LRC - Instructable

```
# LRC - Instructable  (  version: 2014/03/03 )
#
#           This contains the code for the instructable:
#
#           Incredibility Powerful Resistance Calculator by russ_hensel
#
#           The code computes series/parallel properties of various inductors,
resistors, and capacitors.
#           The worksheet can be downloaded from
#                 Instructables, as part of a larger zip file:   <url>
#
#
#     Intended Use:
#           Read and tinker around with it, you may want to copy and rename so you
still have
#           an original to mess with.
#     Version:    March 3, 2014  Status: Done
#                 Built and tested on Sage 5.1 Running under Virtual Box on Windows 7
#
#     Estimated Minimum Level Useful for Understanding the Worksheet
#            SageMath  - beginner
#            Electronics - basic knowledge
#
#     Possibly useful references ( Some are more advanced than the material in this
worksheet )
#     Some related files:
#
#           LRC - Examples
#
#     Authors:   http://www.instructables.com/member/russ_hensel/  ( contact for
comments, additions, or problems )
#
```

```
# Explaining and Understanding the code in the next cell is not part of
#  the instructable, but feel free to read as you wish
# for the instructable, just skip to the end of the next cell and execute it.
# the next cell has a copy of the LRC code
```

```
```

LRC defined

```
```

```
# For the first calculation we will get the total resistance for a resistor
# of 1 K ohms in series with one of 10 K ohms
# ( and yes I know if you know much electronics you can do this in your head )
print "First Calculation - Add 1K resistor to 10K resistor in series:"
print

# Step 1
# make a resistor which is the calculator,  LRC stands for Inductance, Resistance,
# Capacitance, and is # used because the calculator can do all of them.
# I will use the long name “aResistor” to remind you what it stands for,
# but you could use just “r”
# This next line creates a "aResistor" of no value ( technically with a value of None
)

# Step 1
aResistor = LRC()
print "ignore the print out about frequency, this is only used in more advanced
calculations"

# Step 2
# we now add a new resistance to our "resistor"

aResistor.add_series_r( 1000 )  # add a 1 k ohm resistor
```

```
# this will cause a output that tells what we did

# Step 3
# now add the second resistance, in series with the first

aResistor.add_series_r( 10000 ) # 10000 = 10K

# this again will cause a output that tells what we did


# Step 4
# get the final value for the resistance ( note that z is a general symbol for
resistance )

print
print "Final value of combined resistance = ", aResistor.get_z( )

# shows the current value for the resistance, just the sum of resistances

# final comment suppresses default print at end of cell
```

```
First Calculation - Add 1K resistor to 10K resistor in series:

LRC() using internal frequency lrc_freq in Hz
ignore the print out about frequency, this is only used in more
advanced calculations
LRC.add_series_r() 1000
LRC.add_series_r() 10000

Final value of combined resistance =  11000
```

```
# next cell shows same calculation without most of the comments
# and prints, shows that calculations may be shorter than they
# seem in my more verbose examples.
```

```
aResistor = LRC()
aResistor.add_series_r( 1000 )
```

```
aResistor.add_series_r( 10000 )
print "Final value of combined resistance = ", aResistor.get_z( )
# final comment suppresses default print at end of cell
```

```
LRC() using internal frequency lrc_freq in Hz
LRC.add_series_r() 1000
LRC.add_series_r() 10000
Final value of combined resistance =  11000
```

```
print "Second Calculation - Add 1K resistor to 10K resistor in parallel:"
print

# -----
# make a "aResistor" which is the calculator

aResistor = LRC()

# -----
# we now add a resistance to our "resistor"

aResistor.add_parallel_r( 1000 )

# this will cause a output that tells what we did, and the current
# value for the impedance = resistance

# -----
# now add the second resistance, in parallel with the first

aResistor.add_parallel_r( 10000 )

# this again will cause a output that tells what we did, and the

# -----

print
print "Final value of combined resistance = ", aResistor.get_nz( )
#  nz in function above pushes full numeric evaluation to a decimal value
```

```
# thi comment suppress end of cell default print
```

```
Second Calculation - Add 1K resistor to 10K resistor in parallel:

LRC() using internal frequency lrc_freq in Hz
LRC.add_parallel_r() 1000
LRC.add_parallel_r() 10000

Final value of combined resistance =  909.090909090909
```

```
# A More Complicated Circuit
# I will give the schematic in ascii characters -- not part of the calculation,
# just to help you understand
# lots of print statements, not necessary, just to help explain what is going on
#
print "Third Calculation - example of series and parallel resistor combination:"
print
print "Calculate resistance from x to x"
print
print "    |-----------1.5K-------------|"
print "    |                            |"
print "x---|-----------1.5K-----==------|------1.5k------- 1.5K------x"
print "    |                            |"
print "    |-----------1.5K-------------|"
print

# make the resistor calculator
print "Begin..."
aResistor    = LRC(  )
print

print "Do the parallel resistors..."

