this code to the microcontroller through the arduino ide:

```
int joyX1 = A1;
int joyY2 = A0;
int xPos = 0;
int yPos = 0;

void setup(){
Serial.begin(9600);
pinMode(joyX1, INPUT);
pinMode(joyY2, INPUT);
}

void loop(){
xPos = analogRead(joyX1);
yPos = analogRead(joyY2);
Serial.print("xPos: ");
Serial.println(xPos);
Serial.print("yPos: ");
Serial.println(yPos);
delay(100);
}
```

Third, check the base values for when the joystick is not being moved at all/ is in resting position. (You can see feedback from the arduino in the serial monitor under the tools tab.) They should appear something like this:

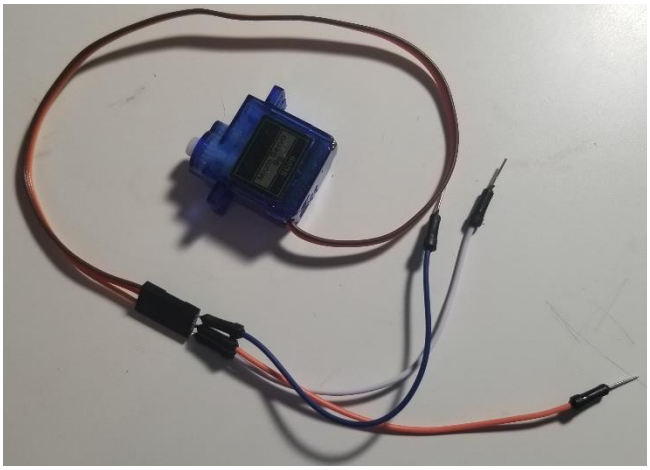


Write those numbers down, for both X and Y. They will be used in the final code as xCenter and yCenter. The numbers may not be 100% consistent. That's ok. Go for a middle ground or the most frequent number if this is the case.
Fourth, move the joysticks to their max positions on each axis, testing the limits. Generally, the values will range from 0 (minimum) to 1023 (maximum). Write down the displayed minimums and maximums for both x and y (these will be xMin, xMax, yMin, and yMax).

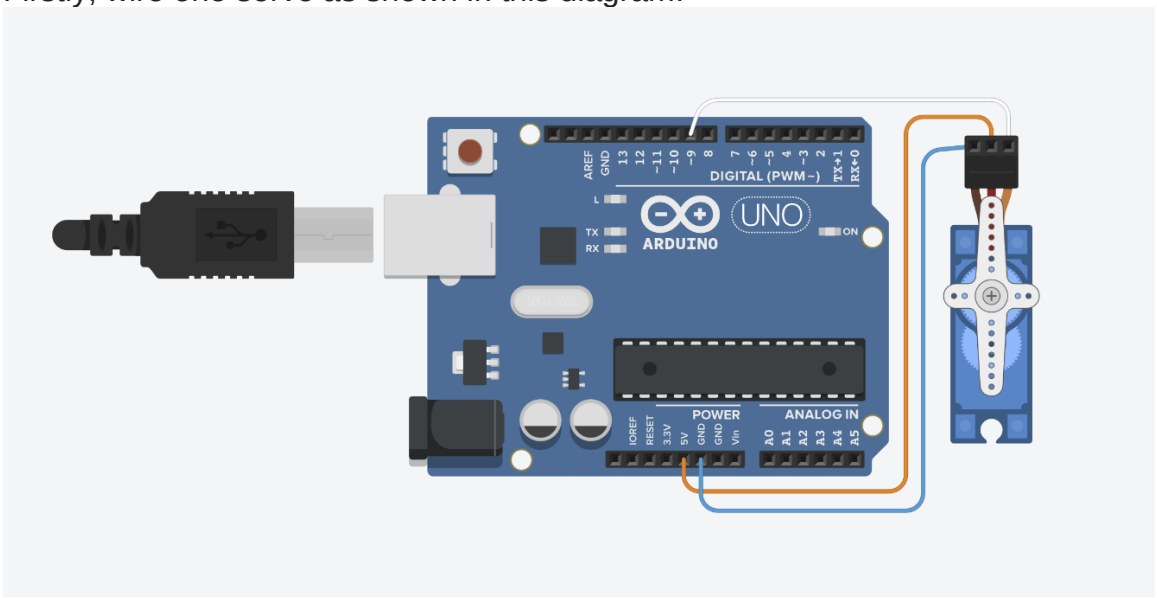
Lastly, slot the joystick into the laser cut piece with a joystick hole:


