

PYTHON SCRIPT FOR CPU, RAM, SWAP MEMORY MONITORING

Welcome! to the documentation of monitoring CPU, RAM, SWAP Memory Utilization using Python Script.

Objective:

Our objective is to create a python script for monitoring CPU, RAM, SWAP Memory utilization. And give a warning about the utilization using python modules.

Modules used:

- Psutil

Steps in the script:

- Import psutil
- CPU monitoring
- RAM monitoring
- SWAP monitoring

Import psutil:

Psutil- python system and process utilities

it is a cross-platform library for retrieving information on running processes and system utilization of CPU, memory, disks, network, sensors in Python. It is useful mainly for system monitoring, profiling, limiting process resources and the management of running processes.

Install:

First you need to check pip package is installed or not?

If not installed install pip package first. Then install psutil using the command below,

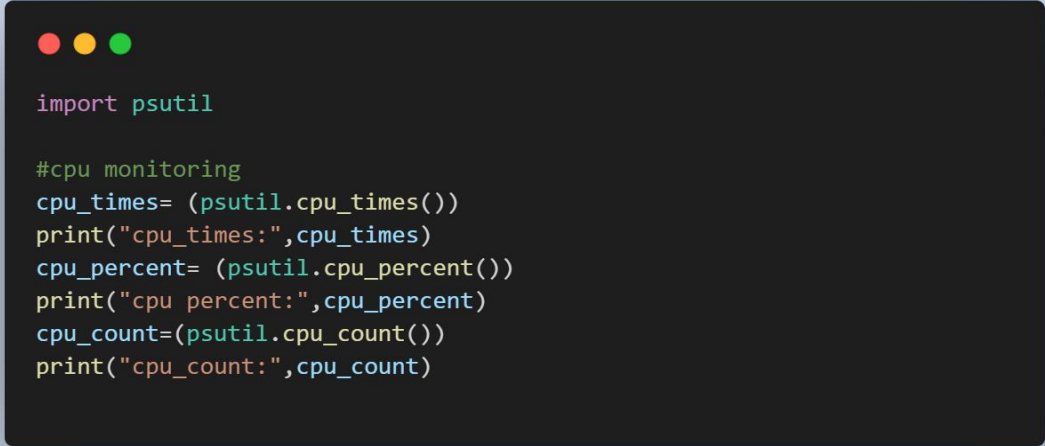
```
Pip install psutil
```

CPU monitoring:

1.CPU _TIMES

Return system CPU times as a named tuple. Every attribute represents the seconds the CPU has spent in the given mode. The attributes availability varies depending on the platform:

- **user**: time spent by normal processes executing in user mode; on Linux this also includes **guest** time
- **system**: time spent by processes executing in kernel mode
- **idle**: time spent doing nothing.



```
import psutil

#cpu monitoring
cpu_times= (psutil.cpu_times())
print("cpu_times:",cpu_times)
cpu_percent= (psutil.cpu_percent())
print("cpu percent:",cpu_percent)
cpu_count=(psutil.cpu_count())
print("cpu_count:",cpu_count)
```

2.CPU_Percent:

This function calculates the current system-wide CPU utilization as a percentage.

It is recommended to provide time interval (seconds) as parameter to the function over which the average CPU usage will be calculated, ignoring the interval parameter could result in high variation in usage values.

3.CPU_Count:

Return the number of logical CPUs in the system.

Output:

