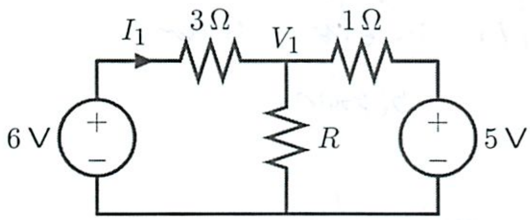
Practice

(w/ same TA review)
when ran out of paper in blue

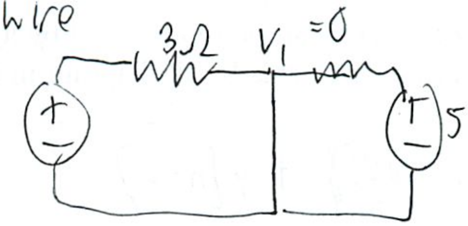
1 Circuits (20 points)

Consider the following circuit where the resistance R is in the range $0 \leq R \leq \infty$.



Part a. Determine I_1 if $R = 0\,\Omega$. wire

$I_1 =$ 2A ✓



$$\frac{6-0}{3} =$$

Part b. Determine V_1 if $R = 1\,\Omega$.

$V_1 =$ 3V ✓

nice!
- really learned this
- well basic stuff
- long work paid off

$$\frac{6-V_1}{3} = \frac{V_1 - 0}{1}$$

$$\frac{6-V_1}{3} + \frac{5-V_1}{1} = \frac{V_1 - 0}{1}$$

right, add

$$\frac{6-V_1}{3} + \frac{15-3V_1}{3} = V_1$$

$$\frac{6-V_1+15-3V_1}{3} = V_1$$

$$\frac{21-4V_1}{3} = V_1$$

$$21 - 4V_1 = 3V_1$$

$+4$

$$21 = 7V_1$$

$$V_1 = 3$$

3 State Machine Behaviors (20 points)

Consider the following state machine.

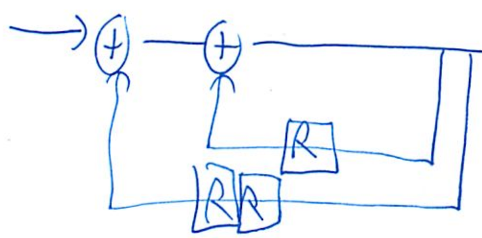
```
class mySystem(sm.SM):
    startState = (0,0)
    def getNextValues(self, state, inp):
        (y1,y2) = state
        y0 = inp + y1 + y2
        return ((y0,y1), y0)
```

TA review session

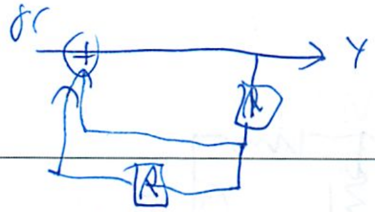
make difference eq

Part a. An instance of mySystem can be represented by a block diagram that contains only adders, gains, and/or delays. Draw this block diagram in the space below.

$Y[n] = x[n] + y[n-1] + y[n-2]$



← not doing anything to top line



Part b. Determine the result of the following Python expression.

```
mySystem().transduce([1,0,0,0,0,0,0,0])
```

Enter the result in the box below.

Fibinical sequence

1 1 2 3 5

1,0,0 → new state (1,0)

0,1,0 → state (1,1)

0,1,1 → state (2,1)

0+2+1 → (3,2)

0+3+2 → (5,3)

Part c. Determine the magnitude of the dominant pole of the system represented by `mySystem()`, and enter it in the box below.

magnitude of dominant pole: $\frac{1+\sqrt{5}}{2}$

(been so long since I did this!)

have diff eq already
need system function

$Y = X + YR + YR^2$

$Y(1-R-R^2) = X$

$\frac{Y}{X} = \frac{1}{1-R-R^2}$

now solve for roots of denom in z form!

$R = \frac{1}{z}$

$z^2 - z - 1$

$\frac{-1 \pm \sqrt{-1^2 - 4(-1)}}{2}$

$= \frac{1+\sqrt{5}}{2}$ ← dom pole

Unrelated

Part d. In the space below, write a new subclass of `sm.SM` called `newSystem`. Instances of `newSystem` should have a system function H given by

$$H = \frac{Y}{X} = 1 - \mathcal{R}^3$$

$Y = X - XR^3$

$y[n] = x[n] - x[n-3]$

remembers format

```
class newSystem(sm.SM):
    def getNextValue(self, state, inp)
        (x1, x2, x3) = state
        y0 = inp - x3
        return((inp, x1, x2), y0)
```

state

remember open book

Own again

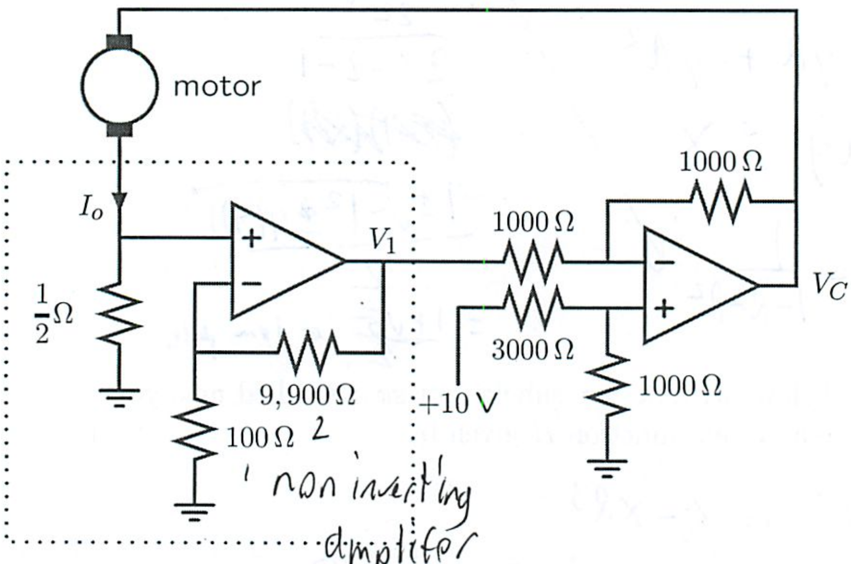
4 Motor Control (20 points)

The following circuit is a proportional controller that regulates the current through a motor by setting the motor voltage V_C to

$$V_C = K(I_d - I_o)$$

op amp h, or w/ circuits

where K is the gain (notice that its dimensions are ohms), I_d is the desired motor current, and I_o is the actual current through the motor. Although K and I_d are not explicit in this circuit, their values can be determined from the resistor values below (see part b).



Part a. Consider the circuit inside the dotted rectangle. Determine an expression for V_1 as a function of I_o.

$V_1 =$

$$V_o = V_i \frac{R_1 + R_2}{R_1}$$

$$= V_i \frac{100 + 9900}{100}$$

?? but have current — voltage divider

$$= V_i \frac{1000}{100}$$

$$= V_i \cdot 10$$

- current in does not matter!

$$= \frac{1}{2}\Omega \cdot I_o \cdot 100$$

why 100, not 10??

look up current thing

$V = iR = \underbrace{I_o \cdot \frac{1}{2}}$

oh duh

Part 2

$k =$ - what is I_d? desired - also need to cons that

- or now $k = 10$ (or 100)?

unless talking second

~~#~~ ? can find V_C and then work backwards

? or property of the op amp

- we already had a non inverting op amp

- other thing is subtractor

$$\frac{V_1 - V_-}{1000} = \frac{V_- - V_C}{1000}$$

$$1000(V_1 - V_-) = 1000(V_- - V_C)$$

$$V_1 - V_- = V_- - V_C$$

$+V_-$ $\quad$ $+V_C$

want V_-

$$\cancel{V_1 = 2V_-}$$

$$V_1 + V_C = 2V_-$$

$$V_- = \frac{V_1 + V_C}{2}$$

$$\frac{10V - V_+}{3000} = \frac{V_+ - 0}{1000}$$

$$1000(10 - V_+) = 3000\,V_+$$

$$10 - V_+ = 3V_+$$

$$10 = 4V_+$$

$$V_+ = 2.5$$

$$V_- = V_+$$

$$2.5 = \frac{V_1 + V_C}{2}$$

$$5 = V_1 + V_C$$

$$V_C = 5 - V_1$$

; so how does that help w/ I_x, I_d?

just cheat

KCL at − input

$$\frac{V_C - 2.5}{1000} = \frac{2.5 - 50\,\Omega \cdot I_o}{1000}$$

oh! had V_1

$$V_C - 2.5 = 2.5 - 50\, I_o$$

$$V_C = 5V - \underbrace{50\, I_o}$$

I had - just needed to plug in

$$= 50(.1A - I_o)$$

what is this? factor

I_x I_d

I just did not "realize" it

Oh! in that form they wanted

- never remember seeing that type of q
- still should be able to solve anyway!

Part b. Determine the gain K and desired motor current I_d.

$K =$

$I_d =$

← Unrelated TA notes

Pot

1k

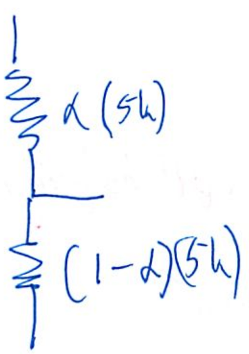
10

Symetric
- does not matter which terminal you use

at start - no resistance take - full Voltage
as slide more " " so voltage drops

α = normalization of angle

5 V = end



basically a voltage divider

5 Members of the Club (20 points)

We would like to make a class to represent a club. A club has a list of members and a scoring function, which takes a member as an input and returns a numerical value. When a member is proposed for addition to the club, it is only added if its score is **greater than the average score of the current members.**

5.1 Join the club

We would like to make a club of basketball players, who are scored on their height. Here is a simple `BallPlayer` class.

```
class BallPlayer:
    def __init__(self, name, height):
        self.name = name
        self.height = height

    def getHeight(self):
        return self.height

    def __str__(self):
        return 'BallPlayer('+self.name+', ' + str(self.height) +')'
```

The `Club` class has an `__init__` method that takes two arguments: an initial member, and a scoring function (that takes a a single member as input and returns a numerical value).

