



**INSTITUTO INFNET  
UNIDADE ACADÊMICA DE GRADUAÇÃO  
CURSO DE ENGENHARIA DE SOFTWARE – EDS-2017-N2**

**ALEXANDER VIEIRA DA SILVA  
MATRÍCULA: 041.380.007-55**

**PROJETO DE BLOCO - ARQUITETURA DE COMPUTADORES,  
SISTEMAS OPERACIONAIS E REDES  
APRESENTAÇÃO DE PROJETO**

**RIO DE JANEIRO-RJ  
2018**

**ALEXANDER VIEIRA DA SILVA**

**PROJETO DE BLOCO - ARQUITETURA DE COMPUTADORES,  
SISTEMAS OPERACIONAIS E REDES  
APRESENTAÇÃO DE PROJETO**

Trabalho apresentado para a disciplina  
**Projeto de Bloco Arquitetura de Computadores, Sistemas Operacionais e Redes**, pelo Curso de **Engenharia de Software** do Instituto Infnet, ministrada pelo(a) professor(a) Cassius Figueiredo

**RIO DE JANEIRO-RJ  
2018**

## SUMÁRIO

<b>1. INTRODUÇÃO.....</b>	<b>3</b>
<b>2. OBJETIVO.....</b>	<b>4</b>
<b>3. TESTES DE PERFORMANCE.....</b>	<b>5</b>
<b>    3.1. Teste de Performance - TP1.....</b>	<b>8</b>
<b>    3.2. Teste de Performance - TP2.....</b>	<b>10</b>
<b>    3.3. Teste de Performance - TP3.....</b>	<b>15</b>
<b>    3.4. Teste de Performance - TP4.....</b>	<b>26</b>
<b>        3.4.2. Execuções.....</b>	<b>29</b>
<b>    3.5. Teste de Performance - TP5.....</b>	<b>31</b>
<b>        3.5.2. Execuções.....</b>	<b>35</b>
<b>    3.6. Teste de Performance - TP6.....</b>	<b>36</b>
<b>        3.6.2. Execuções.....</b>	<b>41</b>
<b>    3.7. Teste de Performance - TP7.....</b>	<b>43</b>
<b>        3.7.1. Justificativa.....</b>	<b>44</b>
<b>        3.7.3. Execuções.....</b>	<b>50</b>
<b>    3.8. Teste de Performance - TP8.....</b>	<b>55</b>
<b>        3.8.1. Código Socket Cliente.....</b>	<b>56</b>
<b>        3.8.3. Código Socket Servidor.....</b>	<b>64</b>
<b>        3.8.4. Relatório.....</b>	<b>68</b>
<b>        3.8.5. Execuções.....</b>	<b>69</b>
<b>4. CONSIDERAÇÕES FINAIS.....</b>	<b>80</b>
<b>REFERÊNCIAS.....</b>	<b>81</b>

## 1. INTRODUÇÃO

Ao fazer uma retrospectiva dos três últimos séculos pode ser constatado que cada um deles foi dominado por uma tecnologia. O século XVIII foi marcado pelos grandes sistemas mecânicos que acompanharam a Revolução Industrial. O século XIX foi a era das máquinas a vapor, enquanto no século XX destaca-se os campos do processamento, distribuição da informação. Entre outros desenvolvimentos, vimos a instalação das redes de telefonia em escala global, a invenção do rádio, da televisão, o nascimento e crescimento espetacular da indústria da informática, o lançamento dos satélites de comunicação e, naturalmente, a internet.

Como o resultado do rápido progresso tecnológico, essas áreas estão convergindo rapidamente no século XXI e as diferenças entre a obtenção, envio, armazenamento e processamento de informações estão desaparecendo. Organizações com centenas de escritórios distribuídos por uma vasta área geográfica esperam com um simples pressionar de botão consultar a situação de suas filiais mais remotas.

Com a fusão dos computadores e comunicações teve uma profunda influência na forma como os sistemas computacionais são organizados. O conceito então predominante de “centro de computação” como uma sala com um grande computador centralizador de todos os trabalhos das organizações agora está completamente obsoleto. Foi substituído por outro em que os trabalhos são realizados por um grande número de computadores autônomos, porém interconectados. Esses sistemas são chamados de redes de computadores.

## 2. OBJETIVO

O escopo do projeto consiste em entregar um aplicativo simples de apresentação textual do monitoramento e análise de computadores em rede. Ele deverá ser implementado em Python usando módulos como psutil (para capturar dados do sistema computacional) e sockets (para criar cliente e servidor) e será desenvolvido de forma incremental. A aplicação executa um processo cliente, efetuando requisições ao processo servidor, este recebe o pedido, processa e responde a requisição. Como arquitetura de redes de computadores é o assunto abordado, o servidor responderá informações estatísticas sobre a arquitetura da CPU, de memória, do disco rígido, de sistema de arquivos, processos e finalmente arquitetura de rede, contando com o apoio da linguagem de programação Python. Esta linguagem foi escolhida pois atualmente tem sido bem aceita no mercado da web, embora tenha sido introduzida no meio acadêmico.

### 3. TESTES DE PERFORMANCE

O TP9 corresponde à junção dos TPs anteriores. Você deve fazer com que seu programa cliente apresente todas as informações presentes nos TPs 4, 5, 6 e 7 sejam exibidas. Tais informações devem ser obtidas da máquina onde o processo servidor está executando. Portanto, o processo servidor deve obtê-las e enviar ao processo cliente.

Lembre-se que as informações requisitadas em cada TP devem estar presentes:

Informações necessárias do TP 4:

- Porcentagem do uso de memória;
- Porcentagem do uso de CPU;
- Porcentagem do uso de disco;
- IP da máquina.

Informações necessárias do TP 5:

- Alterar a porcentagem do uso de CPU para cada núcleo (*core*);
- Nome/modelo da CPU (*brand*);
- Tipo da arquitetura (*arch*);
- Palavra do processador (*bits*);
- Frequência total e frequência de uso da CPU;
- Número total de núcleos (núcleo físico) e threads (núcleo lógico).

Informações necessárias do TP 6:

- Ao menos 3 informações de arquivos/diretórios (exemplos: nome, tamanho em disco, localização, data de criação, data de modificação, tipo, etc.);
- Ao menos 3 informações de processos executando no computador (exemplos: PID, nome do executável, consumo de processamento, consumo de memória, etc.).

Informações necessárias do TP 7:

- Ao menos 3 informações de interfaces de redes (exemplos: interfaces disponíveis, IP, *gateway*, máscara de subrede, etc.).

O TP8 corresponde à criação do cliente e servidor.

Seus experimentos podem ser realizados com cliente e servidor na mesma máquina. Além dos TPs 4, 5, 6, 7 e 8, já mencionados, você deve entregar também

os TP1, TP2 e TP3. Neles, existem apenas as respostas dos problemas enunciados. Você deve adicionar as perguntas e as respostas no documento final. Além disso, você deve contextualizar tais exercícios como parte inicial do aprendizado para a implementação do projeto final.

Como o TP9 corresponde a junção de todos os Tps, então vale ressaltar que cada exercício de programação proposto em cada TP são importantíssimos para o treinamento de programação, a fim de implementar e concluir o projeto.

Os TP1 e TP2 disponibilizaram exercícios visuais de programação simples usando a linguagem de programação visual Google Blockly Game e Google Blockly Code com o objetivo de desenvolver habilidades de lógica de programação. Assim, contribuiram para agregar conhecimento, principalmente para os alunos que nunca tiveram contato com uma linguagem de programação.

Agora, o TP3 abordou exercícios de programação utilizando estruturas condicionais, de repetição, listas e dicionários. Estas são estruturas importantíssimas para qualquer linguagem de programação. Desse modo, foram de suma importância para iniciar o desenvolvimento do projeto.

Enquanto do TP4, até o TP7, proporcionaram exercícios de captura de informações através da rede, usando como apoio a linguagem de programação python, somado aos conhecimentos adquiridos nos Tps anteriores, permitindo dar continuidade ao desenvolvimento do projeto, uma vez que todo artefato criado seria usado. Para finalizar o TP8 apresenta exercícios usando o modulo socket em python para implementar um modelo cliente-servidor, com o objetivo de estabelecer uma conexão entre diferentes usuários de aplicação, permitindo que se comuniquem. Portanto, pode se considerar que este TP possui uma papel fundamental para o desenvolvimento e conclusão do projeto.

Além disso, para obtenção de informações requisitadas para o servidor foram implementadas 6 (seis) funções com a finalidade de capturar tais informações. Segue abaixo as seguintes funções:

1. print\_info\_cpu( ) - exibe informações da arquitetura e percentual de processamento em uso da CPU.
2. print\_info\_memory( ) - exibe informações sobre a capacidade de memória.
3. print\_info\_disk( ) - exibe informações sobre a capacidade de armazenamento do disco rígido.
4. print\_info\_dir( ) - exibe informações sobre partições, sistema de arquivos, diretórios e arquivos.
5. print\_info\_process( ) - exibe informações sobre processos em execução.
6. print\_info\_net( ) - exibe informações sobre dados estatísticos das interfaces, a situação de entrada e saída de dados, mtu, velocidade e afins.

Quanto as informações exibidas a justificativa pela escolha de tais informações foi porque são consideradas essenciais para administração e análise em qualquer sistema computacional e rede de computadores

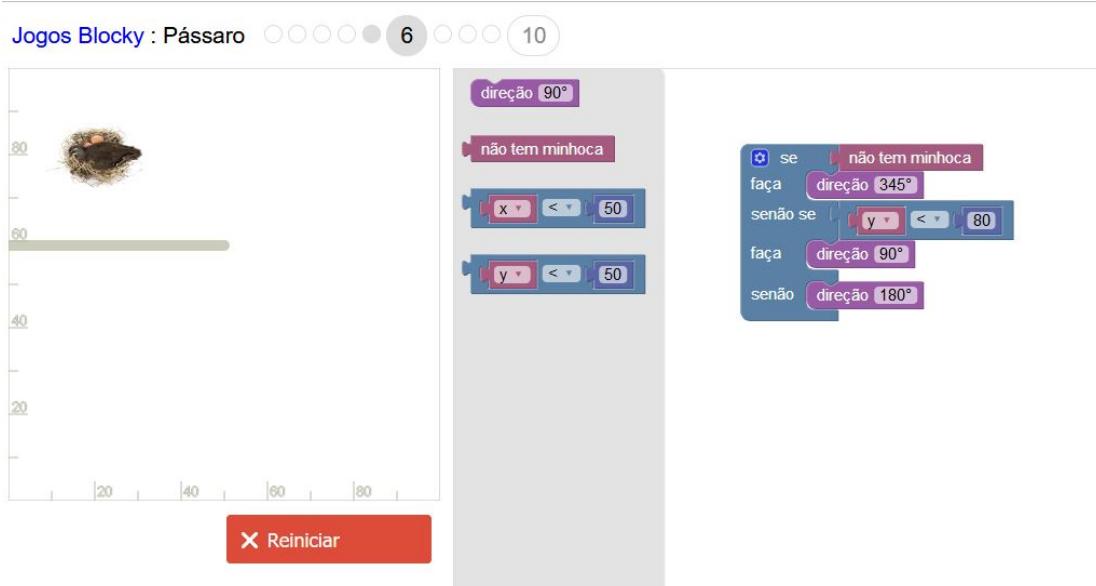
### 3.1. Teste de Performance - TP1

Usando o Blockly Game (acesse: <https://blockly-games.appspot.com/>), resolva cada um dos problemas abaixo:

