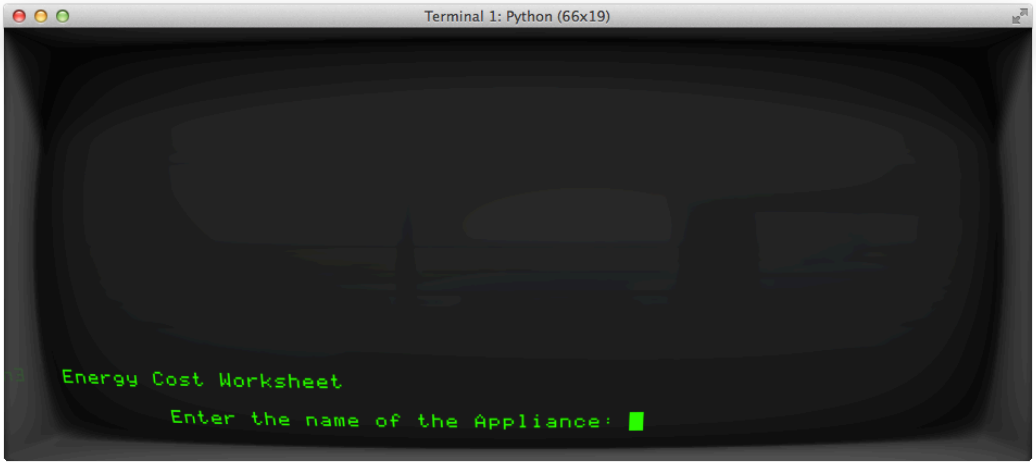
PROGRAM

```
############################################################
# PROJECT:      Lab 1 - Sequential Program Control:  Energy Cost
# ----------------------------------------------------------
# REQUIREMENT:  Write a program that calculates the annual cost
#               of running various appliances.  For each
#               appliance, the program will request that the
#               user enter the cost per kilowatt-hour and
#               the number of kilowatt-hours the appliance
#               uses in a year.
# ----------------------------------------------------------
# AUTHOR:       Brady Houseknecht
# BORN ON:      6/10/2014
# REVISION:     1.0
# ----------------------------------------------------------
# REVISION HISTORY:
# 1.0 Baseline - BH
############################################################
import os

g_title= 'Energy Cost Worksheet\n'
g_footer_prefix='\n\tThe total cost of the annual usage is $ '
g_footer=''
g_column_headers='\tappliance\tcost\t\t\tannual usage'
g_column_sub_headers='\t\t\t( per KW - hr )\t\t( KW - hr )'
g_row_buffer=''
g_total_cost=0.0
g_annual_price=0.0
g_annual_hours=0
g_annual_cost=0
g_appliance=''
g_command=''

def main():
            clear(80)
            write_title()
            #
            ## APPLIANCE/COST/USAGE I/O
            #
            prompt_appliance()
            prompt_cost()
            prompt_usage()
            #
            ## PROCESS INPUT
            #
            process()
            #
            ## WRITE SPREADSHEET TO STDOUT
            #
            write_spreadsheet(1)
            #
            ## COMMAND I/O
            #
            prompt_command()

def clear(num):
            for i in range(num): print

def prompt_appliance():
            global g_appliance
            g_appliance=str(raw_input("\tEnter the name of the Appliance: "))
            # TODO: Validate length of input, uniqueness

def prompt_cost():
            global g_annual_price
            g_annual_price=int(raw_input('\tEnter the cost in cents (per KW-hr): '))
            # TODO: Validate inputted value, non-negative, non-zero, integer

def prompt_usage():
            global g_annual_hours
            g_annual_hours=int(raw_input('\tEnter the annual usage (Kw-hr): '))
            # TODO: Validate inputted value, non-negative, non-zero, integer

def prompt_command():
            global g_command
            g_command=str(raw_input('\n\nx-Exit\ta-Add\n'))
            on_exit_command(g_command)
            on_add_command(g_command)
            write_spreadsheet(1)
            prompt_command()

def on_add_command(cmd):
            if cmd=='a':main()

def on_exit_command(cmd):
            if cmd=='x':
                        write_spreadsheet(0)
                        os._exit(1)

def calc_annual_cost():
            global g_annual_cost
            g_annual_cost=g_annual_price*g_annual_hours

def calc_total_cost(current_total):
            global g_total_cost
            g_total_cost=current_total+g_annual_cost

def add_buffer_row(buffer,app,cost,hours):
            global g_row_buffer
            g_row_buffer=buffer+"\n\t"+\
                 str(format(app,'16'))+\
                 str(format(cost,',d'))+" cents"+\
                 "\t\t"+\
                 str(format(hours,',d'))

def update_footer():
            global g_footer
            g_footer=g_footer_prefix+str(format(g_total_cost/100,',.2f')) + " ."

def write_row_buffer():
            print g_row_buffer
```