Write an expression below that will create a new club. The first member is person named 'Wilt' whose height is 84 inches. The scoring function for club members should return their height.

c = BallPlayer("Wilt", 84)

Club(BallPlayer('Wilt', 84), BallPlayer.getheight)

? so where in all world is that defined?

Now, imagine that we try, successively, to add the following new players, whose names and heights are listed below, to club c:

- 'Shorty', 60
- 'Walt', 86

? what is club schema

-? just a person

- 'Stilt', 90
- 'Larry', 85

List the resulting membership of club c.

Wilt Walt Stilt

?? what are these

5.2 Implementation

Fill in the definition of the Club class. Use a list comprehension in the averageScore method.

ugr

here we go

```
class Club:
    def __init__(self, firstMember, scoreFunction):
        self.members = [firstMember]
        self.scoreFunction = scoreFunction

    # Returns average score of current members.
    def averageScore(self):
        [for x, scoreFunction in self.members]

        n=float(len(self.members)
        return sum([self.scoreFunction(m) for m in self.members])

    # Adds member if it meets the criterion.  Returns True if the
    # member was added and False, if not.
    def proposeMember(self, member):
        if member.scoreFunction(member) > self.averagescore()
            self.members.append(member)
            return True
        else:
            return False
```

Oh score ≠ height

too tired to do this!

5.3 Histogram

Write a procedure to compute a histogram of the member scores, that is, a count of how many members' scores fall within some specified ranges. We specify ranges by a list of N upper bounds. For example, the following bound list ($N = 3$):

```
[3, 9, 12]
```

↑ 3 items

specifies the following $N + 1 = 4$ ranges:

```
x < 3, 3 <= x < 9, 9 <= x < 12, 12 <= x
```

Given a list of scores that is already sorted from smallest to largest, such as: `[1, 1, 4, 5, 7, 15]` the resulting histogram would be: `[2, 3, 0, 1]` That is, 2 scores less than 3, 3 scores between 3 and 9, 0 scores between 9 and 12 and 1 score above 12.

```
>>> histogram([1, 1, 4, 5, 7, 15], [3, 9, 12])
[2, 3, 0, 1]
```

Cool project

The output should always have $N + 1$ values; values should be zero if there are no scores in the appropriate range.

- don't try to make efficent

```
def histogram(scores, bounds):
```

~~for x in scores~~

ans=[]

bottom ~~current~~ = bound[0]

top = 0

~~while~~

for score in scores:

if top ≤ score < bottom:

ans[-1] = ans[-1] + 1

↑last

(then need to increment top bottom)

n=0

for bound in bounds

top = bound[n]

bottom = bound[n+1]

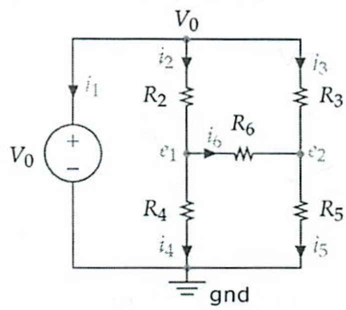
n++ →

~~Score thing again~~

(not doing a clean job - need to write a few times)

1 Short-Answer Questions (10 points)

Part a. Consider the following circuit.



Determine if the following equations and/or statements are

– **Always True** – i.e., true for all possible values of the resistors $R_2 - R_6$ and voltage V_0

or

– **NOT Always True** – i.e., false for some or all resistor and voltage values.

oh not "Always Not true"

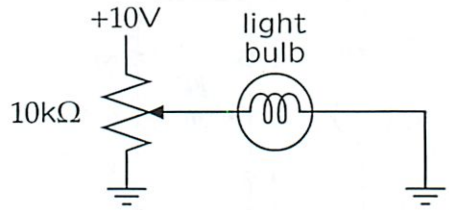
Check the appropriate box for each of the following:

NOT Always True	Always True	
X	X	If $\frac{R_2}{R_4} = \frac{R_3}{R_5}$ then $i_6 = 0$?
X	X	$i_2 + i_3 = i_4 + i_5$ does current change through a resistor? ~~too lazy too look up now~~ no of course it does not!
X		$i_2 + i_6 = i_3$
X	X	? point $e_1 = \frac{R_4}{R_2 + R_4} V_o$ voltage divider — $\frac{\text{bottom}}{\text{all}}$ oh remember point
	X	If $i_6 = 0$ then $\frac{R_2}{R_2 + R_4} = \frac{R_3}{R_3 + R_5}$

yeah same

ouch

Part b. Our goal is to design a dimmer for a light bulb that can be modelled as a constant resistor of 10Ω. We have a single 10V power supply. Our first design uses a 10kΩ potentiometer, with the goal of controlling the current through the lamp so that the current is **proportional** to the potentiometer setting, with a maximum current of 1A.



Briefly (using fewer than 50 words) describe problems with this circuit.

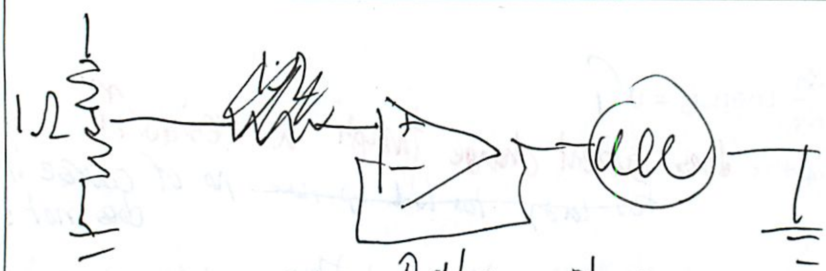
lightbulb resistance much lower - parallel
so $V = iR$
does not do much

$\frac{10k \cdot 10}{10k+10}$ $\frac{10{,}000 \cdot 10}{10010} = \frac{100{,}000}{10010} = 10$

$\frac{5k \cdot 10}{5k+10}$ $\frac{5{,}000 \cdot 10}{5010} = \frac{50{,}000}{5000} = 10$

10A = 1A
oh opps - all other places would limit to < 1mA

Suggest a better circuit. Draw it in the following box. more of an on-off switch



? always op amp

They did that

? opposite that experiment.

? I would say always on

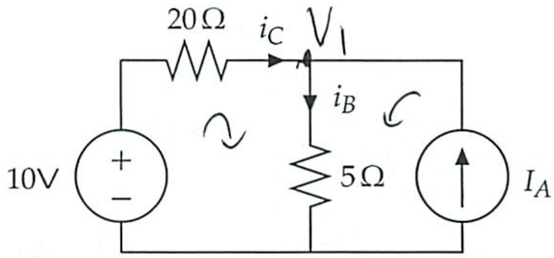
Explain briefly why your circuit is better. ? op amp less way ?

Op Amp buffers bulb - provides constant voltage
does not load pot - pot just specifies
and bulb does that

I just don't like thinking about current!

6 Circuits (20 points)

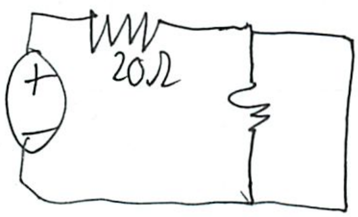
Consider the following circuit.



Part a. Find i_C if $I_A = 0$.

Oh the perpetual halved both V+; Source

i_C = $\boxed{~~1/2 A~~ \ .4}$ ← good ignore it



← wire bypasses 5 Ω

$V = iR$

$10 = i\,20$

$i = \frac{1}{2}$

Part b. Find i_B if $I_A = 5$A.

i_B = $\boxed{~~2.75A~~}$

4.4

$\frac{10V - V_1}{20} + I_A = i_b = \frac{V_1 - 0}{5}$

$\frac{10V - V_1}{20} + I_a = \frac{V_1}{5} = i_b$; two = signs

$\frac{10V - V_1}{4} + I_a \cdot 5 = V_1$

Part c. Find I_A so that $i_C = 0$.

I_A = $\boxed{2A}$

Disaster!

$10V - V_1 + 20 I_a = V_1$

$10V + 20 I_a = 2V_1$

$5V + 10 I_a = V_1$

$\frac{10 - (5 + 10 I_a)}{20} + I_a = i_b$

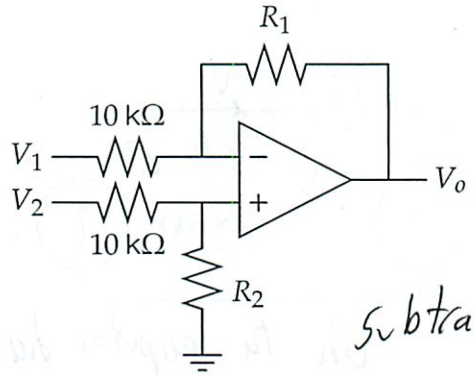
$\frac{5 - 10 I_a}{20} + I_a$

$\frac{1}{4} - \frac{1}{2} I_a + I_A = i_b$

$\frac{1}{4} + \frac{1}{2} I_A = i_B$

$\frac{1}{4} + \frac{1}{2} 5 = 2.75$

Part d. Consider the following op-amp circuit.



Subtractor

Fill in the values of R_1 and R_2 required to satisfy the equations in the left column of the following table. The values must be non-negative (i.e., in the range $[0, \infty]$). If the equation is impossible to implement with non-negative resistors, then write "impossible" for both resistor values.