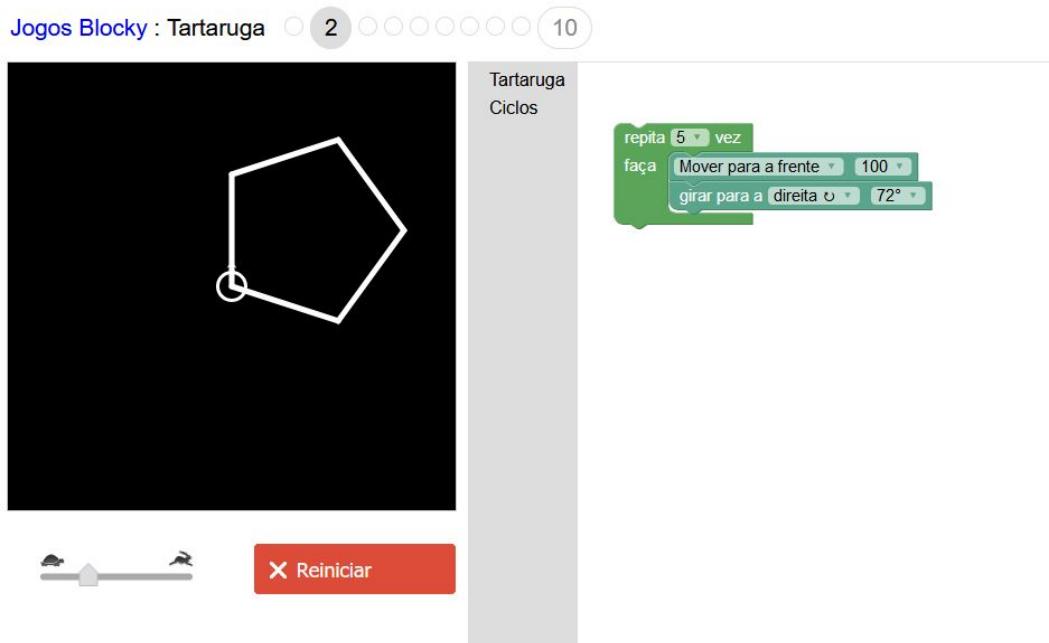
- a) Selecione o jogo do Labirinto e resolva o nível 6.



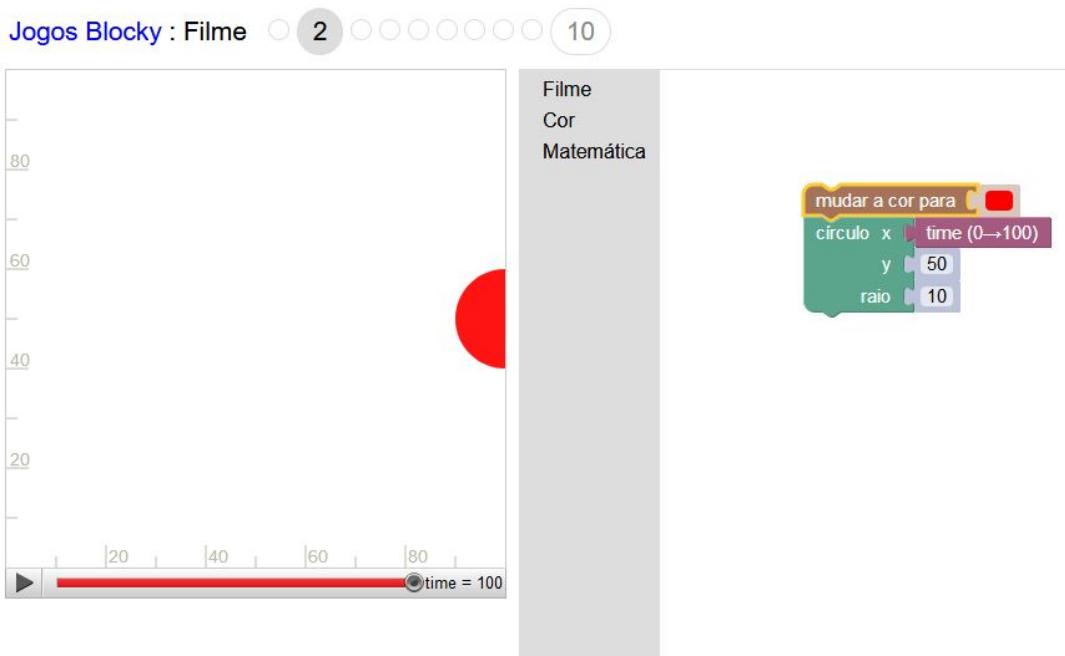
- b) Selecione o jogo do Pássaro e resolva o nível 6.



c) Selecione o jogo do Tartaruga e resolva o nível 2.



d) Selecione o jogo do Filme e resolva o nível 2.



### 3.2. Teste de Performance - TP2

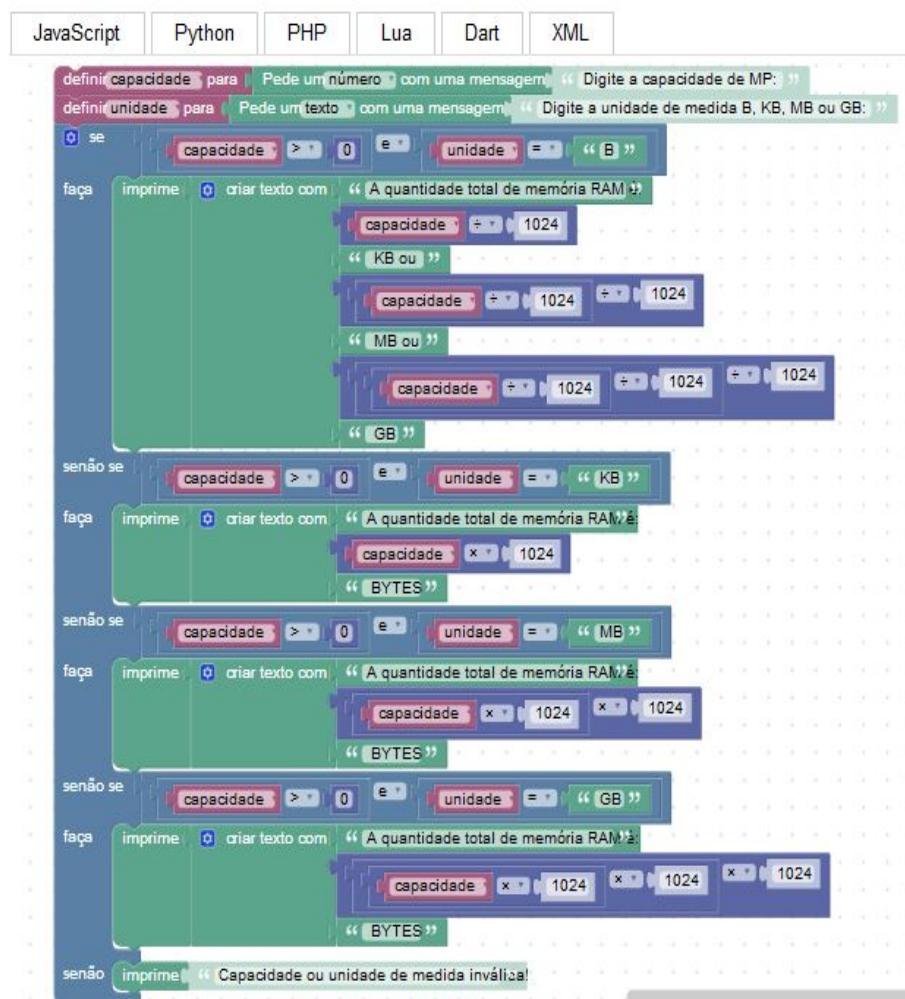
Usando o Google Blockly Code (acesse: <https://blockly-demo.appspot.com/static/demos/code/index.html?lang=pt-br>), faça:

1. Escreva um programa que peça dois valores a um usuário: a capacidade de uma memória principal e a unidade de medida usada para representar esta capacidade (B, KB, MB, GB). Em seguida, o seu programa deve indicar qual é a quantidade de bytes total desta memória dados estes valores. Use a tabela a seguir para fazer a conversão:

1KB | 1024B

1MB |  $1024 \times 1024B = 1048576B$

1GB |  $1024 \times 1024 \times 1024B = 1073741824B$



Blocos JavaScript Python PHP Lua Dart XML

```

capacidade = None
unidade = None

def text_prompt(msg):
    try:
        return raw_input(msg)
    except NameError:
        return input(msg)

capacidade = float(text_prompt('Digite a capacidade de MP: '))
unidade = text_prompt('Digite a unidade de medida B, KB, MB ou GB: ')
if capacidade > 0 and unidade == 'B':
    print(''.join([str(x) for x in ['A quantidade total de memória RAM é:', capacidade / 1024, ' KB ou ', (capacidade / 1024) / 1024, ' MB ou ', ((capacidade / 1024) / 1024) / 1024, ' GB']]))

elif capacidade > 0 and unidade == 'KB':
    print(''.join([str(x2) for x2 in ['A quantidade total de memória RAM é:', capacidade * 1024, ' BYTES']]))

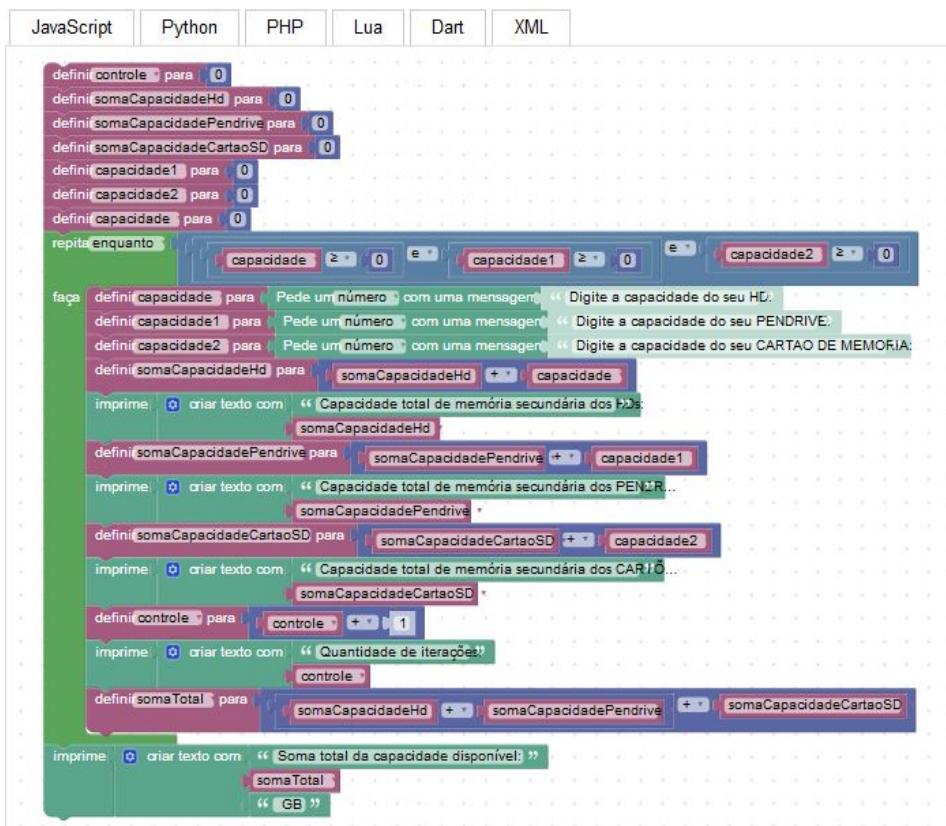
elif capacidade > 0 and unidade == 'MB':
    print(''.join([str(x3) for x3 in ['A quantidade total de memória RAM é:', (capacidade * 1024) * 1024, ' BYTES']]))

elif capacidade > 0 and unidade == 'GB':
    print(''.join([str(x4) for x4 in ['A quantidade total de memória RAM é:', ((capacidade * 1024) * 1024) * 1024, ' BYTES']]))

else:
    print('Capacidade ou unidade de medida inválida!')