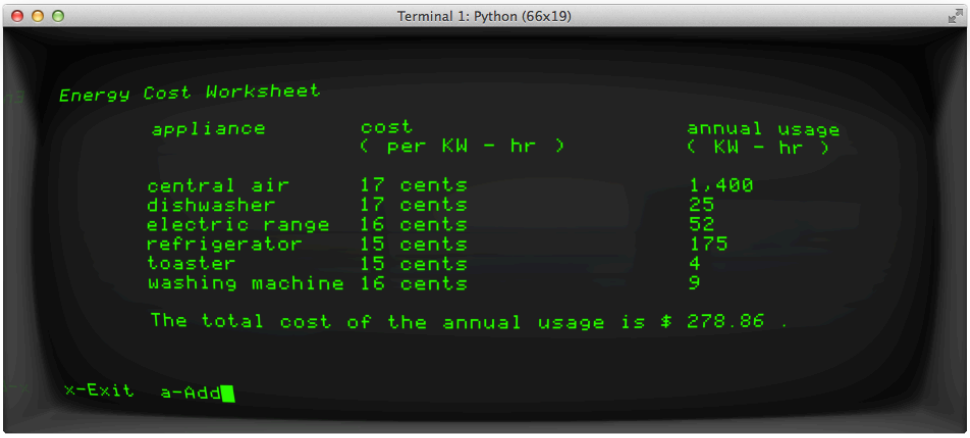
```
def write_footer():
            print g_footer+"\n"

def write_spreadsheet(menu):
            clear(80)
            write_title()
            write_header()
            write_row_buffer()
            if menu==1:write_footer()
            else:clear(4)

def process():
            calc_annual_cost()
            calc_total_cost(g_total_cost)
            add_buffer_row(g_row_buffer,g_appliance,g_annual_price,g_annual_hours)
            update_footer()

def write_title():
            print g_title

def write_header():
            print g_column_headers
            print g_column_sub_headers

main()
```

STARTUP / ADD MODE



On startup, the app clears the screen and enters “Add” mode. In this mode, the user is prompted for the appliance, kw-hour price and annual kw-hour usage.

SPREADSHEET MODE

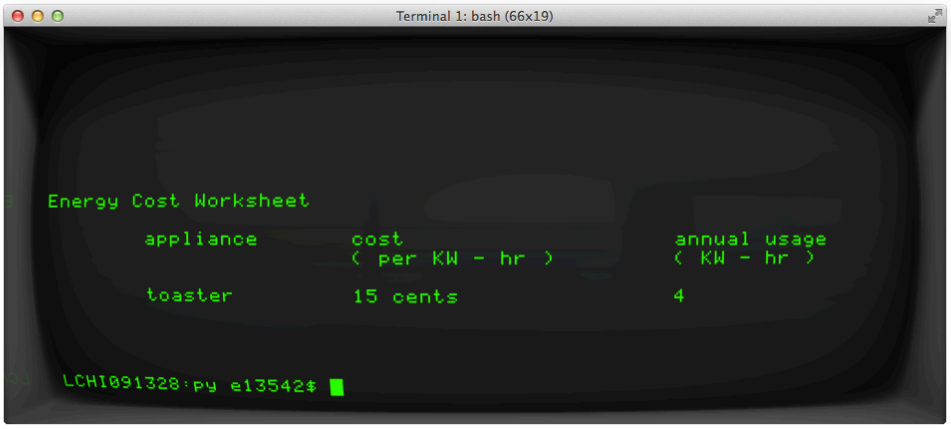


After the user completes adding a new appliance, the application enters "Spreadsheet" mode. In this mode, all data entered by the user is displayed as columnar report followed by a command menu. The user can enter:

- "x" to quit from the program
- "a" to add another appliance

The command input is case sensitive. If the user enters any invalid command key, the screen simply repaints.

EXIT MODE



When the user enters "x" (exit), the screen is repainted in spreadsheet mode minus the command menu.