Introduction to Sage

Author	**Aaron Tresham**
Date	**2017-06-28T17:55:38**
Project	**a8975d68-235e-4f21-8635-2051d699f504**
Location	**01 - Intro to Sage Assignment/Intro to Sage Notes.sagews**
Original file	**Intro to Sage Notes.sagews**

Introduction to Sage

Sage is a powerful set of computational tools that allow you to use the computer for various applications in

- Precalculus (e.g., functions, graphing),
- Calculus (e.g., limits, derivatives, integrals),
- and beyond (e.g., Differential Equations, Linear Algebra).

This semester, we will just begin to learn some of the many features of Sage, and we will use Sage to help us understand calcu better.

This tutorial will introduce you to the basics of Sage.

Input and Output

There are two regions in a Sage worksheet: input cells and output cells.

Output cells have a green, vertical line next to them, while input cells do not. You type commands in an input cell, and you run them by pressing [SHIFT]-[ENTER] (or press the play button [with the triangle] at the top of the page). To get a new input cell, move your cursor to the gray horizontal line above or below any other input cell. A blue highlight will appear. Click and another ir cell will be inserted. Try it now.

An output cell will appear when you press [SHIFT]-[ENTER]. If any commands you have typed in the input cell produce an outp this will appear right below the input cell. If you click and drag, you can highlight multiple output cells and run them all by pressir [SHIFT]-[ENTER] once.

If an input cell has been run in the current session, then a small portion of the gray horizontal line above it will be blue. While inp code is running, part of the horizontal line above it will flash green. If input code is waiting to run, part of the line will flash red.

Basic Arithmetic

Sage can be used very much like your graphing calculator to do computations. Simply type an arithmetic expression in an input and press [SHIFT]-[ENTER]. Here are some examples.

```
1+1
```

```
2
```

```
109-87
```

```
22
```

```
22*93
```

```
2046
```

```
88/22
```

```
4
```

Exponents:

```
4^3
```

```
64
```

```
4^(1/2)
```

```
2
```

```

```

The same symbol is used for subtraction and negative numbers:

```
-9+22
```

```
13
```

```
3*-6
```

```
-18
```

You can type multiple inputs before running.

One way to do this is to hit [ENTER] after each line.

```
1+8
2-5
4*6
```

```
9
-3
24
```

The outputs are produced in the same order as the inputs.

Another way to do this is to put a semicolon between different inputs.

```
1+8; 2-5; 4*6
```

```
9
-3
24
```

If you want to access the last output, use _

```
3*4
_+5
```

```
12
17
```

In the example above, _ $= 12$, since that was the last output.

```
12-10
7*_
```

```
2
14
```

In this example, _ $= 2$, since that was the last output.

Order of Operations

Like a calculator, Sage uses the standard order of operations, often called PEMDAS (parentheses, exponents, multiplication, division, addition, subtraction). Compare the following:

```
2*4^5+3
```

```
2051
```

$$2 \cdot 4^5 + 3 = 2 \cdot 1024 + 3 = 2048 + 3 = 2051$$

21

```
(2*4)^5+3
```

```
32771
```

22

$$(2 \cdot 4)^5 + 3 = 8^5 + 3 = 32768 + 3 = 32771$$

23

```
2*(4^5+3)
```

```
2054
```

24

$$2 \cdot (4^5 + 3) = 2 \cdot (1024 + 3) = 2 \cdot 1027 = 2054$$

25

```
2*4^(5+3)
```

```
131072
```

$$2 \cdot 4^{5+3} = 2 \cdot 4^8 = 2 \cdot 65536 = 131072$$

Be careful with fractions:

26

```
12/3+5
```

```
9
```

27

$$\frac{12}{3} + 5 = 9$$

28

```
12/(3+5)
```

```
3/2
```

29

$$\frac{12}{3+5} = \frac{3}{2}$$

Complete Question 1 of the assignment.

Decimal Approximations

Unlike calculators, Sage tries to return exact answers instead of decimal approximations. If the input involves only integers, then Sage tries to keep integers in the output as well. For example, since 11 divided by 4 is not a whole number, Sage doesn't do anything:

```
11/4
```

```
11/4
```

Since 11 and 4 are both integers, Sage tries to keep everything as integers, so it returns the fraction $\frac{11}{4}$. If you want a decimal instead, Sage has a decimal approximation function called "N" (short for "numerical_approximation").

```
N(11/4)
```

```
2.75000000000000
```

Compare the following:

```
17/5
N(17/5)
```

```
17/5
3.40000000000000
```

This issue does not arise if you don't start with integers. For example 3.1/4.6 returns a decimal approximation:

```
3.1/4.6
```

```
0.673913043478261
```

Scientific Notation

For numbers that are very large or very close to 0, Sage employs scientific notation.

