

```

#!/usr/bin/env python

#####
# selfer.py
# Travis Hudson (2008.5.1)
# Pass in the name of a local Python file, and this program will go through all
# init methods, and do this transformation:
#
# __init__(self, param1, param2):
#     pass
#
# __init__(self, param1, param2):
#     self.param1 = param1
#     self.param2 = param2
#####

import os
import sys

# take the class line and return the class name
def find_class_name(class_line):
    result = class_line.split()[1]
    if (result[-1] == ":"):
        return result[0:-1]
    return result

# take the init line (" def __init__(self, ...)"), and return param names
def find_params(init_line):
    result = []
    init_line = init_line.strip()
    start_index = init_line.find(",") + 1
    cur_index = start_index

    # if this only contains the self param, return an empty array
    if (init_line.find(",") == -1):
        return []

    while (True):

        # if this is the last param
        if (init_line.find(",", cur_index) == -1):
            result.append(init_line[cur_index : init_line.find("(")].strip())
            break

        # there will be other params after this one
        else:
            result.append(init_line[cur_index : init_line.find(",", cur_index)].strip())
            cur_index = init_line.find(",", cur_index) + 1

    return result

#####
# Begin Main Execution
#####

# store the preferred indent spacing for this file
spacing = "    "

# initialize the list of classes changed
changed_classes = []

# store the name of the file to modify
file_name = None
try:
    file_name = sys.argv[1]
except:
    print("USAGE: selfer.py file.py")
    sys.exit()

# store every line in an array
f = open(file_name, "r")
all_lines = f.readlines()

# loop through the array, searching for init functions
i = 0
while (i < len(all_lines)):
    cur_line = all_lines[i]
    if (cur_line.rfind("__init__") != -1):
        all_params = find_params(cur_line)
        if (all_params != []):
            changed_classes.append(find_class_name(all_lines[i-1]))
            for cur_param in all_params:

```

```
        new_line = spacing + spacing + "self." + cur_param + " = " + cur_param + os.linesep
        all_lines.insert(i+1, new_line)
    i += 1

# write the array back out to the file
f.close()
f = open(file_name, "w")
f.write("")
f.close()
f = open(file_name, "a")
for cur_line in all_lines:
    f.write(cur_line)
f.close()

# report the classes changed
print "Classes changed:"
for i in changed_classes:
    print i
```