Python vs Perl: Runnable Classes

def ...
test_x(self)
if '__name__' = '__main__'

```
#!/usr/bin/env python3.3

# Copyright 2012 by Douglas Sweetser, sweetser@alum.mit.edu
# Licensed under the Apache License, Version 2.0.

import argparse

'''Class Size
Used for pixel sizes used in layouts.
Author: sweetser@alum.mit.edu'''
class Size:

    def __init__(self, s=0):
        self.s = s

    def simple_print(self):
        result = str(self.s)
        print(result)
        return result

    def pretty_print(self):
        result = "The size is: " + str(self.s) + "."
        print(result)
        return result

if __name__ == '__main__':
    args_parser = argparse.ArgumentParser(description='Prints
two bits of data')
    args_parser.add_argument('-s', '--sprint', action='store_
default=False, help="Simple print")
    args_parser.add_argument('-p', '--pprint', action='store_
default=False, help="Pretty print, more verbose")
    args_stuff = args_parser.parse_known_args()
    args = args_stuff[0]
    argv = args_stuff[1]

    if (not args.pprint):
        args.sprint = True

    while argv:
        s = argv.pop(0)
        foo_size = Size(s)
        if args.sprint:
            foo_size.simple_print()
```

my
package + sub new() = class
sub func = class.method
my ($self) = @_
-> for .

```
#!/usr/bin/env perl -w
$|++;

# Copyright 2012 by Douglas Sweetser, sweetser@alum.r
# Licensed under the Apache License, Version 2.0.

package Visualphysics::Layout::Size;

sub new {
    my $class = shift;
    my $self = {
        s => shift
    };
    bless $self, $class;
    return $self;
}

sub simple_print {
    my ($self) = shift @_;

    my $result = "$self->{s}";
    print ("$result\n");
    return $result;
}

sub pretty_print {
    my ($self) = shift @_;

    $result = "The size is: $self->{s}.";
    print("$result\n");
    return $result;
}
```

main() unless caller
sub main
while, shift
sub __get_opt

```
main() unless caller;

my $sprint;
my $pprint;

sub main {
    __get_opt();

    while(@ARGV) {
        $s = shift @ARGV;

        $foo = new Visualphysics::Layout::Size(

        if ($sprint) {
            $foo->simple_print();
        }
        if ($pprint) {
            $foo->pretty_print();
        }
    }
}

sub __get_opt {

    use Getopt::Long;
    $Getopt::Long::autoabbrev = 1;
    $Getopt::Long::ignorecase = 0;

    my $help;
    my $get = GetOptions(
              "sprint!" => \$sprint,
              "pprint!" => \$pprint,
              "help!" => \$help);

    die("Please check the options.\nProgram exiting.\n")
        unless $get;

    my $help_string = qq(

usage: Size.pm [-h] [-s] [-p]

Prints a point given two bits of data

optional arguments:
  -h, --help    show this help message and exit
  -s, --sprint  Simple print
  -p, --pprint  Pretty print, more verbose

);

    $sprint = 1 unless ($pprint);
}
```