Programming in Sage

Lalit Jain

February 20, 2012

Introduction

This is a phenomenally brief introduction to programming in Sage. For more details and functionality see: http://www.sagemath.org/doc/tutorial/programming.html

External Files

When programming in Sage, it is easier to write your program in a separate file than use the command line. To write a program, use `emacs` or some other editor, save the file with extension `.sage` then run the command,

```
sage file.sage
```

If you need to save the ouput of a program to a file, use the `>` operator.

```
sage file.sage > output
```

Using `>>` will append to the needed output file.

Loops and Control

If Statements

```
a = 10
if gcd(a,4) > 1:
    print "not coprime"
elif gcd(a,4)==2:
    print "both even"
else:
    print "coprime"
```

Loops

```
for i in range(5):
    print i^2
```

Note that the range function is extremely powerful. By default `range(a)` gives the list from 0 to a-1. In general, `range(a,b,d)` produces a list of numbers from a to b in steps of d.

Functions, Iteration

```
def plusk(n,k):
    return n+k
```

Don't forget the colon or indentation. You can make a default value for k by:

```
def plusk(n,k=7):
    return n+k
```

but this can be overridden.

Lists and Dictionaries

l = [1,2,3, "hello world"]	create a list
l[0]	the first element in the list
l.append(5)	add the element 5 to the list
len(l)	length of l
del l[i]	delete the i-th element of the list
`[a^2 for a in range(20)]`	makes a list of the squares up to 20
a[i:k]	gives the sublist of a between indices i,k
d = {3:2, 6:7, 2/3:"hiya"}	create a dictionary
d['a'] = 'b'	appends a new rule to the dictionary
d.clear()	empty a dictionary
d.keys()	a list of keys