print
print "    |-----------1.5K-------------|"
print "    |                            |"
print "x---|-----------1.5K-----==------|--"
print "    |                            |"
```

```
print "    |-----------1.5K-------------|"
print

aResistor.add_parallel_r( 1.5e3 )   # 1.5e3 is scientific notation, a shorter way of
writing 1500
aResistor.add_parallel_r( 1.5e3 )
aResistor.add_parallel_r( 1.5e3 )

print "and now the two series reisistors "
print
print "      |----------1.5k----------1.5K------|"
print
print

aResistor.add_series_r( 1.5e3 )
aResistor.add_series_r( 1.5e3 )

print
# done but a final step using n()
print "Final value of combined resistance = ", aResistor.get_nz(  )

# suppress end of cell default print
```

```
Third Calculation - example of series and parallel resistor
combination:

Calculate resistance from x to x

    |-----------1.5K-------------|
    |                            |
x---|-----------1.5K-----==------|------1.5k------- 1.5K------x
    |                            |
    |-----------1.5K-------------|

Begin...
LRC() using internal frequency lrc_freq in Hz

Do the parallel resistors...
```

```
     |-----------1.5K-------------|
     |                            |
 x---|-----------1.5K-----==------|--
     |                            |
     |-----------1.5K-------------|

 LRC.add_parallel_r() 1500.00000000000
 LRC.add_parallel_r() 1500.00000000000
 LRC.add_parallel_r() 1500.00000000000
 and now the two series reisistors

       |----------1.5k----------1.5K------|


 LRC.add_series_r() 1500.00000000000
 LRC.add_series_r() 1500.00000000000

 Final value of combined resistance =  3500.00000000000
```

```
print "Without much explanation get the formula for 2 resistors in parallel"
print "the calculator can even do algebra and plotting with formulas."
print "more of that in an advanced worksheet see files attached to instructable"

# when we do things symbolically we need to define our symbols

var("r1")    # symbol for resistor 1
var("r2")    # symbol for resistor 2
var("r3")    # symbol for resistor 3 the combination

lrc    = LRC()

print
lrc.add_parallel_r( r1 )
lrc.add_parallel_r( r2 )

print
print "final formula is: "
```

```
show( lrc.get_z() )  # show is similar to print but nicer output
```

```
Without much explanation get the formula for 2 resistors in parallel
the calculator can even do algebra and plotting with formulas.
more of that in an advanced worksheet see files attached to
instructable
LRC() using internal frequency lrc_freq in Hz

LRC.add_parallel_r() r1
LRC.add_parallel_r() r2

final formula is:
```

$$\frac{1}{\frac{1}{r_1} + \frac{1}{r_2}}$$

```
print "Repeat and extend symbolic calculation -- solve for r sub 1"
print "Again without much explanation get the formula for 2 resistors in parallel"
print

# when we do things symbolically we need to define our symbols

var("r1")    # symbol for resistor 1
var("r2")    # symbol for resistor 2
var("r3")    # symbol for resistor 3 the combination

lrc    = LRC()

lrc.add_parallel_r( r1 )
lrc.add_parallel_r( r2 )


print
print "so the two in parllel are: ",  lrc.get_z()
print
```

```
print "Now do some algebra and turn the calculation into an equation for r3"
print "then solves for r1 in terms of r3 and r2"

# I am not explaining this, but it is just SageMath, look in web references

equation = ( r3 == lrc.get_z() )
print equation

print
print "solving..."
solution = equation.solve( r1 )
print( solution )  # show( solution ) gives nicer output, try it
```

```
Repeat and extend symbolic calculation -- solve for r sub 1
Again without much explanation get the formula for 2 resistors in
parallel

LRC() using internal frequency lrc_freq in Hz
LRC.add_parallel_r() r1
LRC.add_parallel_r() r2

so the two in parllel are:  1/(1/r1 + 1/r2)

Now do some algebra and turn the calculation into an equation for r3
then solves for r1 in terms of r3 and r2
r3 == (1/(1/r1 + 1/r2))

solving...
[
r1 == r2*r3/(r2 - r3)
]
```