```

**2. Escreva um programa que obtenha valores da capacidade em GB de diversas memórias secundárias (HDs, Pendrives, cartão de memória...) do usuário até que ele digite um valor negativo. Ao terminar de obter estes números, informe quanto de capacidade total de memória o usuário tem disponível, em GB.**



Blocos	JavaScript	Python	PHP	Lua	Dart	XML
--------	------------	--------	-----	-----	------	-----

```

capacidade = None
capacidade1 = None
capacidade2 = None
somaCapacidadeCartaoSD = None
controle = None
somaCapacidadeHd = None
somaCapacidadePendrive = None
somaTotal = None

def text_prompt(msg):
    try:
        return raw_input(msg)
    except NameError:
        return input(msg)

controle = 0
somaCapacidadeHd = 0
somaCapacidadePendrive = 0
somaCapacidadeCartaoSD = 0
capacidade1 = 0
capacidade2 = 0
capacidade = 0
while capacidade >= 0 and capacidade1 >= 0 and capacidade2 >= 0:
    capacidade = float(text_prompt('Digite a capacidade do seu HD: '))
    capacidade1 = float(text_prompt('Digite a capacidade do seu PENDRIVE: '))
    capacidade2 = float(text_prompt('Digite a capacidade do seu CARTAO DE MEMORIA: '))
    somaCapacidadeHd = somaCapacidadeHd + capacidade
    print(str('Capacidade total de memória secundária dos HDs: ') + str(somaCapacidadeHd))
    somaCapacidadePendrive = somaCapacidadePendrive + capacidade1
    print(str('Capacidade total de memória secundária dos PENDRIVES: ') + str(somaCapacidadePendrive))
    somaCapacidadeCartaoSD = somaCapacidadeCartaoSD + capacidade2
    print(str('Capacidade total de memória secundária dos CARTÕES SD: ') + str(somaCapacidadeCartaoSD))
    controle = controle + 1
    print(str('Quantidade de iterações: ') + str(controle))
    somaTotal = (somaCapacidadeHd + somaCapacidadePendrive) + somaCapacidadeCartaoSD
print("".join([str(x) for x in ['Soma total da capacidade disponível: ', somaTotal, ' GB']]))


```

### 3. Converta os códigos de 1 e 2 para Python e mostre o resultado no Thonny executando alguns testes.

#### Código I

Thonny - C:/Users/AlexanderSilva/Desktop/Projeto\_bloco\_Thonny\_questao\_3\_codigo\_1\_tp2.JPG @ 2:16

File Edit View Run Tools Help

soma.py x Projeto\_Bloco\_Questao\_3\_codigo\_2\_tp2.py x Projeto\_bloco\_Thonny\_questao\_3\_codigo\_1\_tp2.JPG x

```

capacidade = float(input('Digite a capacidade de MP: '))
unidade = input('Digite a unidade de medida B, KB, MB ou GB: ')
if capacidade > 0 and unidade == 'B':
    print("".join([str(x) for x in ['A quantidade total de memória RAM é: ', capacidade / 1024, ' KB ou ', (capacidade / 1024) / 1024, ' MB ou ', ((capacidade / 1024) / 1024) / 1024, ' GB']]))
elif capacidade > 0 and unidade == 'KB':
    print("".join([str(x2) for x2 in ['A quantidade total de memória RAM é: ', capacidade * 1024, ' BYTES']]))
elif capacidade > 0 and unidade == 'MB':
    print("".join([str(x3) for x3 in ['A quantidade total de memória RAM é: ', (capacidade * 1024) * 1024, ' BYTES']]))
elif capacidade > 0 and unidade == 'GB':
    print("".join([str(x4) for x4 in ['A quantidade total de memória RAM é: ', ((capacidade * 1024) * 1024) * 1024, ' BYTES']]))
else:
    print('Capacidade ou unidade de memória inválida! ')

```

## Resultado

```

Thonny - C:/Users/Alexander.Silva/Desktop/Projeto_bloco_Thonny_questao_3_codigo_1_tp2.JPG @ 2:16
File Edit View Run Tools Help
soma.py x Projeto_Bloco_Questao-3_codigo_2_tp2.py x Projeto_bloco_Thonny_questao_3_codigo_1_tp2.JPG x
<
Shell

>>> %Run Projeto_bloco_Thonny_questao_3_codigo_1_tp2.JPG
Digite a capacidade de MP: 1024
Digite a unidade de medida B, KB, MB ou GB: B
A quantidade total de memória RAM é: 1.0 KB ou 0.0009765625 MB ou 9.5367431640625e-07 GB

>>> %Run Projeto_bloco_Thonny_questao_3_codigo_1_tp2.JPG
Digite a capacidade de MP: 1
Digite a unidade de medida B, KB, MB ou GB: KB
A quantidade total de memória RAM é: 1024.0 BYTES

>>> %Run Projeto_bloco_Thonny_questao_3_codigo_1_tp2.JPG
Digite a capacidade de MP: 1
Digite a unidade de medida B, KB, MB ou GB: MB
A quantidade total de memória RAM é: 1048576.0 BYTES

>>> %Run Projeto_bloco_Thonny_questao_3_codigo_1_tp2.JPG
Digite a capacidade de MP: 1
Digite a unidade de medida B, KB, MB ou GB: GB
A quantidade total de memória RAM é: 1073741824.0 BYTES

>>>

```

## Código II

```

Thonny - G:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP2\Thonny\Projeto_Bloco_Questao-3_codigo_2_tp2.py @ 11:66
File Edit View Run Tools Help
Projeto_Bloco_Questao-3_codigo_2_tp2.py x
<

controle = 0
somaCapacidadeHd = 0
somaCapacidadePendrive = 0
somaCapacidadeCartaoSD = 0
capacidade1 = 0
capacidade2 = 0
capacidade = 0
while capacidade >= 0 and capacidade1 >= 0 and capacidade2 >= 0:
    capacidade = float(input('Digite a capacidade do seu HD: '))
    capacidade1 = float(input('Digite a capacidade do seu PENDRIVE: '))
    capacidade2 = float(input('Digite a capacidade do seu CARTAO SD: '))
    somaCapacidadeHd = somaCapacidadeHd + capacidade
    print(str('Capacidade total de memória secundária dos HDs: ') + str(somaCapacidadeHd))
    somaCapacidadePendrive = somaCapacidadePendrive + capacidade1
    print(str('Capacidade total de memória secundária dos PENDRIVES: ') + str(somaCapacidadePendrive))
    somaCapacidadeCartaoSD = somaCapacidadeCartaoSD + capacidade2
    print(str('Capacidade total de memória secundária dos CARTÕES SD: ') + str(somaCapacidadeCartaoSD))
    controle = controle + 1
    print(str('Quantidade de iterações: ') + str(controle))
    somaTotal = (somaCapacidadeHd + somaCapacidadePendrive) + somaCapacidadeCartaoSD
    print(''.join([str(x) for x in ['Soma total da capacidade disponível: ', somaTotal, ' GB']]))

<

```

## Resultado

```
<  
Shell  
>>> %Run Projeto_Bloco_Questao-3_codigo_2_tp2.py  
  
Digite a capacidade do seu HD: 120  
Digite a capacidade do seu PENDRIVE: 64  
Digite a capacidade do seu CARTAO SD: 32  
Capacidade total de memória secundária dos HDs: 120.0  
Capacidade total de memória secundária dos PENDRIVES: 64.0  
Capacidade total de memória secundária dos CARTÕES SD: 32.0  
Quantidade de iterações: 1  
Digite a capacidade do seu HD: 320  
Digite a capacidade do seu PENDRIVE: 32  
Digite a capacidade do seu CARTAO SD: 16  
Capacidade total de memória secundária dos HDs: 440.0  
Capacidade total de memória secundária dos PENDRIVES: 96.0  
Capacidade total de memória secundária dos CARTÕES SD: 48.0  
Quantidade de iterações: 2  
Digite a capacidade do seu HD: -20  
Digite a capacidade do seu PENDRIVE: -16  
Digite a capacidade do seu CARTAO SD: -8  
Capacidade total de memória secundária dos HDs: 420.0  
Capacidade total de memória secundária dos PENDRIVES: 80.0  
Capacidade total de memória secundária dos CARTÕES SD: 40.0  
Quantidade de iterações: 3  
Soma total da capacidade disponível: 540.0 GB  
>>> |
```

### 3.3. Teste de Performance - TP3

**Condicionais:**

**1. Escreva um programa em Python que leia um número e apenas indique se ele é positivo.**

The screenshot shows the Spyder IDE interface. The top menu bar includes Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, and Ajuda. Below the menu is a toolbar with various icons. The left pane displays the code for 'Questao\_1\_NumPositivo.py':

```

1 # coding: iso-8859-1 -*-
2
3 number = float (input('Entre com um número: '))
4 if number > 0:
5     print ('O número %d é positivo!' %number)

```

The right pane shows the IPython console output:

```

In [3]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco_1/Questao_1_NumPositivo.py', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS')
Entre com um número: 1
O número 1 é positivo!

```

**2. Escreva um programa em Python que leia um número e indique se ele é positivo ou negativo.**

The screenshot shows the Spyder IDE interface. The top menu bar includes Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, and Ajuda. Below the menu is a toolbar with various icons. The left pane displays the code for '\_NumPositivo.py' and 'Questao\_2\_NumPositivo\_Negativo.py':

```

1 # coding: iso-8859-1 -*-
2
3 number = float (input('Entre com um número: '))
4 if number > 0:
5     print ('O número %d é positivo!' %number)
6 elif number < 0:
7     print ('O número %d é negativo!' %number)
8 else:
9     print ('O número %d é nulo!' %number)

```

The right pane shows the IPython console output:

```

In [5]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco_1/_NumPositivo.py', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS')
Entre com um número: -1
O número -1 é negativo!

In [6]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco_1/Questao_2_NumPositivo_Negativo.py', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS')
Entre com um número: 0
O número 0 é nulo!

In [7]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco_1/Questao_2_NumPositivo_Negativo.py', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS')
Entre com um número: 1
O número 1 é positivo!