Recall, for example, $1.5 \times 10^4 = 15000$ and $1.5 \times 10^{-4} = 0.00015$.

In Sage, scientific notation is indicated using the letter "e" (for "exponent"). This is similar to graphing calculators that use "E."

Here we'll see the examples of 1.5×10^{25} and 1.5×10^{-25}.

```
1.5*10^(25)
```

```
1.50000000000000e25
```

37

```
1.5*10^(-25)
```

```
1.50000000000000e-25
```

This notation is similar to many graphing calculators.

When the number after the "e" is *postive*, move the decimal point that many spots to the *right* (adding extra zeros as necessary)

When the number after the "e" is *negative*, move the decimal point that many spots to the *left* (adding extra zeros as necessary)

Consider the example below.

```
1.25325428941968e-17
```

At first, this looks like a number around $1\frac{1}{4}$. But this is actually a number close to 0:

$$1.25325428941968\text{e-}17 = 1.25325428941968 \times 10^{-17} = 0.0000000000000000125325428941968\,.$$
$$\approx 0$$

Caution: This use of the letter "e" has nothing to do with the number $e \approx 2.71828$ (the base of the natural logarithm).

38

Standard Functions

Sage includes the standard functions you learned in precalculus, including trigonometric, exponential, and logarithmic functions.

Square root

The Sage command for square root is sqrt.

39

```
sqrt(2)
```

```
sqrt(2)
```

Note that Sage keeps it exact using integers. To convert to a decimal, use N().

40

```
N(sqrt(2))
```

```
1.41421356237310
```

However, notice that if you start with a decimal, Sage will output a decimal.

41

```
sqrt(2.5)
```

```
1.58113883008419
```

Compare with the following:

```
sqrt(5/2)
N(sqrt(5/2))
```

```
sqrt(5/2)
1.58113883008419
```

Other roots

If you want a root other than the square root, use a fractional power. For example, here's a cube root (note the parentheses).

```
2^(1/3)
N(2^(1/3))
```

```
2^(1/3)
1.25992104989487
```

Don't forget the parentheses around 1/3. This is *not* the same as 2^1/3.

```
2^1/3
N(2^1/3)
```

```
2/3
0.666666666666667
```

Here's a fourth root.

```
8^(1/4)
N(8^(1/4))
```

```
8^(1/4)
1.68179283050743
```

Trig functions

Notice how Sage keeps it exact again.

Note that pi = π.

Another note: Sage assumes all angles are measured in radians.

```
sin(3)
N(sin(3))
```

```
sin(3)
0.141120008059867
```

```
sin(pi)
sin(-pi/2)
```

```
0
-1
```

In the example below, note the 5*pi/4 - the multiplication must be explicit - Sage will not accept 5pi/4.

```
sin(5*pi/4)
sin(pi/12)
```

```
-1/2*sqrt(2)
-1/12*sqrt(6)*(sqrt(3) - 3)
```

Notice that Sage knows many exact values for the trig functions - even some that you did not memorize in precalc.

These answers are kind of hard to read. You can make them look nicer by using the "show" command.

```
show(sin(5*pi/4))
show(sin(pi/12))
```

$$-\frac{1}{2}\,\sqrt{2}$$

$$-\frac{1}{12}\,\sqrt{6}(\sqrt{3}-3)$$

```
cos(pi/7)
N(cos(pi/7))
```

```
cos(1/7*pi)
0.900968867902419
```

```
tan(1/2)
N(tan(1/2))
tan(0.5)
```

```
tan(1/2)
0.546302489843790
0.546302489843790
```

```
sec(2)
N(sec(2))
```

```
sec(2)
-2.40299796172238
```

```
csc(pi/4)
N(csc(pi/4))
```

```
sqrt(2)
1.41421356237310
```

```
cot(pi/3)
N(cot(pi/3))
```

```
1/3*sqrt(3)
0.577350269189626
```

```
show(cot(pi/3))
```

$$\frac{1}{3}\sqrt{3}$$

Inverse trig functions

Sage uses arcsin, arccos, and arctan, *not* $\sin^{-1}$, $\cos^{-1}$, and $\tan^{-1}$.

```
arcsin(.5)
arccos(.25)
arctan(pi/2)
```

```
0.523598775598299
1.31811607165282
arctan(1/2*pi)
```

Let's convert that last one to a decimal. Remember that _ accesses the last output.

```
N(_)
```

```
1.00388482185389
```

Exponential functions

For exponential functions, just use ^ as we did above.