	R_1	R_2
$V_o = 2V_2 - 2V_1$		
$V_o = 2V_2 - V_1$		
$V_o = V_2 - 2V_1$		
$V_o = 4V_2 - 2V_1$		

- oh this was before
- not in lecture notes

$$\frac{V_1 - V_-}{\cancel{R_1}\ 10k} = \frac{V_- - V_o}{R_1}$$

$$\frac{V_2 - V_+}{10k} = \frac{V_+ - 0}{R_2}$$

$$V_+ = V_-$$

$$R_1(V_1 - V_-) = 10(V_- - V_o)$$

$$R_2(V_2 - V_+) = 10k(V_+)$$

$$R_1V_1 - R_1V_- = 10V_- - 10V_o$$

$$R_2V_2 - R_2V_+ = 10V_+$$

$$R_2V_2 = V_+(10 + R_2)$$

$$V_+ = \frac{R_2V_2}{10 + R_2}$$

a) $i_c = \dfrac{10V}{20\Omega + 5\Omega} = \dfrac{2}{5}A = .4A$

Oh I ~~assumed~~ would become wire
-opps becomes a air gap

b) what is this? ~~voltage divider~~? current divider
-resistors in parallel
↓only interested in part going down?
-that does not make sense though

$$i_b = \frac{10V}{20+5\Omega} + \underbrace{\overbrace{\frac{20\Omega}{5+20}}}_{\text{remember } I_a \text{ the same!}} \cdot 5 = 4.4$$

← want that ant not
↓

c) try again
-want $i_c = 0$

$$i_c = 0 = \frac{10V}{20+5} - \overbrace{\frac{5}{5+20}}^{\text{now}} \cdot I_A$$

← part of I_a

ah want
→ ← to = out to 0

$$= \frac{2}{5} - \frac{1}{5} I_A$$

$I_A = 2$

part d cont

$$R_1 V_1 + 10 V_0 = 10 V_- + R_1 V_-$$

$$R_1 V_1 + 10 V_0 = V_- (10 + R1)$$

$$V_- = \frac{R_1 V_1 + 10 V_0}{10 + R_1}$$

$$\frac{R_2 V_2}{10 + R_2} = \frac{R_1 V_1 + 10 V_0}{10 + R_1}$$

$$(10 + R_1) R_2 V_2 = (10 + R_2)(R_1 V_1 + 10 V_0)$$

$$10 R_2 V_2 + R_1 R_2 V_2 = 10 R_1 V_1 + R_2 R_1 V_1 + 100 V_0 + 10 R_2 V_0$$

$\cancel{100 V_0}$

$$V_0 (100 + 10 R_2) = 10 R_2 V_2 + R_1 R_2 V_2 - 10 R_1 V_1 + R_2 R_1 V_1$$

$$V_0 = \frac{10 R_2 V_2 + R_1 R_2 V_2 - 10 R_1 V_1 + R_2 R_1 V_1}{100 + 10 R_2}$$

oh crap
what to do now
Say ~~R2=1~~ $V_1, V_2 = 1$

$10 R_2 2 + R_1 R_2$

something went wrong I think

$$V_+ = \frac{R_2}{10 + R_2} V_2 = V_- = \frac{R_1}{10 + R_1} V_1 + \frac{10}{10 + R_1} V_0$$

$$V_0 = \frac{10 + R_1}{10 + R_2} \cdot \frac{R_2}{10} \cdot V_2 - \frac{R_1}{10} \cdot V_1$$

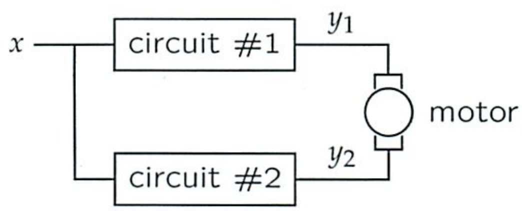
(or maybe it was that complex

3. Circuits (30 / 100 points)

Motor driver

When we built the robot head, we made the motor move in both directions by connecting one side of the head motor to an op amp circuit and the other side to a buffered voltage divider that produced +5V. This method limited the peak speeds of the motor because the full +10V that is available from the power supply never appeared across the motor.

Our goal is to build two circuits, one to drive each of the two motor wires. Let x represent the input voltage and let y_1 and y_2 represent the voltages applied to the two motor wires. Assume that you have a single 10-volt power supply, so that only +10V and 0V are available.



Question 13: **Recall the supply voltage constraints: ALL voltages (including x, y_1 and y_2) must be between 0V and +10V.** Determine expressions for y_1 and y_2 so that the voltage across the motor is given by

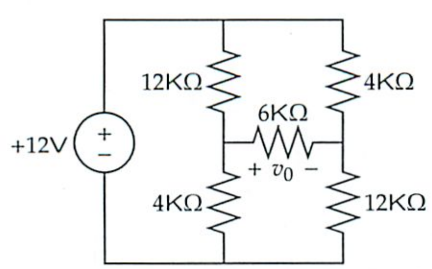
$$y_1 - y_2 = \begin{cases} 10 & \text{if } x = 10 \\ -10 & \text{if } x = 0 \end{cases}$$

and both y_1 and y_2 have the form $y_i = m_i x + b_i$, where m_i and b_i are constants.

Question 14: Design circuits to implement your solutions to question 13 using resistors, op amps, and a single +10V power supply. You can assume that x is buffered (i.e., it is the output of an op amp). Draw the circuits and label them clearly.

Analysis

A "bridge" circuit consisting of five resistors is connected to a 12 volt source as shown below.



Bridge circuit

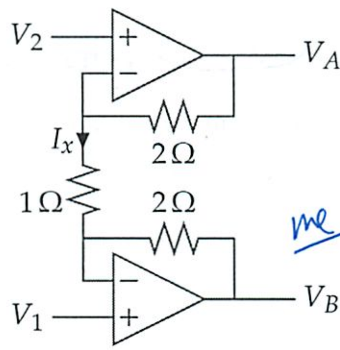
TA Review

NanoQuiz 10 Makeup

Name:

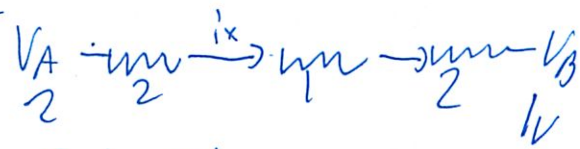
Part 1: Op Amps. Assume the op-amps in the following circuit are "ideal."

$V_+ = V_-$



ha - the one I was stuck on
can I remember?

me



$2-1 = 5i$

~he liked my trick

–oh V_A and V_B are not really known

a) Determine the current I_x when $V_1 = 1\,\text{V}$ and $V_2 = 2\,\text{V}$.

$I_x =$ 1

b) Determine the voltage V_A when $V_1 = 1\,\text{V}$ and $V_2 = 2\,\text{V}$.

$V_A =$ 4

c) Determine a general expression for V_A in terms of V_1 and V_2.

$V_A =$ -2 $* V_1 +$ 3 $* V_2$

↓ Opposite here

i same over whole thing

so $V_2 - V_1 = i \cdot 1$ $\quad \frac{V_2 - V_1}{R} = I_x$

$1 = i \cdot 1 \quad i = 1$

b) $V_A - V_2 = iR$

$V_A - 2 = 1 \cdot 2$

$V_A = 4$

$V_A = iR + V_2$

c) but where V_1?

$\frac{V_A - V_1}{3} = \frac{V_2 - V_1}{1}$ current same on both so =

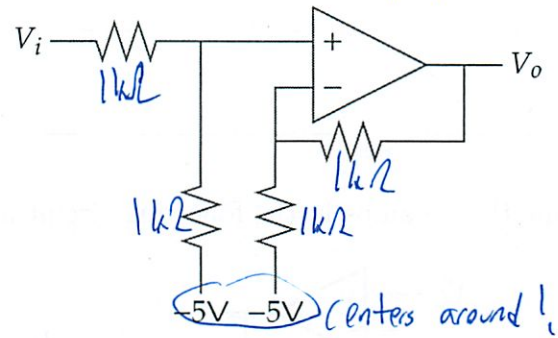
(don't rely too much on tricks!)

$V_A - V_1 = 3V_2 - 3V_1$

$V_A = 3V_2 - 2V_1$

... continued on back of page

Part 2: Op Amps. Consider the following circuit: $V_+ = V_-$



centers around !

where all of the resistors have the same value $R = 1\,\mathrm{k\Omega}$.

If $V_i = 3\,\mathrm{V}$, then $V_o =$

If $V_i = 7\,\mathrm{V}$, then $V_o =$

If $V_i = 9\,\mathrm{V}$, then $V_o =$

both voltage dividers

a)

$$\frac{V_1 - -5V}{2k} = \frac{V_+ - -5V}{1k}$$

a) $4 - 5 = V_+ = -1$

$V_o = 3V$ ← must be = in between 4V drop across both resistors

b) $V_1 = 7$ a bit different

$$V_+ = \frac{V_1 - 5}{2} = V_- = \frac{V_o - 5}{2}$$

? must be ½ - voltage divider

not easy to see - but ideal voltage dividers !

$V_1 - 5 = V_o - 5$

$V_1 = V_o$

so $V_1 = 5V \rightarrow V_o = 5V$

6.01 Midterm 2: Spring 2010

Name:	**Section:**

Enter all answers in the boxes provided.