B: Servos

Before beginning, attach 3 male-male wires like this on the end of each servo:



This testing is done to properly configure the servos and be able to attach them to the final assembly. Firstly, wire one servo as shown in this diagram:



Secondly, run this code to the microcontroller through the arduino ide:

```
#include <Servo.h>

Servo servo1;

void setup(){
servo1.attach(9);
}

void loop(){
for (int pos = 0; pos <180; pos +=1){
servo1.write(pos);
}
for (int pos = 180; pos > 0; pos -=1){
servo1.write(pos);
}
}
```

The servo should be moving fluidly in its full range of motion. If it is not, stop and address the problem. Check that the wiring is right and that the code is right before trying to see if something is wrong with the servo or microcontroller.
Provided that there weren't any problems with the first code, run the following through the ide:

```
#include <Servo.h>

Servo servo1;

void setup(){
servo1.attach(9);
}

void loop(){
servo1.write(90);
}
```

This will set the servo to its middle position. You can attach the servo arm upright here using the small screw from the bag with the servo arms.

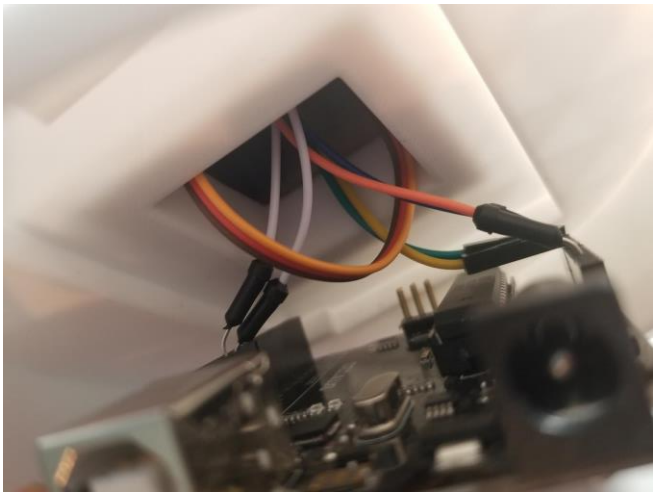
Run the first code again to make sure that the servo arm will still move fluidly in its full range of motion. Once that's working properly, run all of the same steps for the second servo and then move on.
Lastly, attach the servo arm 3D print pieces onto the servos using paperclip pieces as shown here:

You could alternatively try using the other screws that come with the servo arms, though I have not gotten much luck for those.
Step 2: Final Code and Wiring
A: Wiring
Place the servos into their slots in the 3D printed pieces, being sure to cautiously slide the wires through the designated holes. My suggestion would be to place the top servo first, so that you can slide the wires through the bottom servo hole before placing the bottom servo in place.

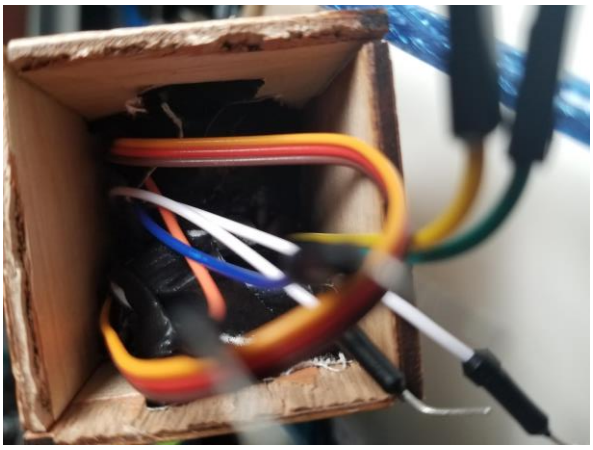
note: picture shown of final product, servo is super-glued in place. that is why it has an odd-looking texture
Alright, this is the point in this process that was the least planned out in advance and will require some compromising on your part:
Each of the servos and the joystick have a ground and a 5V wire, but there is only one 5V port on the arduino uno. Therefore, all of the 5V wires (and optionally all of the ground wires- there are actually 3 ground ports on the microcontroller) need to be connected before attaching to the arduino uno.
Luckily for me, I had a breadboard that I had cut up for a previous project and was able to easily use a piece of it to connect all of the 5V wires and all of the ground wires to the arduino.

Otherwise, you could solder pieces together or find some sort of wire connecting piece. Sorry for the unclearness here.

