MICRO PROJECT

INDEX

TOPIC

Display of length of time, the kernel has been running since boot and spent in idle mode.

GROUP MEMBERS

- CHRISTY THOMAS - FIT18CS041
- DEEPU CYRIAC - FIT18CS042
- DEVANA C U - FIT18CS043
- DEVNANDHAN S - FIT18CS044

INTRODUCTION

- In Windows NT operating systems, the System Idle Process contains one or more kernel threads which run when no other runnable thread can be scheduled on a CPU. In a multiprocessor system, there is one idle thread associated with each CPU core
- Because of the idle process's function, its CPU time measurement may make it appear to users that the idle process is monopolizing the CPU.
- Its CPU time "usage" is a measure of how much CPU time is not being used by other threads
- Uptime is a measure of system reliability, expressed as the percentage of time a machine, typically a computer, has been working and available
- It is often used as a measure of computer operating system reliability or stability, in that this time represents the time a computer can be left unattended without crashing, or needing to be rebooted for administrative or maintenance purposes

DESIGN

The program is coded using python language.Graphical User Interface(GUI) is designed using Tkinter.

- Since the uptime is a kernel internal value, which ticks up every cycle, it starts counting when the kernel has initialized. That is, when the first cycle has ended. Even before anything is mounted, directly after the bootloader gives control to the kernel image.
- From the proc manpage: /proc/uptime This file contains two numbers: the uptime of the system (seconds), and the amount of time spent in idle process (seconds). The proc filesystem contains a set of pseudo files. ... Every time you read a file, such as /proc/uptime , its contents are regenerated on the fly
- The first number is the uptime in seconds , the uptime command gives you the time for which the system has been up (or running)
- The second number represents, in seconds, how long the processor has been idle. On multi core systems, this is accumulated on a per CPU basis.