During the exam you may:

- read any paper that you want to
- use a calculator

You may not

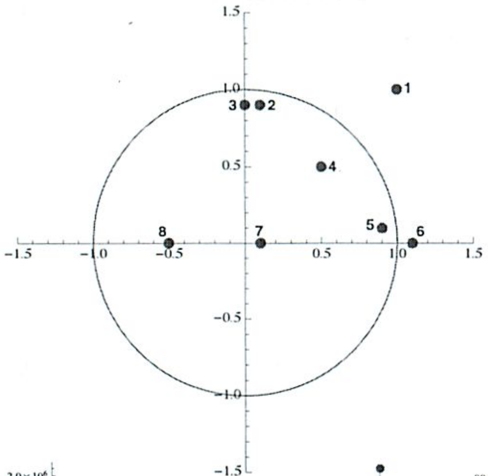
- use a computer, phone or music player

For staff use:

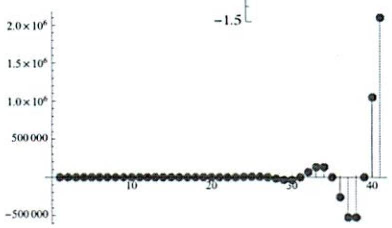
1.	/16
2.	/16
3.	/8
4.	/25
5.	/5
6.	/8
7.	/12
8.	/10
total:	/100

1 Pole Position (16 points)

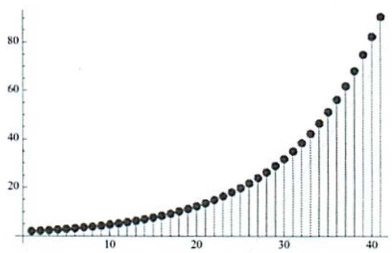
The polar plot shows the dominant pole for several systems. Match each pole to the unit sample response of the system.



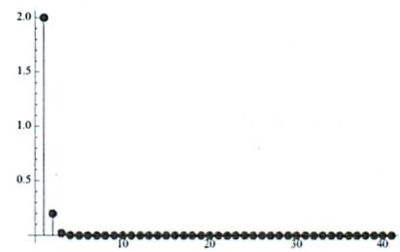
Did in review session



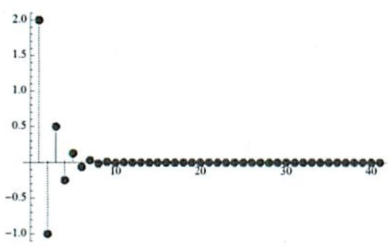
A = ☐



B = ☐

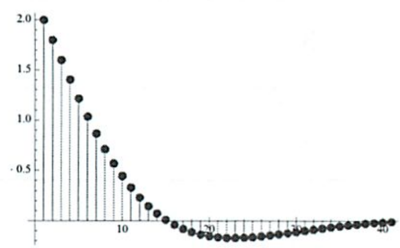


C = ☐

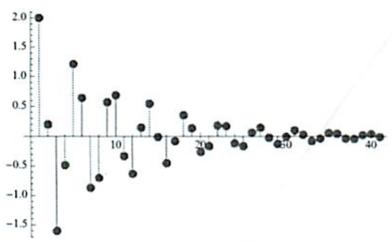


D = ☐

E = ☐



F = ☐



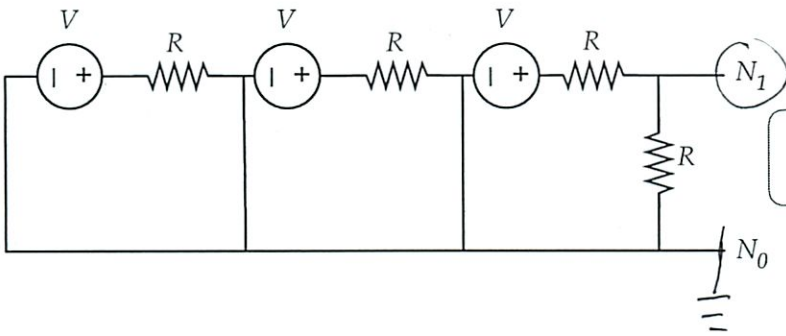
G = ☐

H = ☐

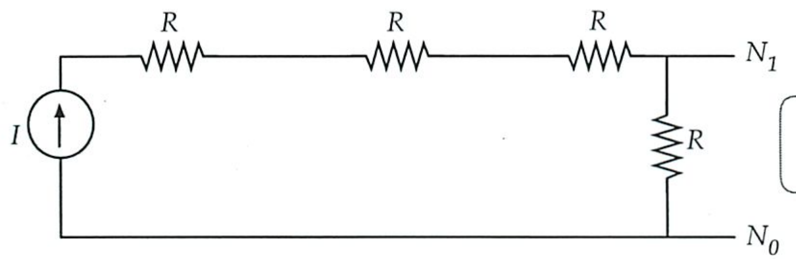
2 What's the difference? (16 points)

Assuming the voltage at node $N_0 = 0$, compute the voltage at node N_1 in each of these circuits.

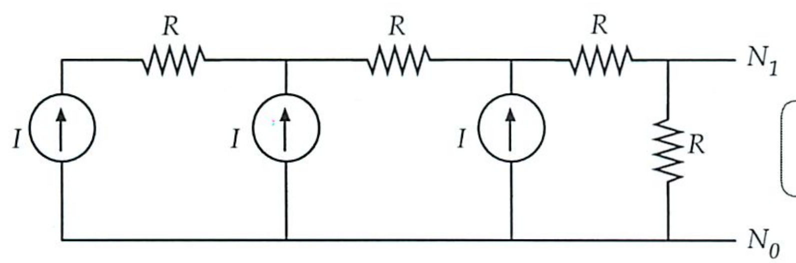
2.1



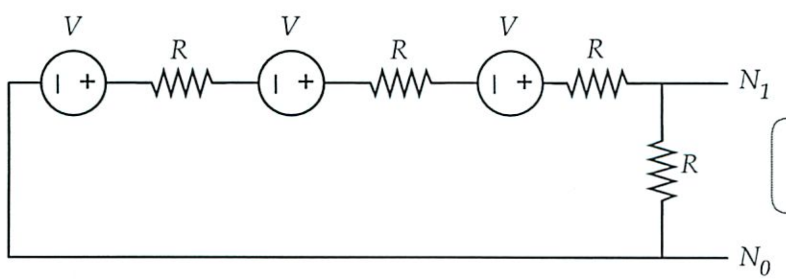
2.2



2.3



2.4

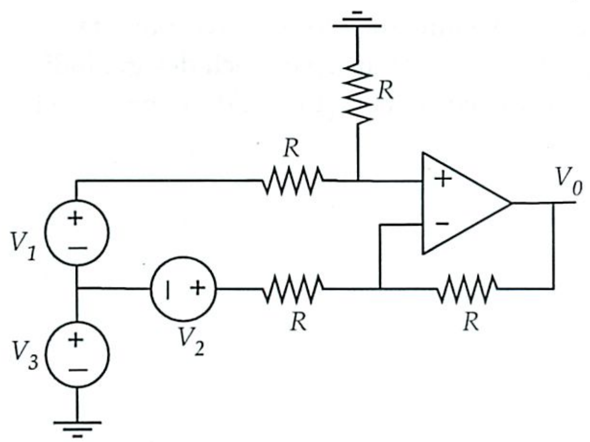


Scratch Paper

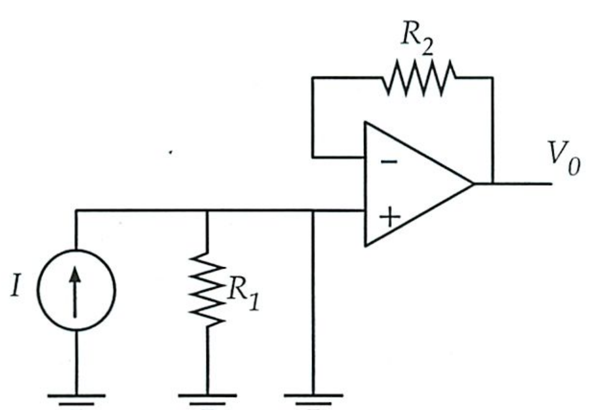
3 Op Amps (8 points)

For each of the following circuits, give the output voltage V_0 as a function of the input voltage sources, input current sources, and resistances. Assume the op-amps are ideal.

3.1



3.2

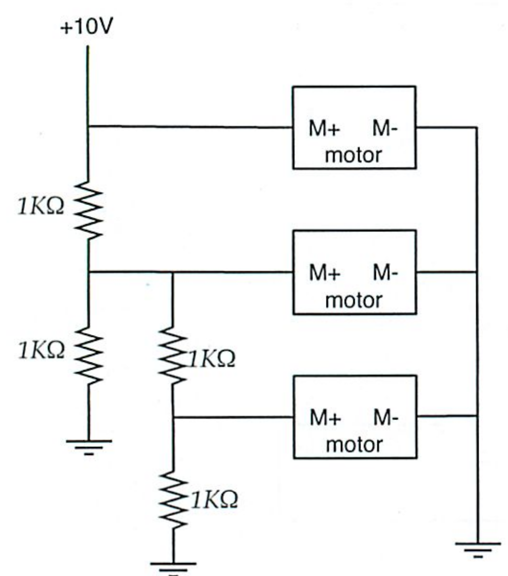


4 Motor control (25 points)

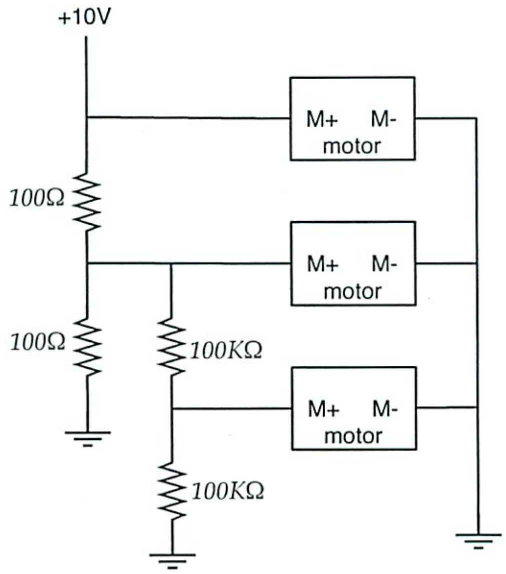
WhizzyLand engineers Kim, Pat, Jody, Chris, and Jamie are trying to design a controller for a display of three dancing robotic mice, using a 10V power supply and three motors. The first is supposed to spin as fast as possible (in one direction only), the second at half of the speed of the first, and the third at half of the speed of the second.