```

**3. Escreva um programa em Python que leia um número (inteiro) e indique se ele é par ou ímpar.**

```

Spyder (Python 3.6)

Arquivo Editar Pesquisar Código Executar Depurar Consoles Projetos Ferramentas Ver Ajuda

Editor - F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\Questao_3_NumPar_Impar.py | Console IPython
Questao_1_NumPositivo.py Questao_3_NumPar_Impar.py | Console 1/A

1 # coding: iso-8859-1 -*-
2
3 number = float(input('Entre com um número: '))
4 if number % 2 == 0:
5     print ('O número %d é par!' %number)
6 else:
7     print ('O número %d é ímpar!' %number)
8

```

```

In [14]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Questao_3_NumPar_Impar.py', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco')
Entre com um número: 0
O número 0 é par!

In [15]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Questao_3_NumPar_Impar.py', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco')
Entre com um número: 1
O número 1 é ímpar!

In [16]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Questao_3_NumPar_Impar.py', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco')
Entre com um número: 2
O número 2 é par!

```

**4. Escreva um programa em Python que leia um número (inteiro) e verifique se ele é positivo ou negativo. Se for positivo, indique se ele é par ou ímpar.**

```

Spyder (Python 3.6)

Arquivo Editar Pesquisar Código Executar Depurar Consoles Projetos Ferramentas Ver Ajuda

Editor - F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\Questao_4_NumPosNeg_Par_Impar.py | Console IPython
Questao_1_NumPositivo.py Questao_4_NumPosNeg_Par_Impar.py | Console 1/A

1 # coding: iso-8859-1 -*-
2
3 number = float(input("Entre com um número: "))
4 if number > 0:
5     if number % 2 == 0:
6         print ("O número %d é positivo e par!" % number)
7 if number < 0:
8     if number % 2 == 0:
9         print ("O número %d é negativo!" % number)
10 if number > 0:
11     if number % 2 == 1:
12         print ("O número %d é positivo ímpar!" % number)
13 if number < 0:
14     if number % 2 == 1:
15         print ("O número %d é negativo!" % number)
16 if number == 0:
17     if number % 2 == 0:
18         print ("O número %d é nulo e par!" % number)
19

```

```

In [18]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Questao_4_NumPosNeg_Par_Impar.py', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco')
Entre com um número: 0
O número 0 é nulo e par!

In [19]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Questao_4_NumPosNeg_Par_Impar.py', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco')
Entre com um número: 1
O número 1 é positivo ímpar!

In [20]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Questao_4_NumPosNeg_Par_Impar.py', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco')
Entre com um número: 2
O número 2 é positivo e par!

In [21]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Questao_4_NumPosNeg_Par_Impar.py', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco')
Entre com um número: -1
O número -1 é negativo!

In [22]: |

```

**5. Escreva um programa em Python que leia dois números e indique qual deles é o maior.**

The screenshot shows the Spyder IDE interface. The top menu bar includes Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, and Ajuda. Below the menu is a toolbar with various icons. The main area has tabs for 'Questao\_1\_NumPositivo.py' and 'Questao\_5\_NumMaior.py'. The code in 'Questao\_5\_NumMaior.py' is:

```

1 # -*- encoding: utf-8 -*-
2
3 n1 = float(input('Digite o primeiro número: '))
4 n2 = float(input('Digite o segundo número: '))
5 if n1 > n2:
6     print('n1 é maior! ')
7 elif n1 < n2:
8     print('n2 é maior! ')
9 else:
10    print('Os números são iguais! ')

```

To the right, the 'Console IPython' window shows the execution of the script. It prompts for two numbers, 10 and 20, and prints that n2 is greater than n1.

```

In [28]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Co...', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3')
Digite o primeiro número: 10
Digite o segundo número: 20
n2 é maior!

```

It then runs the script again with 30 and 20, printing that n1 is greater than n2.

```

In [29]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Co...', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3')
Digite o primeiro número: 30
Digite o segundo número: 20
n1 é maior!

```

**6. Escreva um programa em Python que leia três números e indique qual deles é o menor.**

The screenshot shows the Spyder IDE interface. The top menu bar includes Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, and Ajuda. Below the menu is a toolbar with various icons. The main area has tabs for 'vo.py', 'Questao\_5\_NumMaior.py', and 'Questao\_6\_NumMenor.py'. The code in 'Questao\_6\_NumMenor.py' is:

```

1 # -*- encoding: utf-8 -*-
2
3 x = float(input('Digite o primeiro número: '))
4 y = float(input('Digite o segundo número: '))
5 z = float(input('Digite o terceiro número: '))
6 if x < y and x < z:
7     print('O 1º número é menor! ')
8 elif y < x and y < z:
9     print('O 2º número é menor! ')
10 elif z < x and z < y:
11     print('O 3º número é menor! ')
12 else:
13     print('Pelo menos dois números são iguais! ')

```

To the right, the 'Console IPython' window shows the execution of the script. It prompts for three numbers, 10, 20, and 30, and prints that the first number is the smallest.

```

In [31]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Co...', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3')
Digite o primeiro número: 10
Digite o segundo número: 20
Digite o terceiro número: 30
O 1º número é menor!

```

It then runs the script again with 20, 10, and 30, printing that the second number is the smallest.

```

In [32]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Co...', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3')
Digite o primeiro número: 20
Digite o segundo número: 10
Digite o terceiro número: 30
O 2º número é menor!

```

Finally, it runs the script again with 30, 20, and 10, printing that the third number is the smallest.

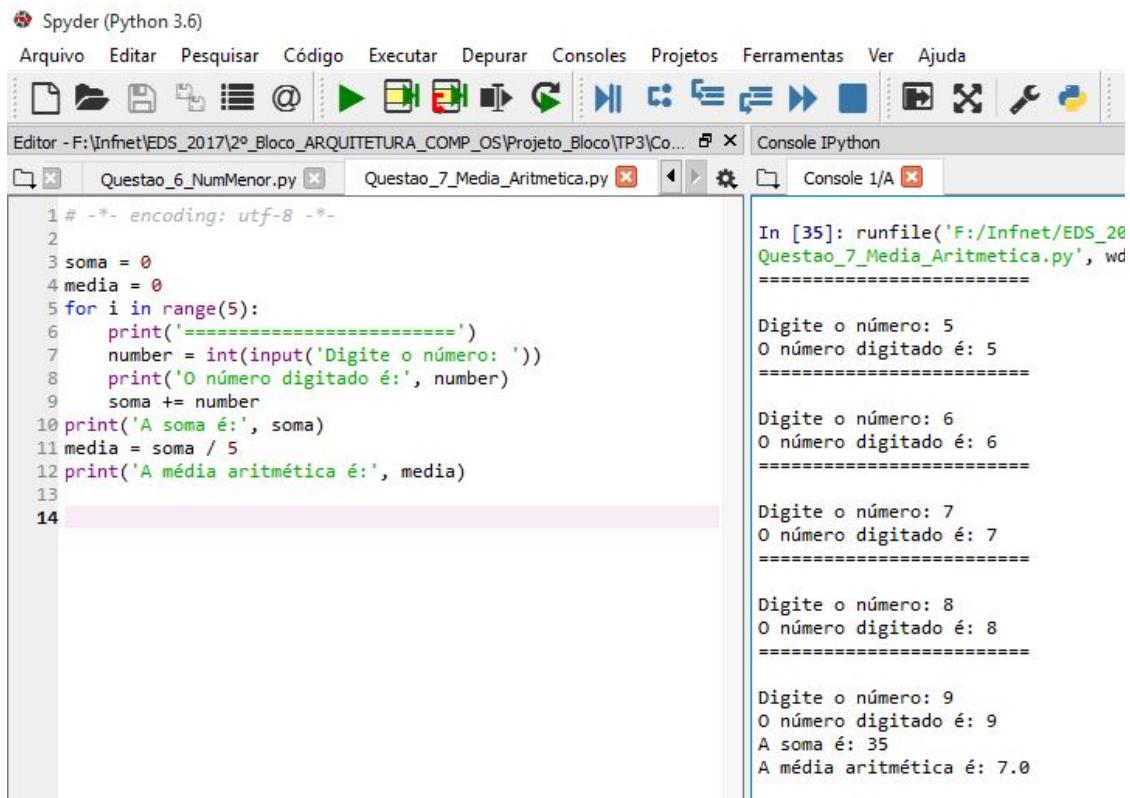
```

In [33]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Co...', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3')
Digite o primeiro número: 30
Digite o segundo número: 20
Digite o terceiro número: 10
O 3º número é menor!

```

## Repetições

**7. Escreva um programa em Python que leia 5 números e escreva a média aritmética entre eles.**



The screenshot shows the Spyder IDE interface with the following details:

- Title Bar:** Spyder (Python 3.6)
- Menu Bar:** Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, Ajuda
- Toolbar:** Standard file operations (New, Open, Save, etc.) and code execution icons.
- Editor Area:** Shows the code for `Questao_7_Media_Aritmetica.py`. The code reads five numbers from the user, calculates their sum, and then divides by 5 to find the average.

```

1 # -*- encoding: utf-8 -*-
2
3 soma = 0
4 media = 0
5 for i in range(5):
6     print('=====')
7     number = int(input('Digite o número: '))
8     print('O número digitado é:', number)
9     soma += number
10 print('A soma é:', soma)
11 media = soma / 5
12 print('A média aritmética é:', media)
13
14

```
- Console Area:** Shows the output of running the script. It prompts the user to enter five numbers (5, 6, 7, 8, 9) and then displays the calculated sum (35) and average (7.0).

```

In [35]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Co...
=====
Digite o número: 5
O número digitado é: 5
=====

Digite o número: 6
O número digitado é: 6
=====

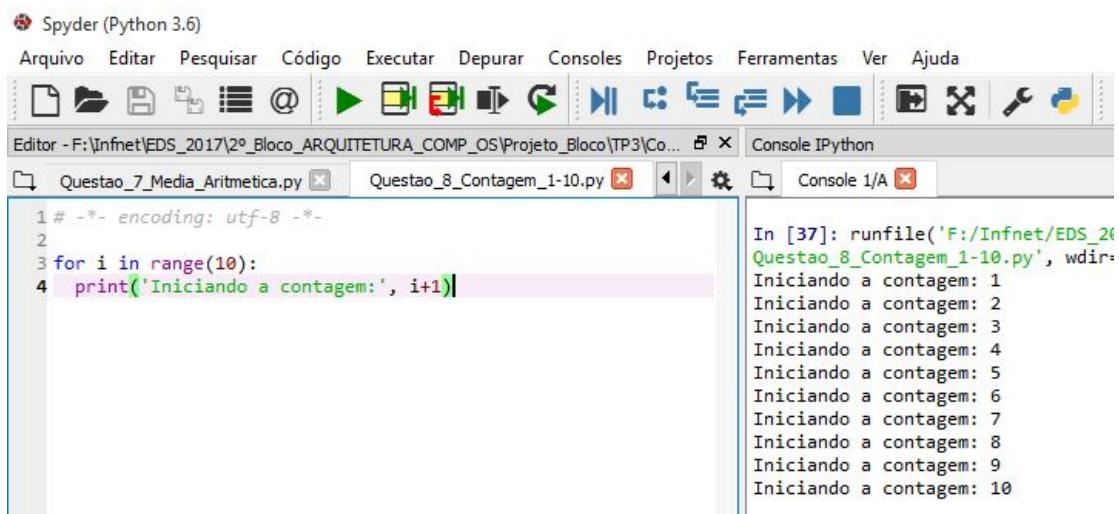
Digite o número: 7
O número digitado é: 7
=====

Digite o número: 8
O número digitado é: 8
=====

Digite o número: 9
O número digitado é: 9
A soma é: 35
A média aritmética é: 7.0
=====

.
.
```