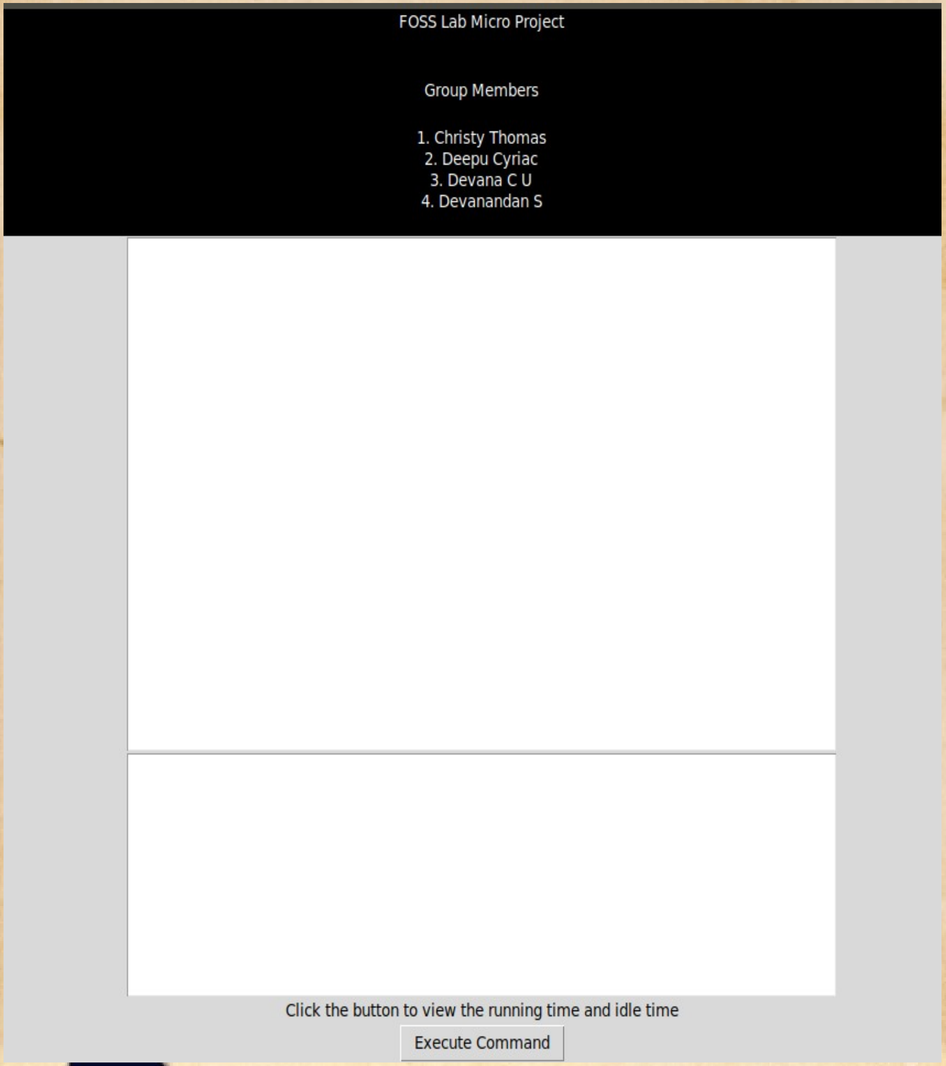
IMPLEMENTATION

Source Code:

```
from tkinter import *

root = Tk()

aframe = Frame(root, height=800, width=800)
aframe.pack()

headLabel = Label(aframe, text="FOSS Lab Micro Project\n\n", bg="black", fg="white", width=800)
headLabel.pack(fill=X)

h1Label = Label(aframe, text="Group Members\n", bg="black", fg="white")
h1Label.pack(fill=X)

nLabel = Label(aframe, text="1. Christy Thomas\n2. Deepu Cyriac\n3. Devana C U\n4. Devanandan S\n",
bg="black", fg="white")
nLabel.pack(fill=X)

bframe = Frame(root, height=800, width=800)
bframe.pack(side=BOTTOM)
```

```
def syscmd():
    import subprocess

    out = subprocess.Popen(['cat','/proc/uptime'], stdout=subprocess.PIPE, stderr=subprocess.STDOUT)

    stdout,stderr = out.communicate()

    b = "Running time \n"
    i = "Idle time \n"

    textbox1.insert(INSERT, "\n"+b)
    textbox2.insert(INSERT, "\n"+i)

    textbox1.insert(INSERT, str(stdout.split()[0])+"\n")
    textbox2.insert(INSERT, str(stdout.split()[1])+"\n")


btlabel = Label(bframe, text="Click the button to view the running time and idle time", fg="black",
width=800)
button1 = Button(bframe, text="Execute Command", command=syscmd)

btlabel.pack()
button1.pack()


textbox1 = Text(root)
textbox2 = Text(root)

textbox1.pack()
textbox2.pack()

root.mainloop()
```

Output:

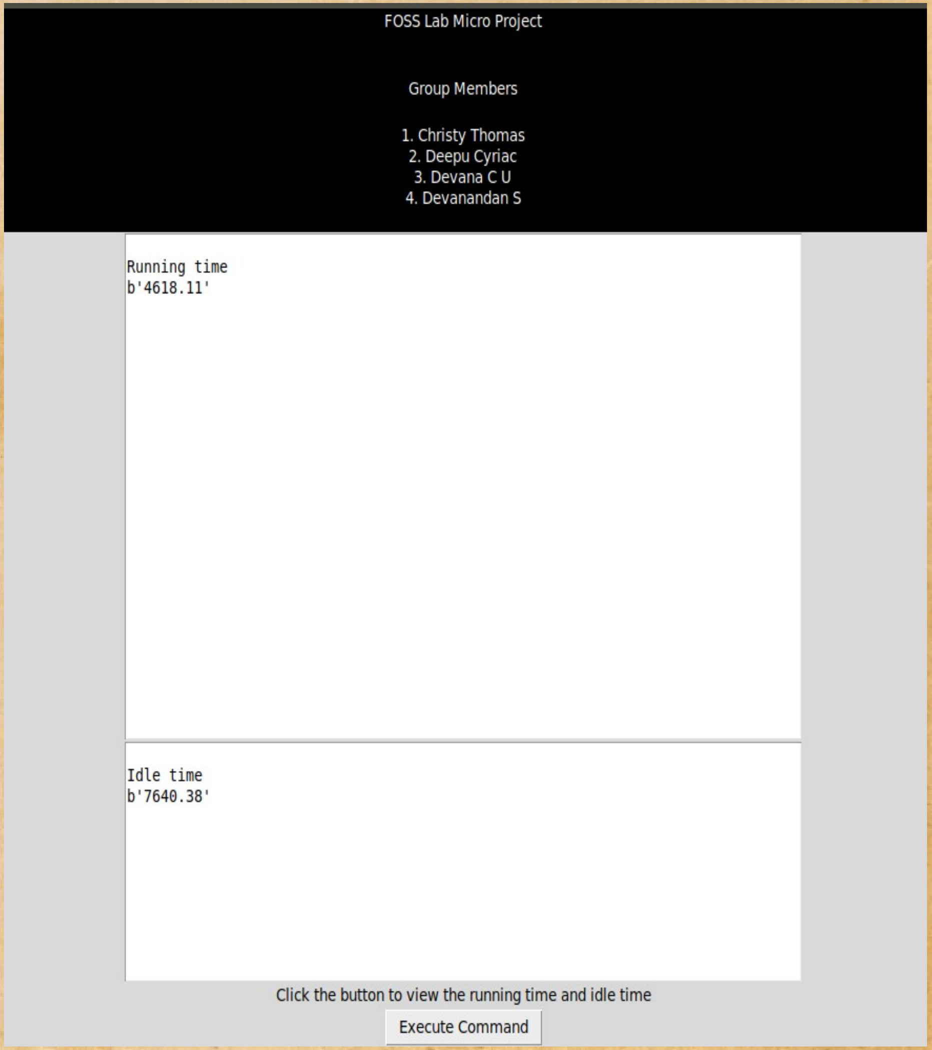


FOSS Lab Micro Project

Group Members

1. Christy Thomas
2. Deepu Cyriac
3. Devana C U
4. Devanandan S

Click the button to view the running time and idle time



FOSS Lab Micro Project

Group Members

1. Christy Thomas
2. Deepu Cyriac
3. Devana C U
4. Devanandan S

```
Running time
b'4618.11'
```

```
Idle time
b'7640.38'
```

Click the button to view the running time and idle time

THANK YOU