Each engineer has come up with a design, as shown below. Assume the motors have a resistance of approximately 5Ω and that rotational speed is proportional to voltage. For each design, indicate the voltage across each of the motors. Your answer only needs to be within 0.5V of the correct answer.

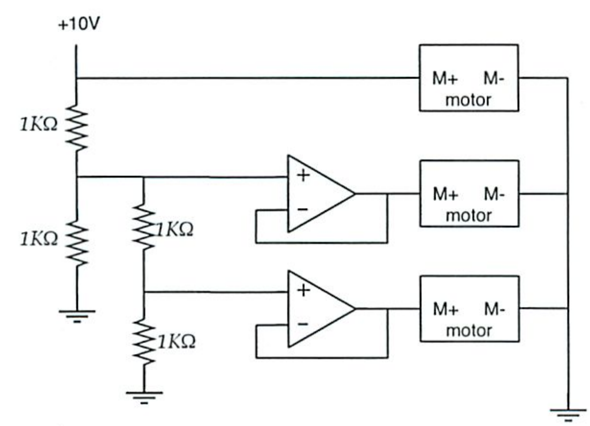
4.1 Jody



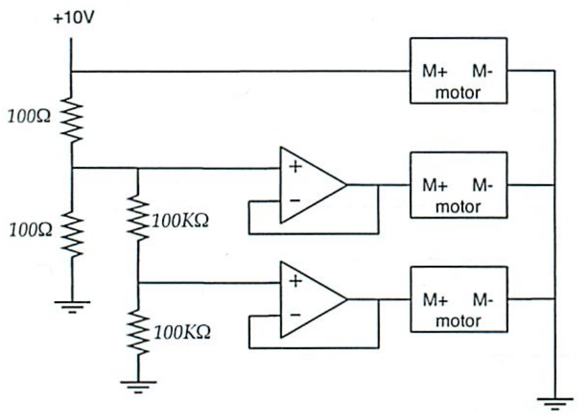
4.2 Chris



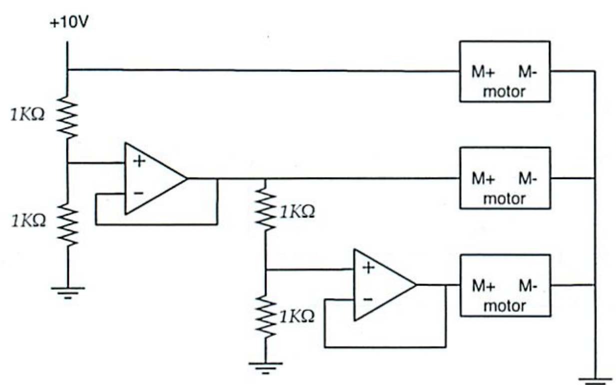
4.3 Pat



4.4 Kim

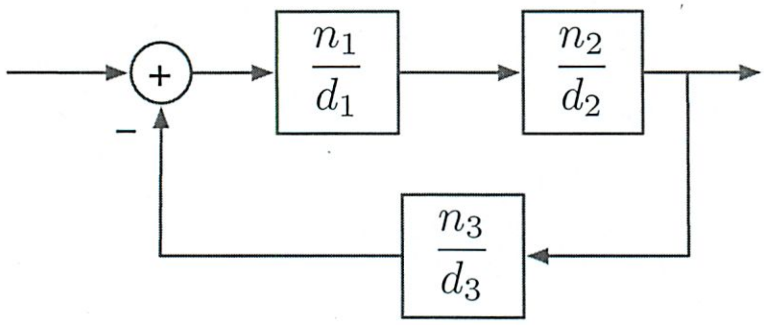


4.5 Jamie



Scratch Paper

5 Composition (5 points)



Write an expression for the system function for this whole system, in terms of $n_1, d_1, n_2, d_2, n_3, d_3$, which are the numerator and denominator polynomials of the system function for each of the component systems.

6 OOP (6 points)

If a procedure has no side effects but may be used multiple times, then a convenient concept for efficiency is called memoization – we keep track of whether the procedure has previously been called with a particular argument, and the value computed at that time. If it has already been computed, we just return the saved value, otherwise we do the computation, store away the value for future use, and return the value.

Here is an object oriented approach for memoization, with one piece missing. We assume that `proc`, the procedure to be "memoized", has a single argument.

```
class Memo:
    def __init__(self, proc):
        self.proc = proc
        self.history = {}
    def val(self,arg):
        if arg in self.history:
            print 'found', arg, 'in history'
            return self.history[arg]
        else:
            # YOUR ANSWER HERE
```

An example usage would be

```
>>> def sq(x): return x*x
>>> test = Memo(sq)
>>> test.val(2)
4
>>> test.val(3)
9
>>> test.val(3)
found 3 in history
9
>>> test.val(8)
64
>>> test.val(5)
25
>>> test.history
{8: 64, 2: 4, 3: 9, 5: 25}
```

6.1

Complete the definition of this class, by supplying the code for the else part of the val method. Make sure that you do not execute the procedure more than once for any given argument.

6.2

The approach above works with any procedure of one argument that does not have side-effects. However, if we know that the procedure has some additional structure, we can be even more efficient. Assume we know that our procedure takes a single number as an argument and that the procedure is **even**, that is, `proc(arg) == proc(-arg)`. Examples of even procedures are `abs`, `math.cos` and `sq` as defined above.

Write a definition for the `MemoEven` class. Your solution must use inheritance, and ensure that there is a `val` method defined. Do not change the definition of `Memo` or repeat any of the code it contains.

7 Red vs Blue (12 points)

Let's build a voting tabulator for the national elections (except that we're going to assume that there are only 4 states). The definitions below initialize the instance and allow us to enter votes, by state, for each candidate.

We're going to keep the votes data in the dictionary `stateVotes`. There is an entry in `stateVotes` for each state. The value stored for a state is itself a dictionary that stores the votes from that state for each candidate. A simple example, for two states (`'MA'` and `'TX'`) and two candidates, `'O'` and `'M'`, would be:

```
{'MA': {'O':100, 'M':50}, 'TX': {'O':50, 'M':100}}
```

```
class Votes:
    def __init__(self):
        self.states = ['MA', 'TX', 'CA', 'FL']
        self.candidates = ['O', 'M', 'N']
        self.stateVotes = {}
    def addVotes(self, cand, votes, state):
        if not state in self.stateVotes:
            self.stateVotes[state] = {cand : votes}
        else:
            if not cand in self.stateVotes[state]:
                self.stateVotes[state][cand] = votes
            else:
                self.stateVotes[state][cand] += votes
```

7.1

Write a method for the `Votes` class, called `popularVotesTotal`, that computes the total number of votes for a candidate, across all states.

- Use a list comprehension, not a `for` loop.
- Use the Python `sum` procedure, which takes a list of numbers as input and returns the sum.

7.2

Write a method for the `Votes` class, called `stateWinner`, that computes the winner for a state, that is, the candidate with the most votes in that state.

Use the procedure `argmaxDict(d)`, which is called with a dictionary `d` and returns the key in `d` whose associated value is highest.

7.3

Write a method for the `Votes` class, called `statesWon`, that takes a candidate as an argument and returns a list of the states that candidate won. Use the `stateWinner` method you just implemented.

7.4

Write a method for the `Votes` class, called `winnerOfMostStates`, that returns the candidate that won the most states. Use the `statesWon` method you just implemented. For full credit, use `util.argmax`.

If `l` is a list of items and `f` is a procedure that maps an item into a numeric score, then `util.argmax(l, f)` returns the element of `l` that has the highest score.

8 Balanced Machine (10 points)

Write a state machine that counts the depth of nesting of parens in a string. It takes a character from the string as input and outputs an integer, indicating how many unmatched open parens there are (on or) before this character or None if the parens are unbalanced. Note that once the parens become unbalanced, all future outputs will be None. Here are some examples:

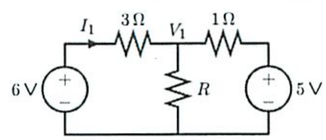
```
>>> bp = BalancedParens()
>>> bp.transduce('(ab (c (d (e)) f))')
[1, 1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 3, 2, 2, 2, 1, 0]
>>> bp.transduce('(())  ((()))  ()()()')
[1, 2, 1, 0, 0, 0, 1, 2, 3, 2, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0]
>>> bp.transduce('(()))  ()()()')
[1, 2, 1, 0, None, None, None, None, None, None, None, None, None]
```

Your state machine should be a subclass of sm.SM.

Solutions

1 Circuits (20 points)

Consider the following circuit where the resistance R is in the range $0 \le R \le \infty$.



Part a. Determine I_1 if $R = 0\,\Omega$.

$I_1 =$ 2 A

$$I_1 = \frac{6\,\text{V}}{3\,\Omega} = 2\,\text{A}$$

Part b. Determine V_1 if $R = 1\,\Omega$.