**8. Escreva um programa em Python que escreva na tela os números de 1 a 10. Use a estrutura de repetição for.**



The screenshot shows the Spyder IDE interface with the following details:

- Title Bar:** Spyder (Python 3.6)
- Menu Bar:** Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, Ajuda
- Toolbar:** Standard file operations (New, Open, Save, etc.) and code execution icons.
- Editor Area:** Shows the code for `Questao_8_Contagem_1-10.py`. The code uses a for loop to iterate from 1 to 10, printing each number.

```

1 # -*- encoding: utf-8 -*-
2
3 for i in range(10):
4     print('Iniciando a contagem:', i+1)

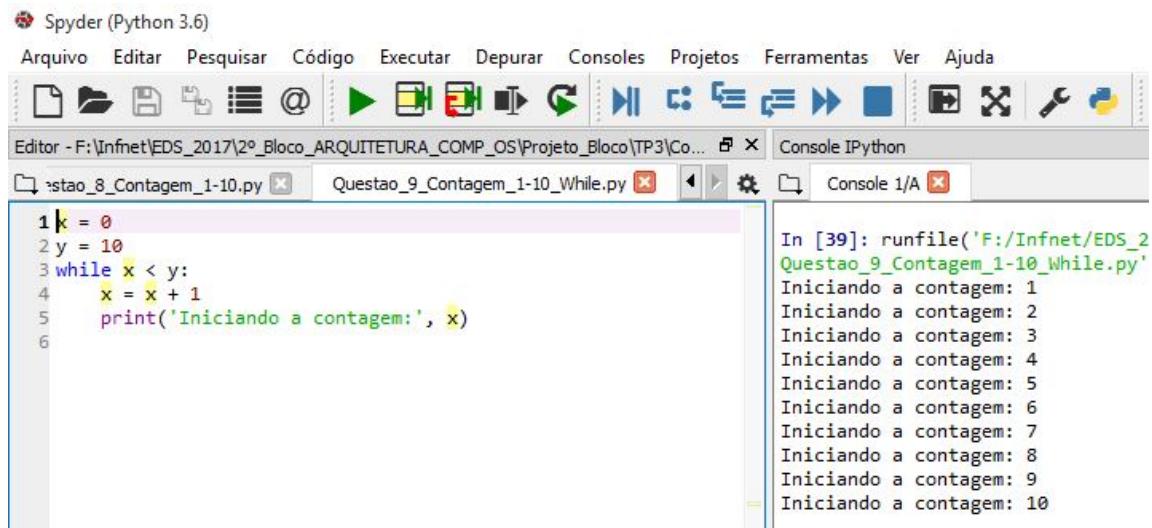
```
- Console Area:** Shows the output of running the script. It prints the message "Iniciando a contagem:" followed by the numbers 1 through 10.

```

In [37]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Co...
=====
Iniciando a contagem: 1
Iniciando a contagem: 2
Iniciando a contagem: 3
Iniciando a contagem: 4
Iniciando a contagem: 5
Iniciando a contagem: 6
Iniciando a contagem: 7
Iniciando a contagem: 8
Iniciando a contagem: 9
Iniciando a contagem: 10
=====

.
.
```

**9. Escreva um programa em Python que escreva na tela os números de 1 a 10.**  
**Use a estrutura de repetição while.**

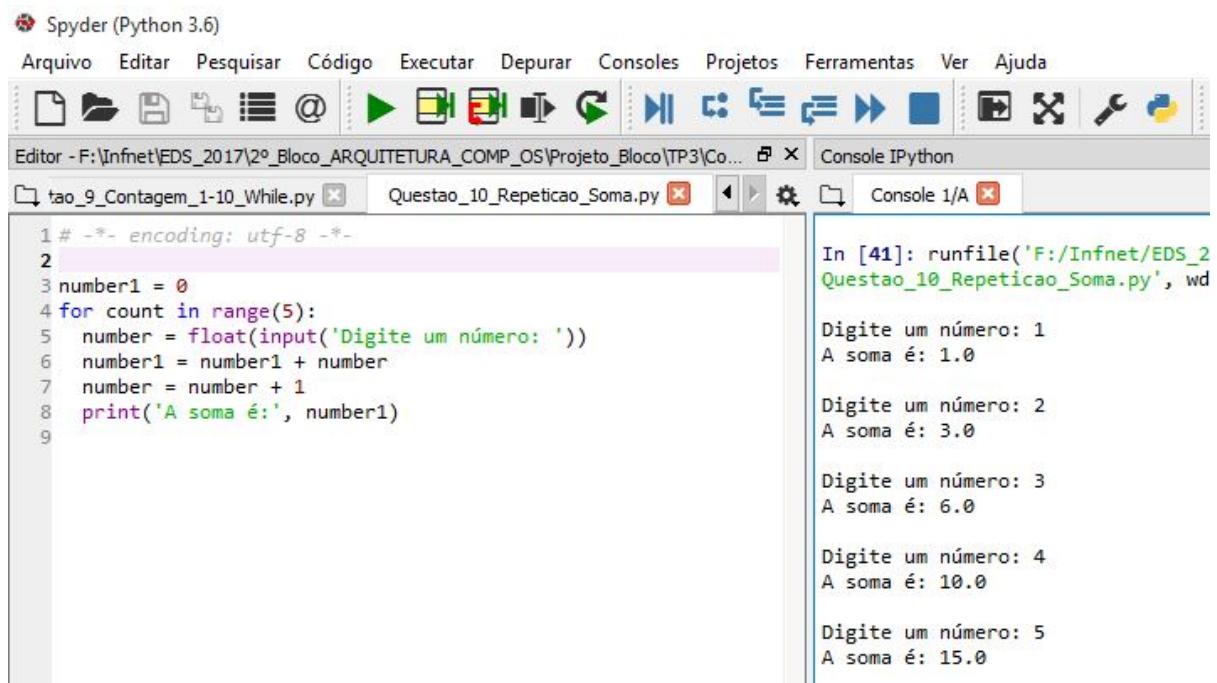


```

Spyder (Python 3.6)
Arquivo Editar Pesquisar Código Executar Depurar Consoles Projetos Ferramentas Ver Ajuda
Editor - F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\Co... × Console IPython
Questao_9_Contagem_1-10_While.py × Console 1/A ×
1 k = 0
2 y = 10
3 while x < y:
4     x = x + 1
5     print('Iniciando a contagem:', x)
6
In [39]: runfile('F:/Infnet/EDS_2
Questao_9_Contagem_1-10_While.py'
Iniciando a contagem: 1
Iniciando a contagem: 2
Iniciando a contagem: 3
Iniciando a contagem: 4
Iniciando a contagem: 5
Iniciando a contagem: 6
Iniciando a contagem: 7
Iniciando a contagem: 8
Iniciando a contagem: 9
Iniciando a contagem: 10

```

**10. Escreva um programa em Python que leia repetidamente 5 números do teclado e escreva a soma deles no final.**



```

Spyder (Python 3.6)
Arquivo Editar Pesquisar Código Executar Depurar Consoles Projetos Ferramentas Ver Ajuda
Editor - F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\Co... × Console IPython
tao_9_Contagem_1-10_While.py × Questao_10_Repeticao_Soma.py × Console 1/A ×
1 # -*- encoding: utf-8 -*-
2
3 number1 = 0
4 for count in range(5):
5     number = float(input('Digite um número: '))
6     number1 = number1 + number
7     number = number + 1
8     print('A soma é:', number1)
9
In [41]: runfile('F:/Infnet/EDS_2
Questao_10_Repeticao_Soma.py', wd
Digite um número: 1
A soma é: 1.0

Digite um número: 2
A soma é: 3.0

Digite um número: 3
A soma é: 6.0

Digite um número: 4
A soma é: 10.0

Digite um número: 5
A soma é: 15.0

```

**11. Escreva um programa em Python que leia repetidamente 5 números do teclado e indique, para cada, se é positivo.**

The screenshot shows the Spyder IDE interface with the following details:

- Title Bar:** Spyder (Python 3.6)
- Menu Bar:** Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, Ajuda
- Toolbar:** Standard file operations (New, Open, Save, etc.) and execution controls (Run, Stop, Step).
- Editor Area:** Shows two files: `estao_10_Repeticao_Soma.py` and `Questao_11_Repeticao_Positivo.py`. The code in `Questao_11_Repeticao_Positivo.py` is as follows:
 

```

1 # coding: iso-8859-1 -*-
2
3 for count in range(5):
4     number = float(input('Digite um número: '))
5     if number == 0:
6         print('O número:', str(number), 'é nulo.')
7     elif number > 0:
8         print('O número:', str(number), 'é positivo.')
9     else:
10        print('O número:', str(number), 'é negativo.')
      
```
- Console Area:** Shows the output of running the script:
 

```

In [46]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Co...', 1)
Digite um número: 0
O número: 0.0 é nulo.

Digite um número: 1
O número: 1.0 é positivo.

Digite um número: 2
O número: 2.0 é positivo.

Digite um número: 3
O número: 3.0 é positivo.

Digite um número: -1
O número: -1.0 é negativo.
      
```

**12. Escreva um programa em Python que leia repetidamente 5 números (inteiros) do teclado e indique, para cada, se é par ou ímpar.**

The screenshot shows the Spyder IDE interface with the following details:

- Title Bar:** Spyder (Python 3.6)
- Menu Bar:** Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, Ajuda
- Toolbar:** Standard file operations (New, Open, Save, etc.) and execution controls (Run, Stop, Step).
- Editor Area:** Shows two files: `Questao_1_NumPositivo.py` and `Questao_12_Repeticao_Par_Impar.py`. The code in `Questao_12_Repeticao_Par_Impar.py` is as follows:
 

```

1 # coding: iso-8859-1 -*-
2
3 for count in range(5):
4     number = float(input('Digite um número: '))
5     if number % 2 == 0:
6         print('O número: ', str(number), 'é par!')
7     elif number > 0 and number % 2 == 1:
8         print('O número: ', str(number), 'é ímpar!')
9     else:
10        print('Entre com um número positivo!')
      
```
- Console Area:** Shows the output of running the script:
 

```

In [48]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Co...', 1)
Digite um número: 0
O número: 0.0 é par!

Digite um número: 1
O número: 1.0 é ímpar!

Digite um número: 2
O número: 2.0 é par!

Digite um número: 3
O número: 3.0 é ímpar!

Digite um número: -1
Entre com um número positivo!
      
```

**13. Escreva um programa em Python que leia repetidamente 5 números do teclado e indique, no final, se a média aritmética desses números é igual, > ou < que 6.**

The screenshot shows the Spyder IDE interface with the following details:

- Toolbar:** Standard file operations (New, Open, Save, Print, Find, Copy, Paste, etc.) and execution buttons (Run, Stop, Run Cell).
- Menu Bar:** Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, Ajuda.
- Editor Area:** Shows two files: 'eticao\_Par\_Impar.py' and 'Questao\_13\_Repeticao\_Media\_Aritmetica.py'. The second file contains the following code:

```

1 # -*- encoding: utf-8 -*-
2 soma = 0
3 media = 6
4 for count in range(5):
5     number = float(input('Digite um número: '))
6     soma = soma + number
7     print('A soma é:', soma)
8 mediaAritmetica = soma / 5
9 print('A média aritmética é:', mediaAritmetica)
10 if mediaAritmetica == media:
11     print('A média é igual que', media)
12 if mediaAritmetica > media:
13     print('A média é maior que', media)
14 if mediaAritmetica < media:
15     print('A média é menor que', media)

```

- Console Area:** Shows the output of running the code in cell 52. It prompts for five numbers (7, 8, 10, 9, 5) and prints their sum and the calculated average (34.0, 6.8), then concludes that the average is greater than 6.

