1. Create a Sage Cloud account.
   (a) Start the Firefox or Chrome browser (Sage works poorly with IE).
   (b) Go to http://cloud.sagemath.com
   (c) “Create new account” using your VCU email address.
   (d) Click “Create New Project”, and call it Math 591.
   (e) Click “New”, then “Worksheets”, then call it s01.

2. Evaluate $900\times(1+0.06\times(90/365))$ to find $900(1 + 0.06(90/365))$. Click “Run” or shift-enter to evaluate.

3. Evaluate $2^{25}$ to find $2^{25}$.

4. Find $\frac{550 \times [1 + (1.05)^{-30}]}{0.05}$

5. Evaluate $\sqrt{8}$ to get an exact expression for $\sqrt{8}$.

6. Evaluate $\text{numerical\_approx}(\sqrt{8})$, or simply $n(\sqrt{8})$ to get an approximate expression for $\sqrt{8}$.

7. Evaluate “pi”. Find a decimal approximation for $\pi$. Find a decimal approximation for $2\pi$. Remember to type $2*\pi$.

8. Evaluate “e”. Then use $n(e,\text{digits}=7)$ to find a 7-digit approximation for $e$.

9. Find a 6-digit approximation for $e^3$

10. Find $\log_{10}$. What did Sage compute? Did Sage compute the base-10 log?

11. Evaluate $\text{plot}(x^{*3},-2,2)$ to sketch the graph of $x^3$ on the interval $(-2, 2)$.

12. Use Sage to sketch $\cos x$ on the interval $(-2\pi, 2\pi)$.

13. For any variable other than “x” you must tell Sage that you will use it as a variable. Evaluate `var("y")` to define “y” as a variable. Now evaluate `plot3d(x**2+y**2-2, (-1,1),(-1,1))` to sketch $g(x) = x^2 + y^2 - 2$ for $-1 \leq x \leq 1$ and $-1 \leq y \leq 1$.

14. Sage is written in Python. Type in the following program and evaluate.

```python
def write_string(string_name):
    print string_name
```

Now type `write_string("hello world!")` and evaluate.

In order to do sophisticated calculations, or to allow for multiple inputs, you will need to define procedures (also called functions). Our “hello world!” program was the first example. It included a `print` statement. Other program features, in almost any language, include conditional statements (if..then..) and loops.

15. Type in the following procedure definition and evaluate.

```python
# This function returns the absolute value of a number x
def absolute(x):
    if x>=0:
        return x
    else:
        return -x
```

16. Now test it. Evaluate `absolute(4)`, `absolute(-4)`. “#” is the comment symbol. Everything after “#” is ignored—and not evaluated.

```python
def abs_plus_five(x):
    return absolute(x)+5
```

17. You don’t have to add five, you can add any number by adding a parameter.

```python
def abs_plus(x,y):
    return absolute(x)+y
```

18. Now test it. Evaluate `abs_plus(4,5)`, `abs_plus(-4,5)`, `abs_plus(-4,23)`, etc.