$V_1 =$ 3 V

$$\frac{V_1 - 6}{3} + \frac{V_1}{1} + \frac{V_1 - 5}{1} = 0$$

$$\frac{7}{3}V_1 = 7$$

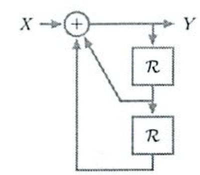
$$V_1 = 3$$

3 State Machine Behaviors (20 points)

Consider the following state machine.

```
class mySystem(sm.SM):
    startState = (0,0)
    def getNextValues(self, state, inp):
        (y1,y2) = state
        y0 = inp + y1 + y2
        return ((y0,y1), y0)
```

Part a. An instance of `mySystem` can be represented by a block diagram that contains only adders, gains, and/or delays. Draw this block diagram in the space below.



Part b. Determine the result of the following Python expression.

`mySystem().transduce([1,0,0,0,0,0,0,0])`

Enter the result in the box below.

`[1, 1, 2, 3, 5, 8, 13, 21]`

Part c. Determine the magnitude of the dominant pole of the system represented by `mySystem()`, and enter it in the box below.

magnitude of dominant pole: $\frac{1+\sqrt{5}}{2}$

$$H = \frac{Y}{X} = \frac{1}{1-\mathcal{R}-\mathcal{R}^2} = \frac{1}{1-\frac{1}{z}-\frac{1}{z^2}} = \frac{z^2}{z^2-z-1}$$

poles are at $\frac{1}{2} \pm \sqrt{\left(\frac{1}{2}\right)^2 + 1} = \frac{1}{2} \pm \sqrt{\frac{5}{4}} = \frac{1 \pm \sqrt{5}}{2}$

Part d. In the space below, write a new subclass of `sm.SM` called `newSystem`. Instances of `newSystem` should have a system function H given by

$$H = \frac{Y}{X} = 1 - \mathcal{R}^3$$

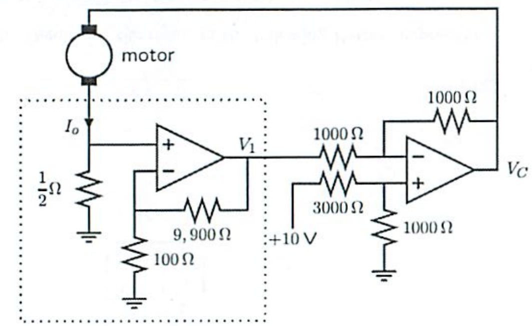
```
class newSystem(sm.SM):

    startState = (0,0,0)
    def getNextValues(self, state, inp):
        (x1,x2,x3) = state
        y0 = inp - x3
        return ((inp,x1,x2), y0)
```


4 Motor Control (20 points)

The following circuit is a proportional controller that regulates the current through a motor by setting the motor voltage V_C to

$$V_C = K(I_d - I_o)$$

where K is the gain (notice that its dimensions are ohms), I_d is the desired motor current, and I_o is the actual current through the motor. Although K and I_d are not explicit in this circuit, their values can be determined from the resistor values below (see part b).



Part a. Consider the circuit inside the dotted rectangle. Determine an expression for V_1 as a function of I_o.

$V_1 =$ $\frac{1}{2}\Omega \times I_0 \times 100$

Part b. Determine the gain K and desired motor current I_d.

$K =$ $50\,\Omega$

$I_d =$ $0.1\,\text{A}$

KCL at negative input to right op-amp:

$$\frac{V_C - 2.5\,\text{V}}{1000\,\Omega} = \frac{2.5\,\text{V} - \frac{1}{2}\Omega \times I_o \times 100}{1000\,\Omega}$$

Solving:

$$V_C - 2.5\,\text{V} = 2.5\,\text{V} - 50\,\Omega \times I_o$$

$$V_C = 5\,\text{V} - 50\,\Omega \times I_o = 50\,\Omega\,(0.1\,\text{A} - I_o)$$

5 Members of the Club (20 points)

We would like to make a class to represent a club. A club has a list of members and a scoring function, which takes a member as an input and returns a numerical value. When a member is proposed for addition to the club, it is only added if its score is **greater than the average score of the current members.**

5.1 Join the club

We would like to make a club of basketball players, who are scored on their height. Here is a simple `BallPlayer` class.

```
class BallPlayer:
    def __init__(self, name, height):
        self.name = name
        self.height = height

    def getHeight(self):
        return self.height

    def __str__(self):
        return 'BallPlayer('+self.name+', ' + str(self.height) +')'
```

The `Club` class has an `__init__` method that takes two arguments: an initial member, and a scoring function (that takes a a single member as input and returns a numerical value).

Write an expression below that will create a new club. The first member is person named 'Wilt' whose height is 84 inches. The scoring function for club members should return their height.

```
c = Club(BallPlayer('Wilt', 84), BallPlayer.getHeight)
```

Now, imagine that we try, successively, to add the following new players, whose names and heights are listed below, to club `c`:

- 'Shorty', 60
- 'Walt', 86

- 'Stilt', 90
- 'Larry', 85

List the resulting membership of club `c`.

```
'Wilt', 'Walt', 'Stilt'
```

5.2 Implementation

Fill in the definition of the `Club` class. Use a `list comprehension` in the `averageScore` method.

```
class Club:
    def __init__(self, firstMember, scoreFunction):
        self.members = [firstMember]
        self.scoreFunction = scoreFunction

    # Returns average score of current members.
    def averageScore(self):

        n = float(len(self.members))
        return sum([self.scoreFunction(m) for m in self.members])/n


    # Adds member if it meets the criterion.  Returns True if the
    # member was added and False, if not.
    def proposeMember(self, member):

        if self.scoreFunction(member) > self.averageScore():
            self.members.append(member)
            return True
        else:
            return False
```


5.3 Histogram

Write a procedure to compute a histogram of the member scores, that is, a count of how many members' scores fall within some specified ranges. We specify ranges by a list of N upper bounds. For example, the following bound list ($N = 3$):

```
[3, 9, 12]
```

specifies the following $N + 1 = 4$ ranges:

```
x < 3, 3 <= x < 9, 9 <= x < 12, 12 <= x
```

Given a list of scores that is already sorted from smallest to largest, such as: `[1, 1, 4, 5, 7, 15]` the resulting histogram would be: `[2, 3, 0, 1]` That is, 2 scores less than 3, 3 scores between 3 and 9, 0 scores between 9 and 12 and 1 score above 12.

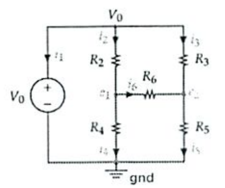
```
>>> histogram([1, 1, 4, 5, 7, 15], [3, 9, 12])
[2, 3, 0, 1]
```

The output should always have $N + 1$ values; values should be zero if there are no scores in the appropriate range.

```
def histogram(scores, bounds):

    counts = []                              # answer list
    index = 0                                # index into scores
    n = len(scores)                          # total no of scores
    for x in bounds:                         # for each bound
        count = 0                            # initialize count
        while index < n and scores[index] < x:
            count += 1                       # increment count
            index += 1                       # move to next score
        counts.append(count)                 # store count in answer
    counts.append(n-index)                   # count any remaning scores
    return counts
```

1 Short-Answer Questions (10 points)

Part a. Consider the following circuit.



Determine if the following equations and/or statements are

- **Always True** – i.e., true for all possible values of the resistors $R_2 - R_6$ and voltage V_0

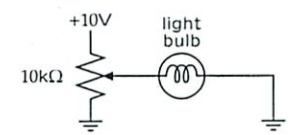
or

- **NOT Always True** – i.e., false for some or all resistor and voltage values.

Check the appropriate box for each of the following:

NOT Always True	Always True	
☐	☒	If $\frac{R_2}{R_4} = \frac{R_3}{R_5}$ then $i_6 = 0$
☐	☒	$i_2 + i_3 = i_4 + i_5$
☒	☐	$i_2 + i_6 = i_3$
☒	☐	$e_1 = \frac{R_4}{R_2 + R_4} V_o$
☐	☒	If $i_6 = 0$ then $\frac{R_2}{R_2 + R_4} = \frac{R_3}{R_3 + R_5}$

Part b. Our goal is to design a dimmer for a light bulb that can be modelled as a constant resistor of 10Ω. We have a single 10V power supply. Our first design uses a 10kΩ potentiometer, with the goal of controlling the current through the lamp so that the current is **proportional** to the potentiometer setting, with a maximum current of 1A.



Briefly (using fewer than 50 words) describe problems with this circuit.

> If the potentiometer is turned nearly all the way to the +10V side, then the current through the light bulb will be near 1A. Otherwise, the resistance of the potentiometer will limit the current through the light bulb to < 1mA. This the potentiometer will work more like an on/off switch than like a dimmer.

Suggest a better circuit. Draw it in the following box.

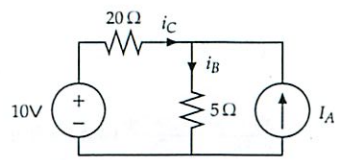
> potentiometer -> unity buffer -> bulb -> ground

Explain briefly why your circuit is better.

> The current into the op-amp buffer will be nearly zero. Therefore, the potentiometer will act as a voltage divider, and its output voltage will be proportional to the angle of its shaft. The op-amp buffer will drive its output to match the potentiometer voltage. Since we are modelling the light bulb as a resistor, its current will be proportional to the potentiometer shaft angle.

6 Circuits (20 points)

Consider the following circuit.



Part a. Find i_C if $I_A = 0$.

$i_C =$ 0.4A

$$i_C = \frac{10\text{V}}{20\Omega + 5\Omega} = \frac{2}{5}\text{A} = 0.4\text{A}$$

Part b. Find i_B if $I_A = 5\text{A}$.