**14. Escreva um programa em Python que leia repetidamente N números do teclado e escreva o valor do dobro destes números. Leia N inicialmente.**

The screenshot shows the Spyder IDE interface with the following details:

- Toolbar:** Standard file operations (New, Open, Save, Print, Find, Copy, Paste, etc.) and execution buttons (Run, Stop, Run Cell).
- Menu Bar:** Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, Ajuda.
- Editor Area:** Shows two files: 'peticao\_Media\_Aritmetica.py' and 'Questao\_14\_Repeticao\_QtdeN\_N.py'. The second file contains the following code:

```

1 # coding: iso-8859-1 -*-
2
3 qtdeN = float(input('Digite um número: '))
4 for count in range(int(qtdeN)):
5     n = float(input('Digite outro número: '))
6     print(n * 2)

```

- Console Area:** Shows the output of running the code in cell 54. It asks for the quantity of numbers (3), then repeatedly prompts for three more numbers (10, 20, 30) and prints their doubles (20.0, 40.0, 60.0).

**15. Escreva um programa em Python que leia repetidamente N números do teclado. Leia N inicialmente. No final, escreva a média aritmética desses números.**

The screenshot shows the Spyder Python 3.6 IDE interface. The top menu bar includes Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, and Ajuda. Below the menu is a toolbar with various icons. The main area has two tabs: 'epeticao\_QtdeN\_N.py' and 'Questao\_15\_Repeticao\_QtdeN\_N\_Media.py'. The right pane is titled 'Console IPython' and shows the following interaction:

```
In [56]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Co...', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Co...')

Digite um número: 3
Digite outro número: 10
Digite outro número: 20
Digite outro número: 30
20.0
```

The code in 'Questao\_15\_Repeticao\_QtdeN\_N\_Media.py' is:

```
1 # coding: iso-8859-1 -*-
2
3 soma = 0
4 qtdeN = float(input('Digite um número: '))
5 for count in range(int(qtdeN)):
6     n = float(input('Digite outro número: '))
7     soma += n
8     #print(resultado)
9 media = soma / qtdeN
10 print(media)
11
```

**16. Escreva um programa em Python que leia números positivos até que um valor negativo seja lido. No final, apresente o resultado da soma deles (não inclua o número negativo).**

The screenshot shows the Spyder Python 3.6 IDE interface. The top menu bar includes Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, and Ajuda. Below the menu is a toolbar with various icons. The main area has two tabs: '1o\_QtdeN\_N\_Media.py' and 'Questao\_16\_Repeticao\_Negativo\_Soma.py'. The right pane is titled 'Console IPython' and shows the following interaction:

```
In [58]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUI...', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUI...')

Digite um número: 1
A soma é 1
Atenção! para sair digite um número negativo.

Digite um número: 2
A soma é 3
Atenção! para sair digite um número negativo.

Digite um número: 3
A soma é 6
Atenção! para sair digite um número negativo.

Digite um número: 4
A soma é 10
Atenção! para sair digite um número negativo.

Digite um número: 5
A soma é 15
Atenção! para sair digite um número negativo.

Digite um número: -1
Tudo bem você saiu do programa!
```

The code in 'Questao\_16\_Repeticao\_Negativo\_Soma.py' is:

```
1 n = 0
2 soma = 0
3 while n >= 0:
4     n = int(input("Digite um número: "))
5     if n >= 0:
6         soma = soma + n
7         print('A soma é', soma)
8         print('Atenção! para sair digite um número negativo.')
9         n = n + 1
10    else:
11        print('Tudo bem você saiu do programa!')
```

## Listas

### 17. Escreva um programa em Python que:

- gera uma lista de nome de cores e depois imprima ela. As cores são: azul, vermelho, verde, amarelo, violeta, marrom, branco e preto.
- Em seguida, remova a cor ‘marrom’ desta lista e imprima a lista após esta operação.
- Em seguida, adicione a cor ‘cinza’ na lista e imprima a lista a lista após esta operação.
- Em seguida, altere a cor ‘violeta’ para ‘rosa’.
- Em seguida, leia um nome de cor do teclado e a remova da lista. Não esqueça de checar se ela existe na lista.

```

Spyder (Python 3.6)
Arquivo Editar Pesquisar Código Executar Depurar Consoles Projetos Ferramentas Ver Ajuda
Editor - F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\Co... F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\CodigoFonte
Questa_17_a-e_ListaCores.py F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\Co... Console IPython
Console 1/A
1 Cores = ['azul', 'vermelho', 'verde', 'amarelo', 'violeta',
2         'marrom', 'branco', 'preto']
3 print('Minha lista de cores é:', Cores)
4 print('')
5 Cores.remove('marrom')
6 print('A cor marrom foi removida com sucesso!')
7 print('Minha lista de cores é:', Cores)
8 print('')
9 Cores.append('cinza')
10 print('A cor cinza foi adicionada com sucesso!')
11 print('Minha lista de cores é:', Cores)
12 print('')
13 Cores[cores.index('violeta')] = 'rosa'
14 print('A cor violeta foi alterada para rosa.')
15 print('Minha lista de cores é:', Cores)
16 print('')
17 cor = input('Digite uma cor: ')
18 if cor in Cores:
19     Cores.remove(cor)
20     print('A cor', cor, 'removida com sucesso!')
21 print('Minha lista de cores é:', Cores)

In [60]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/CodigoFonte/Questa_17_a-e_ListaCores.py', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/CodigoFonte')
Minha lista de cores é: ['azul', 'vermelho', 'verde', 'amarelo', 'violeta', 'marrom', 'branco', 'preto']

A cor marrom foi removida com sucesso!
Minha lista de cores é: ['azul', 'vermelho', 'verde', 'amarelo', 'violeta', 'branco', 'preto']

A cor cinza foi adicionada com sucesso!
Minha lista de cores é: ['azul', 'vermelho', 'verde', 'amarelo', 'violeta', 'branco', 'preto', 'cinza']

A cor violeta foi alterada para rosa.
Minha lista de cores é: ['azul', 'vermelho', 'verde', 'amarelo', 'rosa', 'branco', 'preto', 'cinza']

Digite uma cor: azul
A cor azul removida com sucesso!
Minha lista de cores é: ['vermelho', 'verde', 'amarelo', 'rosa', 'branco', 'preto', 'cinza']

In [61]:

```

### 18. Seja a lista [3, 5, 1, 7, 2, 8, 11, -2, 3, -6, 0]. Escreva um programa em Python que gere esta lista e apresente seus valores na ordem original e na ordem invertida.

```

Spyder (Python 3.6)
Arquivo Editar Pesquisar Código Executar Depurar Consoles Projetos Ferramentas Ver Ajuda
Editor - F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\Co... F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\Co...
Questa_17_a-e_ListaCores.py F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\Co... Console IPython
Console 1/A
1 lista = [3, 5, 1, 7, 2, 8, 11, -2, 3, -6, 0]
2 print('Lista na ordem original:', lista)
3 lista.reverse()
4 print('Lista na ordem invertida:', lista)

In [62]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/CodigoFonte/Questa_18_Lista_Ordem.py', wdir='F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/CodigoFonte')
Lista na ordem original: [3, 5, 1, 7, 2, 8, 11, -2, 3, -6, 0]
Lista na ordem invertida: [0, -6, 3, -2, 11, 8, 2, 7, 1, 5, 3]


```

**19. Escreva um programa em Python que leia nome de pessoas repetidamente e crie uma lista com estes nomes. No final, imprima esta lista. A leitura termina quando a palavra ‘sair’ for digitada.**

The screenshot shows the Spyder IDE interface. The top menu bar includes Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, and Ajuda. Below the menu is a toolbar with various icons. The main area has two tabs: 'Questao\_18\_Lista\_Ordem.py' and 'Questao\_19\_Lista\_Pessoas.py'. The right panel contains a 'Console IPython' tab where the output of the executed code is displayed.

```

Spyder (Python 3.6)
Arquivo Editar Pesquisar Código Executar Depurar Consoles Projetos Ferramentas Ver Ajuda
Editor - F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\Co... × Console IPython
Questao_18_Lista_Ordem.py Questao_19_Lista_Pessoas.py × Console 1/A ×
In [64]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Co...', wdir='F:/Infnet/EDS_2017')
Digite o seu nome: Alexander
Digite o seu nome: Paulo
Digite o seu nome: João
Digite o seu nome: Pedro
Digite o seu nome: Mazinho
Digite o seu nome: sair
['Alexander', 'Paulo', 'João', 'Pedro', 'Mazinho']

```

**20. Escreva um programa em Python que leia uma sequência de números do teclado e os armazene em uma lista. Em seguida, indique qual o maior valor dela.**

The screenshot shows the Spyder IDE interface. The top menu bar includes Arquivo, Editar, Pesquisar, Código, Executar, Depurar, Consoles, Projetos, Ferramentas, Ver, and Ajuda. Below the menu is a toolbar with various icons. The main area has two tabs: 'Questao\_19\_Lista\_Pessoas.py' and 'Questao\_20\_Lista\_Numerica.py'. The right panel contains a 'Console IPython' tab where the output of the executed code is displayed.

```

Spyder (Python 3.6)
Arquivo Editar Pesquisar Código Executar Depurar Consoles Projetos Ferramentas Ver Ajuda
Editor - F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\Co... × Console IPython
Questao_19_Lista_Pessoas.py Questao_20_Lista_Numerica.py × Console 1/A ×
In [66]: runfile('F:/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/TP3/Co...', wdir='F:/Infnet/EDS_2017')
Digite um número: 6
[6]
Digite um número: 15
[6, 15]
Digite um número: 28
[6, 15, 28]
Digite um número: 65
[6, 15, 28, 65]
Digite um número: 99
[6, 15, 28, 65, 99]
O elemento de maior valor é: 99