Connect all of the ground wires.
Connect the x-wire from the joystick to A1 and the y-wire to A0.
Connect the further servo encoder wire to 10 and the closer servo encoder wire to 9.



Ta da! The wiring is done. With things connected, you may want to tape up excess wires:



However, be aware that you may need to make slight alterations to the wiring in case there are problems with the code.
B: Configuration and Final Code
Firstly, copy this code into the arduino ide app:

```
#include <Servo.h>

Servo servo1;
Servo servo2;

int joyX1 = A1;
int joyY2 = A0;

int xCenter = 523;
int yCenter = 526;
int xMax = 1023;
int xMin = 0;
int yMax = 1023;
int yMin = 200;
```

```
int xPos = xCenter;
int yPos = yCenter;
int servo1Position = 90;
int servo2Position = 90;

void setup(){
servo1.attach(9);
servo2.attach(10);
pinMode(joyX1, INPUT);
pinMode(joyY2, INPUT);
}

void loop(){
xPos = analogRead(joyX1);
yPos = analogRead(joyY2);

if (xPos >= ((xMax-xCenter)/2 + xCenter)){
servo1Position = 140;
}
else if (xPos <= (xCenter - (xCenter - xMin)/2)){
servo1Position = 40;
}

if (yPos >= ((yMax-yCenter)/2 + yCenter)){
servo2Position = 140;
}
else if (yPos <= (yCenter - (yCenter - yMin)/2)){
servo2Position = 40;

}

servo1.write(servo1Position);
servo2.write(servo2Position);
delay(100);
}
```

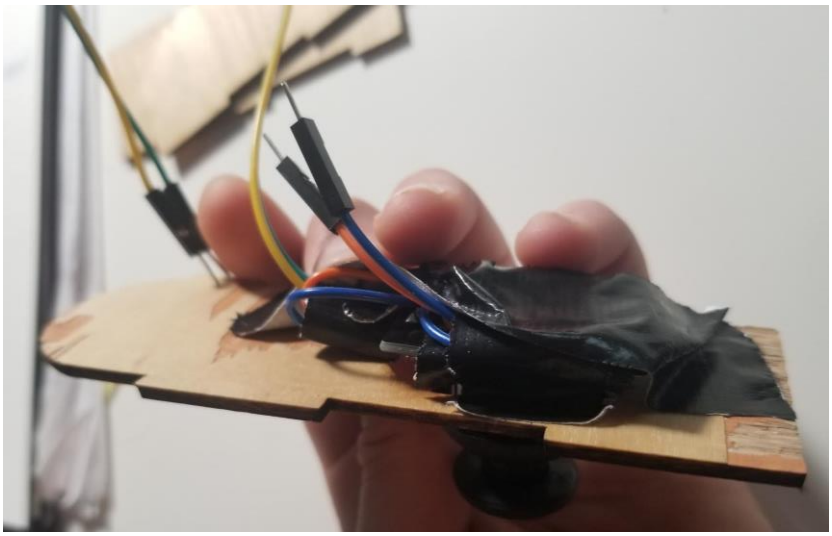
Second, substitute in the previously recorded values for xCenter, xMin, xMax, yCenter, yMin, and yMax.
Third, test the code in the robotic arm.
If the code is having problems, the first changes you'd want to make would be in the minimum and maximum values or the servo values- they may be too large or too small to allow proper movement. You could also change the logic equations used to determine when the joystick has moved enough to warrant moving the servos. (The equations are currently "xPos >= ((xMax-xCenter/2) + xCenter)" and "xPos <= (xCenter - (xCenter - xMin)/2)" and "yPos >= ((yMax-yCenter)/2 + yCenter)" and "yPos <= (yCenter - (yCenter - yMin)/2)".)

Step 3: Further configurations

A: better attaching things

The first suggestion I would have would be to superglue the servos into their 3D printed holders so that they don't move too much during operation. When I was initially testing, the motion would totally throw off the balance of the robotic arm. Super glue worked to greatly reduce this problem.
I would also suggest further attaching the servo arms if you are having problems with wobbling.

If you have not already, please tape the joystick onto the laser cut piece that it is attached to, so that the whole thing doesn't shake as you move the end in operation.



B: code modifications

This is totally down to personal preference, but you may want to add additional increments in the motion of the servos, which would mean adding several if statements dependent on the readings from the joystick (joyX1 and joyY2).

You could even make a formula which would generate a full range of servo values rather than only a few in if-statement increments.

Thank you for reading these instructions and I hope you enjoyed making a robotic arm.