$i_B =$ 4.4A

$$i_B = \frac{10\text{V}}{20\Omega + 5\Omega} + \frac{20\Omega}{5\Omega + 20\Omega} \times 5\text{A} = 4.4\text{A}$$

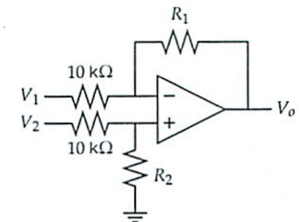
Part c. Find I_A so that $i_C = 0$.

$I_A =$ 2A

$$i_C = 0 = \frac{10\text{V}}{20\Omega + 5\Omega} - \frac{5\Omega}{5\Omega + 20\Omega} \times I_A = \frac{2}{5}\text{A} - \frac{1}{5}I_A$$

$I_A = 2\text{A}$

Part e. Consider the following op-amp circuit.



Fill in the values of R_1 and R_2 required to satisfy the equations in the left column of the following table. The values must be non-negative (i.e., in the range $[0, \infty]$). If the equation is impossible to implement with non-negative resistors, then write "impossible" for both resistor values.

	R_1	R_2
$V_o = 2V_2 - 2V_1$	20kΩ	20kΩ
$V_o = 2V_2 - V_1$	10kΩ	∞
$V_o = V_2 - 2V_1$	20kΩ	5kΩ
$V_o = 4V_2 - 2V_1$	20kΩ or impossible	impossible

$$V_+ = \frac{R_2}{10\text{k}\Omega + R_2} V_2 = V_- = \frac{R_1}{10\text{k}\Omega + R_1} V_1 + \frac{10\text{k}\Omega}{10\text{k}\Omega + R_1} V_o$$

$$V_o = \frac{10\text{k}\Omega + R_1}{10\text{k}\Omega + R_2} \times \frac{R_2}{10\text{k}\Omega} \times V_2 - \frac{R_1}{10\text{k}\Omega} \times V_1$$

Solving the first and third rows requires just simple algebra.

Solving the algebra for the fourth row suggests that R_2 is negative. Therefore, this condition cannot be realized with non-negative resistors.

Solving the second row is a bit more tricky, since the algebraic solution suggests

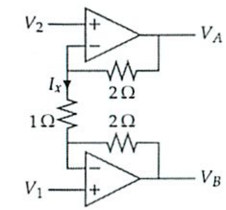
$$2R_2 = 2R_2 + 20\text{k}\Omega$$

which can only be satisfied in the limit as $R_1 \to \infty$.

NanoQuiz 10 Makeup

Name:

Part 1: Op Amps. Assume the op-amps in the following circuit are 'ideal.'



Determine the current I_x when $V_1 = 1\,\text{V}$ and $V_2 = 2\,\text{V}$.

$I_x =$ 1A

Determine the voltage V_A when $V_1 = 1\,\text{V}$ and $V_2 = 2\,\text{V}$.

$V_A =$ 4V

Determine a general expression for V_A in terms of V_1 and V_2.

$V_A =$ −2 $* V_1 +$ 3 $* V_2$

The plus and minus input voltages must be equal for each op-amp. Therefore, the voltage across the 1Ω resistor is $V_2 - V_1$, and the current through the 1Ω resistor is

$$I_x = \frac{V_2 - V_1}{1\Omega} = \frac{2-1}{1} = 1\text{A}.$$

The voltage V_A equals the voltage at the minus input of the top op-amp plus the voltage drop across the top 2Ω resistor. Since the current into the minus input to the op-amp is zero, then the same current (I_x) flows through the 2Ω resistor as well. Therefore

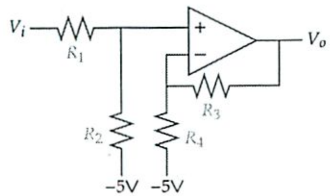
$$V_A = V_2 + I_x \times 2\Omega = 2 + 1 \times 2 = 4\text{V}$$

More generally,

$$V_A = V_2 + I_x \times 2\Omega = V_2 + \frac{V_2 - V_1}{1\Omega} \times 2\Omega = V_2 + 2(V_2 - V_1) = -2V_1 + 3V_2$$

... continued on back of page

Part 2: Op Amps. Consider the following circuit:



where all of the resistors have the same value $R = 1\,\text{k}\Omega$.

If $V_i = 3\,\text{V}$, then $V_o =$ 3V

If $V_i = 7\,\text{V}$, then $V_o =$ 7V

If $V_i = 9\,\text{V}$, then $V_o =$ 9V

Because no current flows into the positive input to an op-amp, all of the current through R_1 flows through R_2. Therefore the voltage drops across these resistors are equal. It follows that the voltage V_+ is half way between V_i and $-5V$:

$$V_+ = \frac{V_i - 5}{2}$$

Similarly, the voltage V_1 is half way between V_o and $-5V$:

$$V_- = \frac{V_o - 5}{2}$$

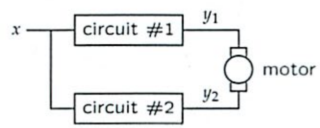
Since $V_+ = V_-$, it follows that $V_o = V_i$.

3. Circuits (30 / 100 points)

Motor driver

When we built the robot head, we made the motor move in both directions by connecting one side of the head motor to an op amp circuit and the other side to a buffered voltage divider that produced +5V. This method limited the peak speeds of the motor because the full +10V that is available from the power supply never appeared across the motor.

Our goal is to build two circuits, one to drive each of the two motor wires. Let x represent the input voltage and let y_1 and y_2 represent the voltages applied to the two motor wires. Assume that you have a single 10-volt power supply, so that only +10V and 0V are available.



Question 13: **Recall the supply voltage constraints: ALL voltages (including x, y_1 and y_2) must be between 0V and +10V.** Determine expressions for y_1 and y_2 so that the voltage across the motor is given by

$$y_1 - y_2 = \begin{cases} 10 & \text{if } x = 10 \\ -10 & \text{if } x = 0 \end{cases}$$

and both y_1 and y_2 have the form $y_i = m_i x + b_i$, where m_i and b_i are constants.

Solution: Since the voltages are limited to the range [0,10], there is only one way to make $y_1 = y_2 = 10$: $y_1 = 10$ and $y_2 = 0$. Similarly, to make $y_1 = y_2 = -10$: $y_1 = 0$ and $y_2 = 10$. Thus y_1 ramps from 0 to 10 as x ramps from 0 to 10, so that

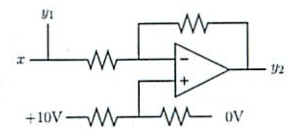
$$y_1 = x$$

and y_2 ramps down from 10 to 0 as x ramps from 0 to 10, so that

$$y_2 = 10 - x.$$

Question 14: Design circuits to implement your solutions to question 14 using resistors, op amps, and a single +10V power supply. You can assume that x is buffered (i.e., it is the output of an op amp). Draw the circuits and label them clearly.

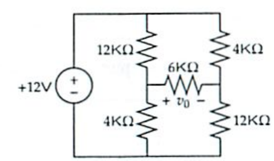
Solution: The y_1 output is equal to the x input (which is buffered), so y_1 can be connected to x with a wire. The y_2 output decreases as x increases, which can be implemented with an inverting amplifier. To get the endpoints right (e.g., x = 0 maps to $y_2 = 10$ and x = 10 maps to $y_2 = 0$) we must make the positive input to the op amp be 5V, which can be obtained with a voltage divider. The following figure shows the result:



where all of the resistors have equal values (e.g., 10KΩ).

Analysis

A "bridge" circuit consisting of five resistors is connected to a 12 volt source as shown below.



Bridge circuit

6.01 Midterm 2: Spring 2010

Name:	Section:

Solutions: Not correct for the make-up exam.

Enter all answers in the boxes provided.

During the exam you may:

- read any paper that you want to
- use a calculator

You may not

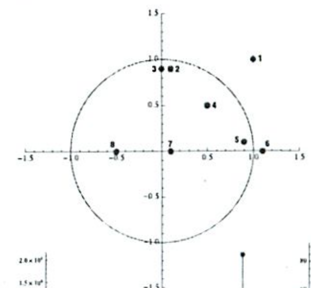
- use a computer, phone or music player

For staff use:

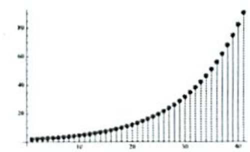
1.	/16
2.	/16
3.	/8
4.	/25
5.	/5
6.	/8
7.	/12
8.	/10
total:	/100

1 Pole Position (16 points)

The polar plot shows the dominant pole for several systems. Match each pole to the unit sample response of the system.



A = 1

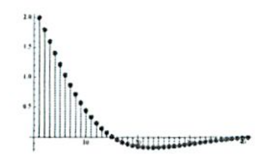


B = 6

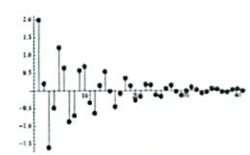
C = 7

D = 8

E = 3



F = 5



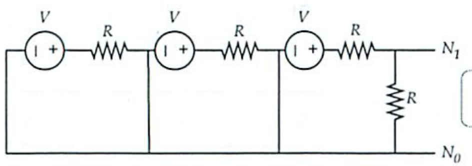
G = 2

H = 4

2 What's the difference? (16 points)

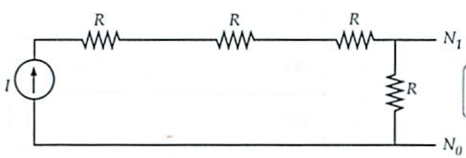
Assuming the voltage at node $N_0 = 0$, compute the voltage at node N_1 in each of these circuits.