```

**21. Escreva um programa em Python que leia as notas de uma turma de 10 estudantes e depois imprima as notas que são maiores do que a média da turma.**

```

Spyder (Python 3.6)
Arquivo Editar Pesquisar Código Executar Depurar Consoles Projetos Ferramentas Ver Ajuda
Editor - F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\Co... ✎ | Console IPython
Questao_20_Lista_Numerica.py Questao_21_Lista_Notas.py ✎ | Console 1/A ✎
1 notas = []
2 soma = 0
3 contador = 0
4 mediaTurma = 0
5 while contador <= 10:
6     nota = float(input('Digite a nota do aluno: '))
7     notas.append(nota)
8     print('=====')
9     soma = sum(notas)
10    mediaTurma = soma/len(notas)
11    contador = contador + 1
12 print('A média da turma é:', mediaTurma)
13 for count in notas:
14     if count > mediaTurma:
15         print('Aprovados:', count)
16
17
18
19
Dígite a nota do aluno: 10
=====
Dígite a nota do aluno: 9.5
=====
Dígite a nota do aluno: 9
=====
Dígite a nota do aluno: 8.5
=====
Dígite a nota do aluno: 8
=====
Dígite a nota do aluno: 7.5
=====
Dígite a nota do aluno: 7
=====
Dígite a nota do aluno: 10
=====
Dígite a nota do aluno: 9
=====
A média da turma é: 8.75
Aprovados: 10.0
Aprovados: 9.5
Aprovados: 9.0
Aprovados: 10.0
Aprovados: 9.0
Aprovados: 9.0
Aprovados: 9.0

```

**22. Escreva um programa em Python que pergunte repetidamente se o usuário quer adicionar (a), remover (r), modificar (m) ou imprimir (i) elementos de uma lista. Adicione também a opção de sair(s), para sair do programa. Dependendo da resposta, o seu programa deve realizar a operação desejada. A lista inicialmente encontra-se vazia.**

```

Spyder (Python 3.6)
Arquivo Editar Pesquisar Código Executar Depurar Consoles Projetos Ferramentas Ver Ajuda
Editor - F:\Infnet\EDS_2017\2º_Bloco_ARQUITETURA_COMP_OS\Projeto_Bloco\TP3\Co... ✎ | Console IPython
Questao_21_Lista_Notas.py Questao_22_Lista_Opcoes.py ✎ | Console 1/A ✎
1 lista = []
2 a = 'a' # adiciona elementos
3 r = 'r' # remove elementos
4 m = 'm' # modifica elementos
5 i = 'i' # imprime a lista
6 s = 's' # sair do programa
7 while True:
8     opcao = input('Digite uma opção: a, r, m, i ou s: ')
9     if opcao == 'a':
10         elemento = input('Digite um elemento: ')
11         lista.append(elemento)
12         print(lista)
13     if opcao == 'r':
14         elemento = input('Digite um elemento: ')
15         lista.remove(elemento)
16         print(lista)
17     if opcao == 'm':
18         elemento = input('Digite um elemento: ')
19         elementoAtual = input('Digite um elemento existente: ')
20         lista[elementoAtual] = elemento
21         print(lista)
22     if opcao == 'i':
23         #elemento = input('Digite um elemento: ')
24         #for count in range(len(lista)):
25         #    print(count + 1, lista[count])
26     if opcao == 's':
27         print('Saindo do programa...')
28         break
Dígite uma opção: a, r, m, i ou s: a
Dígite um elemento: abacaxi
['abacaxi']
Dígite uma opção: a, r, m, i ou s: a
Dígite um elemento: banana
['abacaxi', 'banana']
Dígite uma opção: a, r, m, i ou s: a
Dígite um elemento: morango
['abacaxi', 'banana', 'morango']
Dígite uma opção: a, r, m, i ou s: r
Dígite um elemento: morango
['abacaxi', 'banana']
Dígite uma opção: a, r, m, i ou s: m
Dígite um elemento: laranja
Dígite um elemento existente: banana
['abacaxi', 'laranja']
Dígite uma opção: a, r, m, i ou s: i
1 abacaxi
2 laranja
Dígite uma opção: a, r, m, i ou s: s
Saindo do programa...

```

### 3.4. Teste de Performance – TP4

Escreva um programa em Python que exiba graficamente através do uso do módulo Pygame:

Uma barra de indicação da porcentagem do uso de memória;

Uma barra de indicação da porcentagem do uso de CPU;

Uma barra de indicação da porcentagem do uso de disco;

Um texto com a informação do IP da máquina.

Todas as barras e informações devem estar na mesma tela, podendo fazer o uso de superfícies ou surfaces (recomendado).

### 3.4.1. Código

```

1  # -*- coding: utf-8 -*-
2 """
3     Created on Mon Jan  8 22:42:19 2018
4
5     @author: Alexander.Silva
6 """
7
8 import ...
11
12 pygame.init()
13
14 # Iniciando a janela principal
15 larguraTela = 800
16 alturaTela = 600
17 tela = pygame.display.set_mode((larguraTela, alturaTela))
18 pygame.display.set_caption('Uso de memória')
19 pygame.display.init()
20
21 # Cores:
22 vermelha = (255, 0, 0)
23 azul = (0, 0, 255)
24 branca = (255, 255, 255)
25 preta = (0, 0, 0)
26
27 # Cria relógio
28 clock = pygame.time.Clock()
29 # Contador de tempo
30 count = 60
31
32 # Superfícies para mostrar as informações:
33 s1 = pygame.surface.Surface((larguraTela, alturaTela/3))
34 s2 = pygame.surface.Surface((larguraTela, alturaTela/3))
35 s3 = pygame.surface.Surface((larguraTela, alturaTela/3))
36
37 # Inicializa fonte do sistema
38 pygame.font.init()
39 fonte = pygame.font.Font(None, 28)
40 fonteBarra = pygame.font.Font(None, 24)
41
42 # Mostra a capacidade de MP em uso e total:
43 def mostraUsoMemoria():
44     mem = psutil.virtual_memory()
45     larg = larguraTela-2*20
46     tela.fill(preta)
47     pygame.draw.rect(s1, azul, (20,50,larg,70))
48     tela.blit(s1, (0,0))
49     larg = larg*mem.percent/100
50     pygame.draw.rect(s1, vermelha, (20,50,larg,70))

```

## Continuação...

```

1: Proj
  51     tela.blit(sl,(0,0))
  52     total = round(mem.total/(1024*1024*1024),2)
  53     totalEmUso = round(mem.used/(1024*1024*1024),2)
  54     textoMemTopo = fonte.render('Capacidade total MP: '
  55                                     + str(total) + ' GB', 1, branca)
  56     textoBarra = fonteBarra.render(str(totalEmUso)
  57                                     + ' GB', 1, branca)
  58     tela.blit(textoMemTopo,(20,10))
  59     tela.blit(textoBarra,(200,75))
  60
  61     # Mostra a capacidade do disco rígido em uso e total:
  62     def mostraUsoDisco():
  63         disco = psutil.disk_usage('.')
  64         larg = larguraTela-2*20
  65         pygame.draw.rect(s2, azul, (20,50, larg,70))
  66         tela.blit(s2, (0,alturaTela/3))
  67         larg = larg*disco.percent/100
  68         pygame.draw.rect(s2, vermelha, (20,50, larg,70))
  69         tela.blit(s2, (0,alturaTela/3))
  70         total = round(disco.total/(1024*1024*1024),2)
  71         discoUso = round(disco.used/(1024*1024*1024),2)
  72         textoDiscoTopo = fonte.render('Capacidade total Disco Rígido: '
  73                                     + str(total) + ' GB', 1, branca)
  74         textoBarra = fonteBarra.render(str(discoUso)
  75                                     + ' GB', 1, branca)
  76         tela.blit(textoDiscoTopo,(20, alturaTela/3))
  77         tela.blit(textoBarra,(25, alturaTela/3 + 72))
  78
  79     # Mostra o percentual de uso da CPU:
  80     def mostraUsoCpu():
  81         cpu = psutil.cpu_percent(interval=None, percpu = False)
  82         percentCpu = round(cpu,2)
  83         larg = larguraTela-2*20
  84         pygame.draw.rect(s3, azul, (20,50, larg,70))
  85         tela.blit(s3, (0, 2*alturaTela/3))
  86         larg = (larg*cpu)/100
  87         pygame.draw.rect(s3, vermelha, (20,50, larg,70))
  88         tela.blit(s3, (0, 2*alturaTela/3))
  89         textoCpuTopo = fonte.render('Percentual corrente CPU: '
  90                                     + str(percentCpu) + ' %', 1, branca)
  91         tela.blit(textoCpuTopo,(20, 2*alturaTela/3))
  92
  93     # Mostra o ip da interface de rede local:
  94     def mostraIpEmUso():
  95         dicIpAdress = psutil.net_if_addrs()
  96         textoIpAdress = fonte.render('Rede local Ip: '
  97                                     + dicIpAdress['Ethernet'][1].address, 1, branca)
  98         tela.blit(textoIpAdress,(20, 550))
mostraUsoCpu()

```

## Continuação...

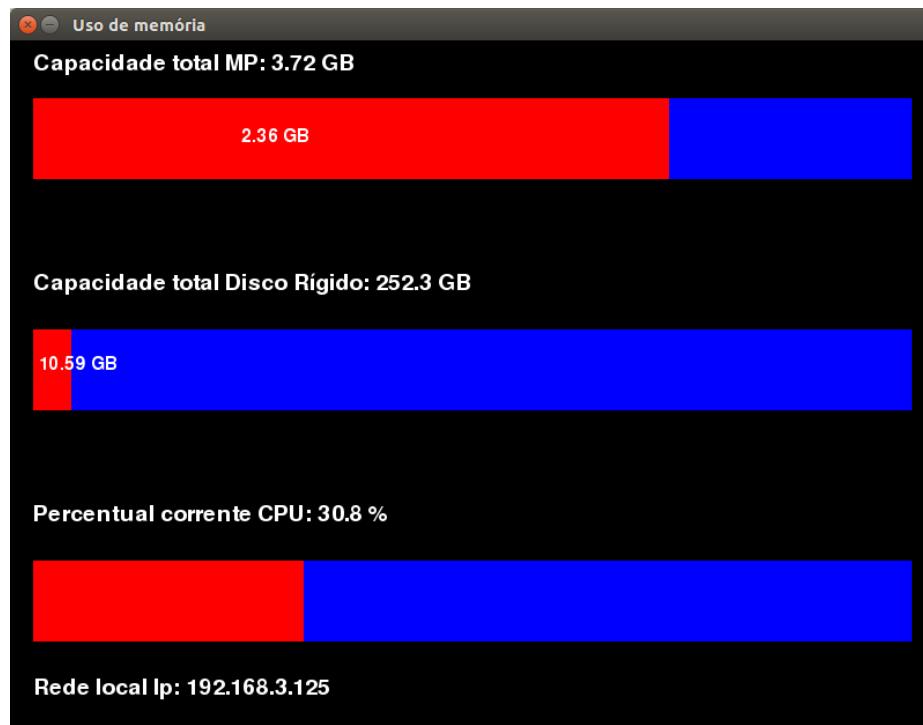
```

99
100     # Repetição para capturar eventos e atualizar tela
101     sair = False
102     while not sair:
103         # Checar os eventos do mouse aqui:
104         for event in pygame.event.get():
105             if event.type == pygame.QUIT:
106                 sair = True
107         # Fazer a atualização a cada segundo:
108         if count == 60:
109             mostraUsoMemoria()
110             mostraUsoDisco()
111             mostraUsoCpu()
112             mostraIpEmUso()
113             count = 0
114         # 60 frames por segundo
115         count = count + 1
116         clock.tick(60)
117         # Atualiza o desenho na tela
118         pygame.display.update()
119         # Finaliza a janela
120         pygame.display.quit()
121         pygame.quit()

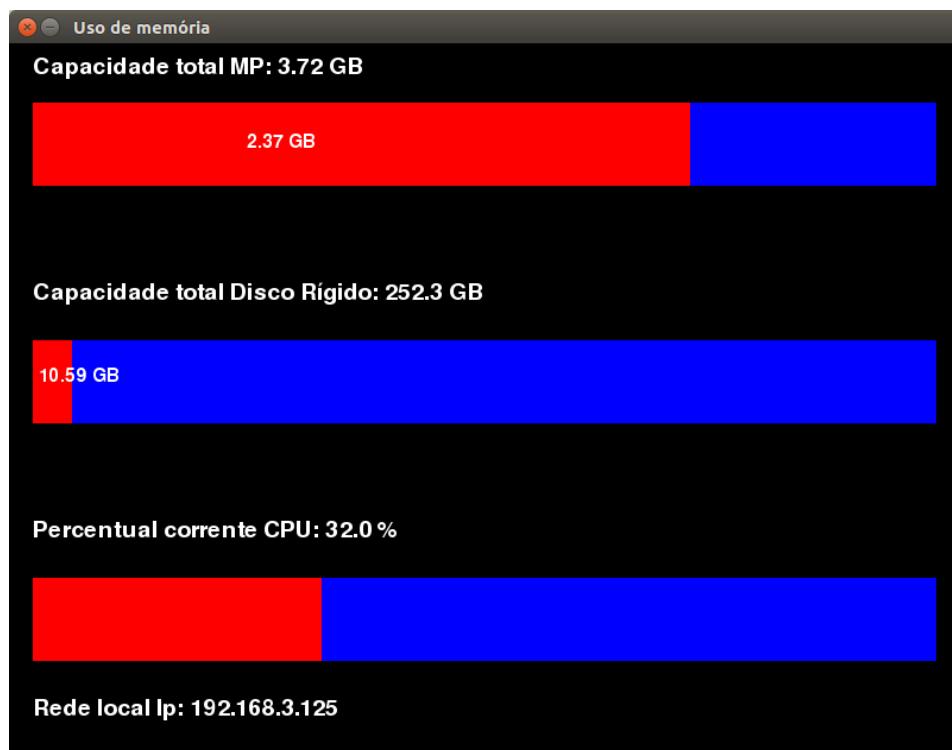
```

