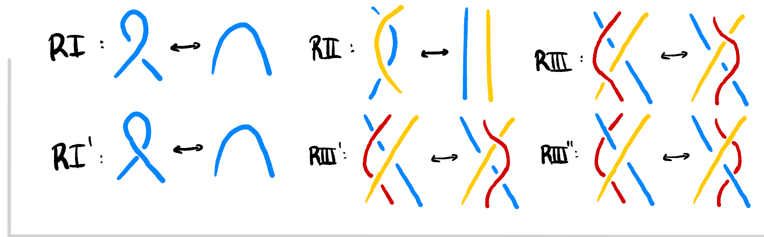
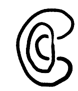
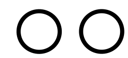
0-1: The unknot and Reidemeister moves.



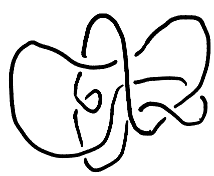
0. Find:

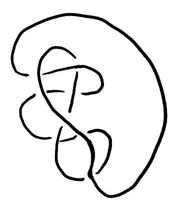
a) A link diagram for the unknot with n crossings, for any $n \geq 1$.

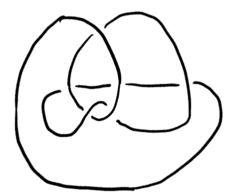
b) A link diagram with n components $(n \geq 2)$ and $2(n-1)$ crossings. (The link must be connected, e.g.  doesn't count)

c) A link with three components such that removing any one component yields two separate unknots: 

1. Simplify the following diagrams until obtaining the unknot, indicating each Reidemeister move.

a) 

b) 

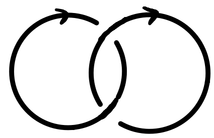
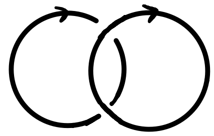
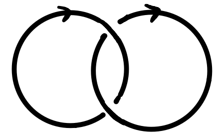
c) 

2. You're handed a link diagram for a knot (1 connected component). You know it only has one crossing. Does the knot have to be the unknot? What if it has 2 crossings? What if it has 3?

3. (Optional) Classify links with 2 components which have a diagram with 2 crossings.

2. Knot theory for computers

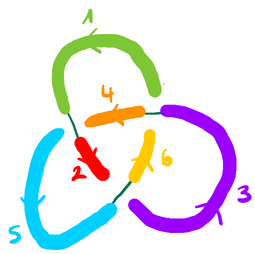
1. Write down the PD code for the following link diagrams:

a)  b)  c) 

Hint: recall the procedure was:

- Step 1:

  Break into pieces:

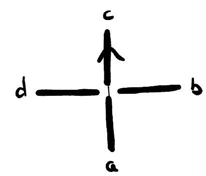
  

- Step 2:

  Decode using the rule:

  $\rightsquigarrow$ [(1,5,2,4),(5,3,6,2),(3,1,4,6)]

RULE:



$\longleftrightarrow$ [a,b,c,d]

2. Draw the link diagrams associated to the following PD codes:

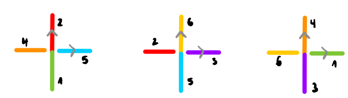
a) [(1,2,2,1)] b) [(1,1,2,2)]

c) [(4,2,3,1), (1,3,2,4)] d) [(3,1,4,2), (4,1,3,2)]

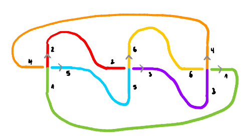
Recall:

- Step 1: draw the crossings you need, anywhere you like:

  [(1,5,2,4),(5,3,6,2),(3,1,4,6)]

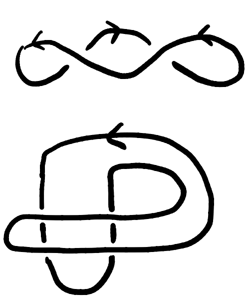
  

- Step 2: match the edges accordingly:

3. Plot the links you obtained in 1 and 2 in Sage.

4. Obtain PD codes for the following link diagrams using SnapPy. Then plot them in Sage.

a) 

b)