2.1



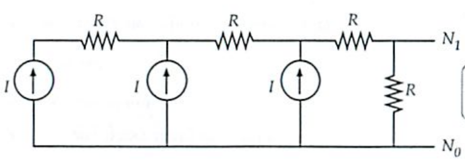
V/2

2.2



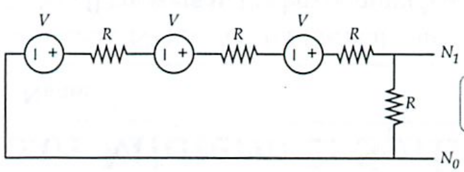
IR

2.3



3IR

2.4



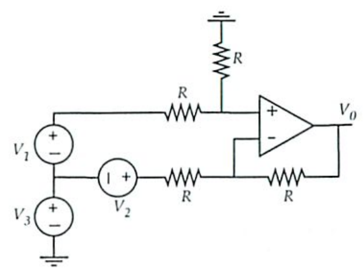
3V/4

Scratch Paper

3 Op Amps (8 points)

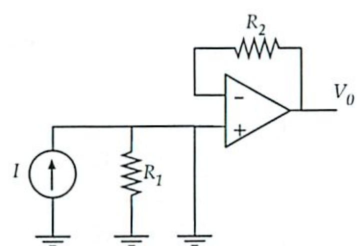
For each of the following circuits, give the output voltage V_O as a function of the input voltage sources, input current sources, and resistances. Assume the op-amps are ideal.

3.1



$V_1 - V_2$

3.2



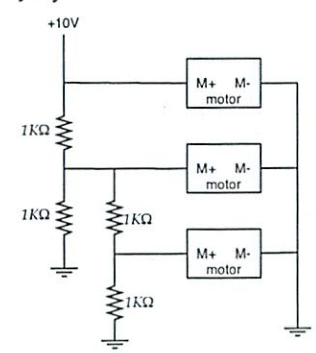
0

4 Motor control (25 points)

WhizzyLand engineers Kim, Pat, Jody, Chris, and Jamie are trying to design a controller for a display of three dancing robotic mice, using a 10V power supply and three motors. The first is supposed to spin as fast as possible (in one direction only), the second at half of the speed of the first, and the third at half of the speed of the second.

Each engineer has come up with a design, as shown below. Assume the motors have a resistance of approximately 5Ω and that rotational speed is proportional to voltage. For each design, indicate the voltage across each of the motors. Your answer only needs to be within 0.5V of the correct answer.

4.1 Jody

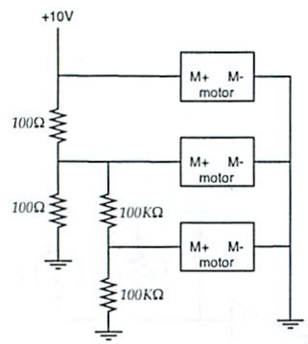


10

0.05

0

4.2 Chris

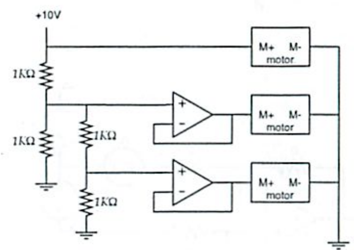


10

0.45

0

4.3 Pat

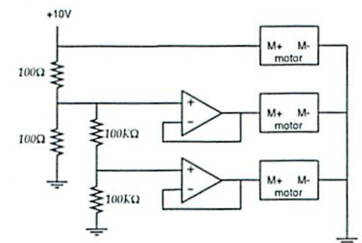


10

4

2

4.4 Kim

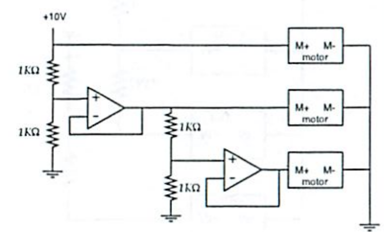


10

5

2.5

4.5 Jamie



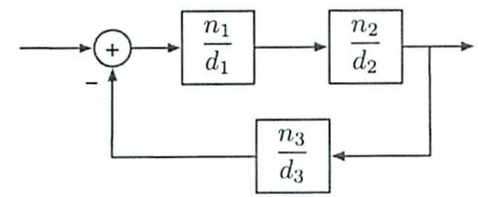
10

5

2.5

Scratch Paper

5 Composition (5 points)



Write an expression for the system function for this whole system, in terms of $n_1, d_1, n_2, d_2, n_3, d_3$, which are the numerator and denominator polynomials of the system function for each of the component systems.

$$\frac{d_3 n_1 n_2}{d_1 d_2 d_3 + n_1 n_2 n_3}$$

6 OOP (6 points)

If a procedure has no side effects but may be used multiple times, then a convenient concept for efficiency is called memoization – we keep track of whether the procedure has previously been called with a particular argument, and the value computed at that time. If it has already been computed, we just return the saved value, otherwise we do the computation, store away the value for future use, and return the value.

Here is an object oriented approach for memoization, with one piece missing. We assume that proc, the procedure to be "memoized", has a single argument.

```
class Memo:
    def __init__(self, proc):
        self.proc = proc
        self.history = {}
    def val(self,arg):
        if arg in self.history:
            print 'found', arg, 'in history'
            return self.history[arg]
        else:
            # YOUR ANSWER HERE
```

An example usage would be

```
>>> def sq(x): return x*x
>>> test = Memo(sq)
>>> test.val(2)
4
>>> test.val(3)
9
>>> test.val(3)
found 3 in history
9
>>> test.val(8)
64
>>> test.val(5)
25
>>> test.history
{8: 64, 2: 4, 3: 9, 5: 25}
```


6.1

Complete the definition of this class, by supplying the code for the else part of the val method. Make sure that you do not execute the procedure more than once for any given argument.

```
self.history[arg] = self.proc(arg)
return self.history[arg]
```

6.2

The approach above works with any procedure of one argument that does not have side-effects. However, if we know that the procedure has some additional structure, we can be even more efficient. Assume we know that our procedure takes a single number as an argument and that the procedure is **even**, that is, `proc(arg) == proc(-arg)`. Examples of even procedures are abs, `math.cos` and `sq` as defined above.

Write a definition for the `MemoEven` class. Your solution must use inheritance, and ensure that there is a `val` method defined. Do not change the definition of `Memo` or repeat any of the code it contains.

```
class MemoEven(Memo):
    def val(self, arg):
        return Memo.val(self, abs(arg))
```

7 Red vs Blue (12 points)

Let's build a voting tabulator for the national elections (except that we're going to assume that there are only 4 states). The definitions below initialize the instance and allow us to enter votes, by state, for each candidate.

We're going to keep the votes data in the dictionary `stateVotes`. There is an entry in `stateVotes` for each state. The value stored for a state is itself a dictionary that stores the votes from that state for each candidate. A simple example, for two states (`'MA'` and `'TX'`) and two candidates, `'O'` and `'M'`, would be:

```
{'MA': {'O':100, 'M':50}, 'TX': {'O':50, 'M':100}}
```

```
class Votes:
    def __init__(self):
        self.states = ['MA', 'TX', 'CA', 'FL']
        self.candidates = ['O', 'M', 'N']
        self.stateVotes = {}
    def addVotes(self, cand, votes, state):
        if not state in self.stateVotes:
            self.stateVotes[state] = {cand : votes}
        else:
            if not cand in self.stateVotes[state]:
                self.stateVotes[state][cand] = votes
            else:
                self.stateVotes[state][cand] += votes
```

7.1

Write a method for the `Votes` class, called `popularVotesTotal`, that computes the total number of votes for a candidate, across all states.

- Use a list comprehension, not a `for` loop.
- Use the Python `sum` procedure, which takes a list of numbers as input and returns the sum.

```
def popularVotesTotal(self, cand):
    return sum([self.stateVotes[state][cand] for state in self.states])
```


7.2

Write a method for the `Votes` class, called `stateWinner`, that computes the winner for a state, that is, the candidate with the most votes in that state.

Use the procedure `argmaxDict(d)`, which is called with a dictionary `d` and returns the key in `d` whose associated value is highest.

```
def stateWinner(self, state):
    return argmaxDict(self.stateVotes[state])
```

7.3

Write a method for the `Votes` class, called `statesWon`, that takes a candidate as an argument and returns a list of the states that candidate won. Use the `stateWinner` method you just implemented.

```
def statesWon(self, cand):
    return [state for state in self.states if self.stateWinner(state) == cand]
```

7.4

Write a method for the Votes class, called winnerOfMostStates, that returns the candidate that won the most states. Use the statesWon method you just implemented. For full credit, use util.argmax.

If l is a list of items and f is a procedure that maps an item into a numeric score, then util.argmax(l, f) returns the element of l that has the highest score.

```
def winnerOfMostStates(self):
    return util.argmax(self.candidates,
                       lambda cand: len(self.statesWon(cand)))
```


8 Balanced Machine (10 points)

Write a state machine that counts the depth of nesting of parens in a string. It takes a character from the string as input and outputs an integer, indicating how many unmatched open parens there are (on or) before this character or None if the parens are unbalanced. Note that once the parens become unbalanced, all future outputs will be None. Here are some examples:

```
>>> bp = BalancedParens()
>>> bp.transduce('(ab (c (d (e)) f))')
[1, 1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 3, 2, 2, 2, 1, 0]
>>> bp.transduce('(())  ((()))  ()()()')
[1, 2, 1, 0, 0, 0, 1, 2, 3, 2, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0]
>>> bp.transduce('(()))  ()()()')
[1, 2, 1, 0, None, None, None, None, None, None, None, None, None]
```

Your state machine should be a subclass of sm.SM.

```
class BalancedParens(sm.SM):
    startState = 0
    def getNextValues(self, state, inp):
        if state == None:
            return (None, None)
        elif inp == '(':
            return (state+1, state+1)
        elif inp == ')':
            if state == 0:
                return (None, None)
            else:
                return (state-1, state-1)
        else:
            return (state, state)
```