The built-in function "exp" is the exponential base e; this is equivalent to "e^" (recall, $e \approx 2.71828$).

Notice Sage's use of exact values.

```
e^3
```

```
e^3
```

```
N(e^3)
```

```
20.0855369231877
```

```
exp(3)
```

```
e^3
```

```
10^6
```

```
1000000
```

```
3^(3/5)
N(3^(3/5))
3^0.6
```

```
3^(3/5)
1.93318204493176
1.93318204493176
```

Logarithmic functions

In Sage, the function "log" is the logarithm **base e**, *not* base 10 as on your calculator.

```
ln(e^2)
log(e^2)
```

```
2
2
```

```
log(10^3)
N(log(10^3))
```

```
log(1000)
6.90775527898214
```

Sage can also handle logarithms with other bases: log(x,b) = $\log_b x$

Here's $\log_{10} 10^3 = 3$

```
log(10^3,10)
```

```
3
```

Here's $\log_2 1800 \approx 10.81$. Notice Sage gives an exact answer in terms of log unless you ask otherwise.

```
log(1800,2)
N(log(1800,2))
```

```
log(1800)/log(2)
10.8137811912170
```

Explicit Multiplication

When we write mathematical expressions, we often omit the multiplication symbol. For example, when we write $10\cos(2)$ we mean $10 \cdot \cos(2)$.

When we write $3(4+8)$ we mean $3 \cdot (4+8)$.

Implicit multiplication like in the above examples is **not allowed** in Sage. Every multiplication must be explicitly typed using the asterisk * (Shift-8).

If we try to run 10cos(2) we get a "SyntaxError." [When you see lots of red in the output, that's an error of some sort. You can ignore everything at the beginning. The nature of the error is in the last line.]

```
93  10cos(2)

Error in lines 1-1
Traceback (most recent call last):
  File "/projects/a8975d68-235e-4f21-8635-2051d699f504/.sagemathcloud/sage_server.py", line 879, in
execute
    exec compile(block+'\n', '', 'single') in namespace, locals
  File "", line 1
    10cos(Integer(2))
        ^
SyntaxError: invalid syntax
```

We get an error if we try 3(4+8) as well. Sage thinks we are using function notation, but "3" can't be the name of a function, so w get a "TypeError."

```
94  3(4+8)

Error in lines 1-1
Traceback (most recent call last):
  File "/projects/a8975d68-235e-4f21-8635-2051d699f504/.sagemathcloud/sage_server.py", line 879, in
execute
    exec compile(block+'\n', '', 'single') in namespace, locals
  File "", line 1, in
TypeError: 'sage.rings.integer.Integer' object is not callable
```

Explicit multiplication is extremely important when you are multiplying binomials. Sage may not give you an error, but it will not give you the correct answer. Compare the following two examples:

```
95  (x+2)(x+1)

x + 3
```

```
96  (x+2)*(x+1)

(x + 2)*(x + 1)
```

Notice that Sage does something in the first example (no error indicated), but what it does is wrong (or, at least, not what we expect)! [When you do (x+2)(x+1) Sage actually substitutes x+1 for x in x+2.]

We know $(x+2)(x+1)=x^2+3x+2$, which is definitely not $x+3$.

The moral is:

Make sure you explicitly type every multiplication, every time - no exceptions!

On the other hand, don't put multiplication where it doesn't belong. For example, there is no multiplication involved in log(4). This **not** log*(4) - log is a function, and $\log(4)$ is function notation, just like $\sin(4)$ or $f(4)$ [for some function f].

Complete Question 2 of the assignment.

Defining your own functions

You can define new functions with whatever name you want (some characters are not allowed) by typing a function name follow by a variable name in parentheses, then an equal sign, then the formula for the function, and finally pressing [SHIFT]-[ENTER].

For example, let's define $f(x)=5x^2+5$.

Caution: Don't forget to type 5*x^2 - the multiplication must be explicit.

```
f(x)=5*x^2+5

```

Once a function has been defined (make sure you run it), then you can use that function in other cells.

Note: If you close a worksheet and return to it later, you may have to run the function definition again before using it.

```
f(2)
```

```
25
```

```
f(14)
```

```
985
```

```
f(a+1)
```

```
Error in lines 1-1
Traceback (most recent call last):
  File "/projects/a8975d68-235e-4f21-8635-2051d699f504/.sagemathcloud/sage_server.py", line 865, in
```

```
execute
    exec compile(block+'\n', '', 'single') in namespace, locals
  File "", line 1, in
NameError: name 'a' is not defined
```

Oops, we got an error: "name 'a' is not defined."