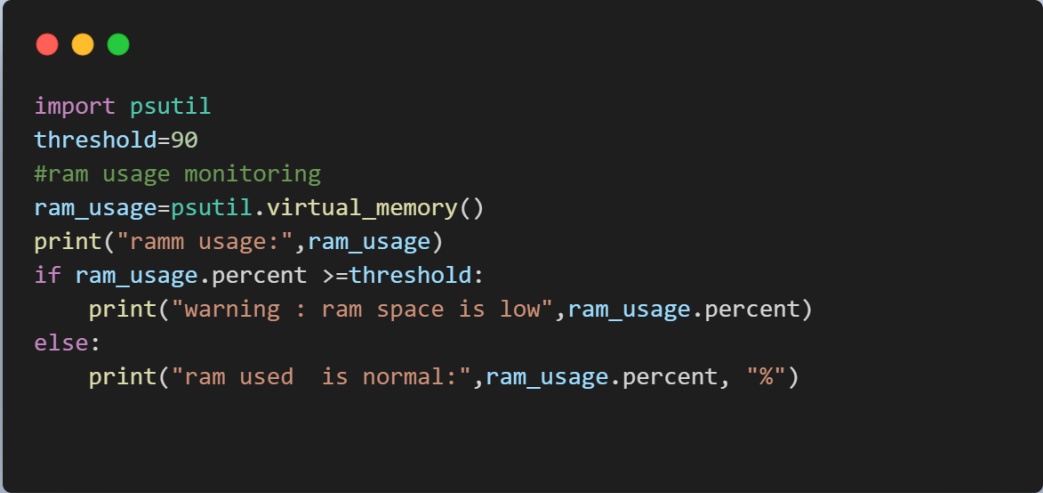
```
cpu_times:sctputimes(user=10381.328125,system=14195.890625,idle=99128.453125,interrupt=482.73
4375, dpc=563.890625)

cpu percent: 0.0

cpu_count: 4
```

RAM(virtual memory) Monitoring:

This function gives system memory usage in bytes. The sum of used and available may or may not be equal to total. In order to get details of free physical memory this function is used.



```
import psutil
threshold=90
#ram usage monitoring
ram_usage=psutil.virtual_memory()
print("ram usage:",ram_usage)
if ram_usage.percent >=threshold:
    print("warning : ram space is low",ram_usage.percent)
else:
    print("ram used is normal:",ram_usage.percent, "%")
```

Parameters:

- total – total physical memory excluding swap.
- available – the memory that can be given instantly to processes without the system going into swap.
- used – memory used.
- free – memory not used at and is readily available

Output:

```
ram usage: svmem(total=3723874304, available=871391232, percent=76.6,
used=2852483072, free=871391232)
```

```
ram used is normal: 76.6 %
```

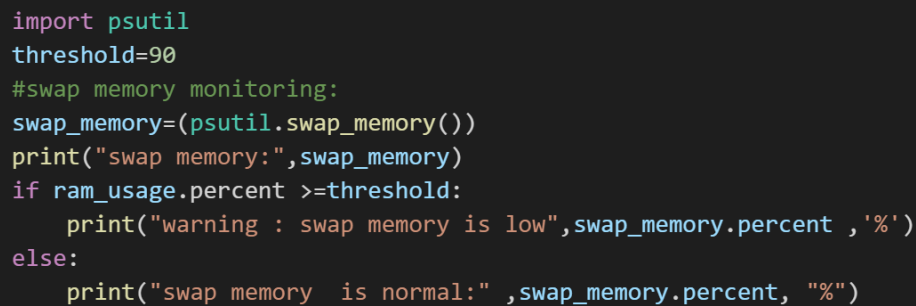
SWAP Monitoring:

swap memory is the dedicated amount of hard drive that is used whenever RAM runs out of memory.

This function provides details of swap memory statistics as a tuple.

Parameters:

- total – total swap memory in bytes
- used – used swap memory in bytes
- free – free swap memory in bytes
- percent – the percentage usage that is calculated as $(\text{total} - \text{available}) / \text{total} * 100$
- sin – the number of bytes the system has swapped in from disk
- sout – the number of bytes the system has swapped out from disk



```
import psutil
threshold=90
#swap memory monitoring:
swap_memory=(psutil.swap_memory())
print("swap memory:",swap_memory)
if ram_usage.percent >=threshold:
    print("warning : swap memory is low",swap_memory.percent ,'%')
else:
    print("swap memory is normal:" ,swap_memory.percent, "%")
```

Output:

```
swap memory: sswap(total=2281701376, used=1451925504,
free=829775872,percent=63.6, sin=0, sout=0)

swap memory is normal: 63.6 %
```

Conclusion:

Python script for Monitoring CPU,RAM,SWAP Is created and tested successfully.