Sage automatically defines one variable called "x." If you want to use another variable, you have to declare it using the "var" command.

Declaring Variables

To declare a variable called "a," type %var a and press [SHIFT]-[ENTER]. You can now use the variable "a" anywhere in the worksheet.

```
%var a
f(a+1)
```

```
5*(a + 1)^2 + 5
```

Here we use a variable called "b."

```
%var b
f(b)
```

```
5*b^2 + 5
```

You can use new variables in function definitions without declaring them.

```
g(t)=t^3-2*t+4
g(3)
```

```
25
```

```
h(w)=sqrt(w)-w^2
h(4)
```

```
-14
```

You do not have to declare variables that are set equal to a number.

```
z=22
f(z)
```

```
2425
```

In the example above, we do not have to declare z, since z is not really a variable at all, it's just a different way of writing 22.

You also don't have to declare a variable that is set equal to other variables.

```
xyz=a+b
f(xyz)
```

```
5*(a + b)^2 + 5
```

Here xyz is just another name for a+b, and we already declared a and b above.

If we later declare xyz to be a variable, it will erase the previous assignment.

```
%var xyz
f(xyz)
```

```
5*xyz^2 + 5
```

You can also declare multiple variables at once - just separate with commas:

```
%var x1, x2, x3
f(x1)
g(x2)
h(x3)
```

```
5*x1^2 + 5
x2^3 - 2*x2 + 4
-x3^2 + sqrt(x3)
```

```
%var u, v, w
f(u)*g(v)/h(w)
```

```
-5*(v^3 - 2*v + 4)*(u^2 + 1)/(w^2 - sqrt(w))
```

Complete Question 3 of the assignment.

Adding Comments

If you want to add a comment to your input code, then type #. Anything after the # will be ignored when the input is run.

```
#This line will be ignored.
1+5-8
#If I leave off the # sign, Sage will return an error. It would try to interpret this text as a comma
```

```
-2
```

```
This line will not be ignored.
1+5-8
```

```
Error in lines 1-1
Traceback (most recent call last):
  File "/projects/a8975d68-235e-4f21-8635-2051d699f504/.sagemathcloud/sage_server.py", line 879, in
execute
    exec compile(block+'\n', '', 'single') in namespace, locals
  File "", line 1
    This line will not be ignored.
            ^
SyntaxError: invalid syntax
```

Dividing Input Cells

If you want to divide a single input cell into multiple cells (i.e., add a horizontal gray line between them), then put your cursor at t place you want to divide the cell and hit [CTRL]-[;]. A horizontal gray line will be inserted above the line with the cursor.

Try splitting the input cell below.

```
33+1

22-9
```

```
34
13
```

```

```

Dealing with Sage Worksheets

- If you want to **download, delete, or rename** a Sage worksheet, click on the check box to the left of the worksheet name the "Files" list. These options will appear right above the list of files. Or, if the worksheet is open, click on the blue box w a blue "i" in a white circle (it's on the left side of the page), and you will see these options.

- If you want to **create** a new Sage worksheet, click on the button with a + sign that says "New." Type the name you want the box, and click the button that says "SageMath Worksheet."

- If you want to **print** a worksheet, open it and click on the white button with a printer-shaped icon (it's next to the green "Save" button). This will produce an HTML file that you can open in any web browser for printing.

- If you want to **split your screen** (so you can view two parts of a worksheet at the same time), click on the split view button: it's between the button with the two arrows and the button with the letter "A." If you click once, it splits the screer horizontally; if you click twice, it splits the screen vertically; if you click a third time, it returns to normal.

- **When you are finished with Sage**, click on your name or email address (top right next to a gear icon), and then click "Si Out."

Auto Close Brackets

> By default, when you type an open parenthesis or other bracket, Sage automatically inserts a matching close bracket.
>
> However, when you already have a close bracket, if you try to insert another one before it, then it will not insert another bracket will assume you are typing the one that is already there). So if you want a second close bracket, you have to put it after the existing one.
>
> Personally, I get so irritated by this, that I have disabled auto close brackets. To do this, open the general settings by clicking on the gear icon with your name next to it (upper right). [Note: this is not the same as the project settings, which has a wrench icon the upper left.] Then you can uncheck the box with "auto close brackets." The next time you open a Sage worksheet, it will not automatically close brackets.
>
> P.S. I also disable the "Extra button bar," since I never use it and it just takes up space on my screen.

generated 2017-06-28T17:55:38 on CoCalc