### 3.4.2. Execuções

#### 3.4.2.1. Primeira



### 3.4.2.2. Segunda



### 3.4.2.3. Terceira



### 3.5. Teste de Performance - TP5

Adicione ao seu programa feito no TP4 informações mais detalhadas de uso de CPU. Você pode adicionar as informações da maneira que achar mais interessante. No entanto, algumas são obrigatórias:

- Alterar a barra de CPU do TP4 para ter barras de CPU associadas a cada núcleo (*core*);
- Adicionar informação de nome/modelo da CPU (*brand*);
- Adicionar informação do tipo da arquitetura (*arch*);
- Adicionar informação da palavra do processador (*bits*);
- Adicionar informação sobre a frequência total e frequência de uso da CPU;
- Adicionar informação do número total de núcleos (núcleo físico) e threads (núcleo lógico).

### 3.5.1. Código

Spyder (Python 3.6)

Arquivo Editar Pesquisar Código Executar Depurar Consoles Projetos Ferramentas Ver

Editor - /home/alex/Documentos/Projeto\_Bloco/TP5/MonitorSistema\_v0.3.py

MonitorSistemaPsutil.py X MonitorSistema\_v0.3.py X

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Jan 11 22:00:19 2018
4
5 @author: Alexander.Silva
6 """
7 import pygame
8 import psutil
9 import cpuinfo
10
11 pygame.init()
12 # Cores:
13 preto = (0, 0, 0)
14 branco = (255, 255, 255)
15 cinza = (100, 100, 100)
16 azul = (0,0,255)
17 vermelho = (255,0,0)
18 # Iniciando a janela principal
19 larguraTela = 800
20 alturaTela = 600
21 tela = pygame.display.set_mode((larguraTela, alturaTela))
22 pygame.display.set_caption("Monitor de Sistema")
23 pygame.display.init()
24 # Superfícies para mostrar as informações:
25 s0 = pygame.surface.Surface((larguraTela, alturaTela/5))
26 s1 = pygame.surface.Surface((larguraTela, alturaTela/5))
27 s2 = pygame.surface.Surface((larguraTela, alturaTela/5))
28 s3 = pygame.surface.Surface((larguraTela, alturaTela/5))
29 s4 = pygame.surface.Surface((larguraTela, alturaTela/5))
30 # Inicializa fonte do sistema
31 pygame.font.init()
32 font = pygame.font.Font(None, 24)
33 # Cria relógio
34 clock = pygame.time.Clock()
35 # Contador de tempo
36 cont = 60
37 # Obtém informações da CPU
38 infoCpu = cpuinfo.get_cpu_info()
```

Pern

## Continuação...



```

39 # Retorna uma lista de percentual de uso de cada CPU:
40 listCpuPercent = psutil.cpu_percent(interval=1, percpu=True)
41
42 # Mostra as informações de CPU escolhidas:
43 def mostraInfoCpu():
44     s0.fill(branco)
45     mostraTexto(s0, "Nome:", "brand", 10)
46     mostraTexto(s0, "Arquitetura:", "arch", 30)
47     mostraTexto(s0, "Palavra (bits):", "bits", 50)
48     mostraTexto(s0, "Frequência (MHz):", "freq", 70)
49     mostraTexto(s0, "Núcleos (físicos):", "nucleos", 90)
50     tela.blit(s0, (0, 0))
51
52 # Mostra texto de acordo com uma chave:
53 def mostraTexto(s0, nome, chave, pos_y):
54     text = font.render(nome, True, preto)
55     s0.blit(text, (10, pos_y))
56     if chave == "freq":
57         s1 = str(round(psutil.cpu_freq().current, 2))
58     elif chave == "nucleos":
59         s1 = str(psutil.cpu_count())
60         s1 = s1 + " (" + str(psutil.cpu_count(logical=True)) + ")"
61     else:
62         s1 = str(infoCpu[chave])
63     text = font.render(s1, True, cinza)
64     s0.blit(text, (160, pos_y))
65
66 # Mostra o percentual de uso de CPU:
67 def mostraUsoCpu(s1, listCpuPercent):
68     s1.fill(cinza)
69     numCpu = len(listCpuPercent)
70     x = y = 10
71     desl = 10
72     alt = s0.get_height()-2*y
73     larg = (s0.get_width()-2*x - (numCpu+1)*desl)/numCpu
74     d = x + desl
75     for i in listCpuPercent:
76         pygame.draw.rect(s0, vermelho, (d, y, larg, alt))
    
```

Perm



```

76     pygame.draw.rect(s0, vermelho, (d, y, larg, alt))
77     pygame.draw.rect(s0, azul, (d, y, larg, (1-i/100)*alt))
78     d = d + larg + desl
79     tela.blit(s0, (0, alturaTela/5))
80     textoCpu1 = font.render('Core 1: ' + str(listCpuPercent[0]) + ' %', 1, branco)
81     tela.blit(textoCpu1,(60, alturaTela/5 + 50))
82     textoCpu2 = font.render('Core 2: ' + str(listCpuPercent[1]) + ' %', 1, branco)
83     tela.blit(textoCpu2,(260, alturaTela/5 + 50))
84     textoCpu3 = font.render('Core 3: ' + str(listCpuPercent[2]) + ' %', 1, branco)
85     tela.blit(textoCpu3,(460, alturaTela/5 + 50))
86     textoCpu4 = font.render('Core 4: ' + str(listCpuPercent[3]) + ' %', 1, branco)
87     tela.blit(textoCpu4,(640, alturaTela/5 + 50))
88
89 # Mostra a capacidade de MP em uso e total:
90 def mostraUsoMemoria():
91     mem = psutil.virtual_memory()
92     largBarraMem = larguraTela-2*20
93     s2.fill(cinza)
94     pygame.draw.rect(s2, azul, (20,30, largBarraMem,70))
95     tela.blit(s2, (0,2*alturaTela/5))
96     largBarraMem = largBarraMem*mem.percent/100
97     pygame.draw.rect(s2, vermelho, (20,30, largBarraMem,70))
98     tela.blit(s2,(0,2*alturaTela/5))
99     total = round(mem.total/(1024*1024*1024),2)
100    totalEmUso = round(mem.used/(1024*1024*1024),2)
101    textoMemTopo = font.render('Capacidade total MP: '
102                                + str(total) + ' GB', 1, branco)
103    textoBarra = font.render(str(totalEmUso) + ' GB', 1, branco)
104    tela.blit(textoMemTopo,(20,2*alturaTela/5))
105    tela.blit(textoBarra,(250,300))
106
107 # Mostra a capacidade do disco rígido em uso e total:
108 def mostraUsoDisco():
109     disco = psutil.disk_usage('.')
110     largBarraDisco = larguraTela-2*20
111     s3.fill(cinza)
112     pygame.draw.rect(s3, azul, (20,30, largBarraDisco,70))
113     tela.blit(s3, (0,3*alturaTela/5))
    
```

Permissõ

## Continuação...



```

114     largBarraDisco = largBarraDisco*disco.percent/100
115     pygame.draw.rect(s3, vermelho, (20,30, largBarraDisco, 70))
116     tela.blit(s3,(0,3*alturaTela/5))
117     total = round(disco.total/(1024*1024*1024),2)
118     discoUsa = round(disco.used/(1024*1024*1024),2)
119     textoDiscoTopo = font.render('Capacidade total Disco Rígido: '
120                                     + str(total) + ' GB', 1, branco)
121     textoBarra = font.render(str(discoUsa) + ' GB', 1, branco)
122     tela.blit(textoDiscoTopo,(20, 3*alturaTela/5))
123     tela.blit(textoBarra,(25, 3*alturaTela/5 + 60))
124
125 # Mostra o ip da interface de rede local:
126 def mostraIp():
127     diciIpAdress = psutil.net_if_addrs()
128     s4.fill(branco)
129     tela.blit(s4,(0,4*alturaTela/5))
130     textoIpAdress = font.render('Rede local Ip: '
131                               + diciIpAdress['wlp2s0'][0].address, 1, preto)
132     tela.blit(textoIpAdress,(20, 520))
133
134 # Repetição para capturar eventos e atualizar tela
135 sair = False
136 while not sair:
137     # Checar os eventos do mouse aqui:
138     for event in pygame.event.get():
139         if event.type == pygame.QUIT:
140             sair = True
141
142     # Fazer a atualização a cada segundo:
143     if cont == 60:
144         mostraInfoCpu()
145         mostraUsoCpu(s0, listCpuPercent)
146         mostraUsoMemoria()
147         mostraUsoDisco()
148         mostraIp()
149         cont = 0
150
151     # Atualiza o desenho na tela
152     pygame.display.update()
153
154     # 60 frames por segundo
155     clock.tick(60)
156     cont = cont + 1
157
158 # Finaliza a janela
159 pygame.display.quit()
160 pygame.quit()

```

Perr



```

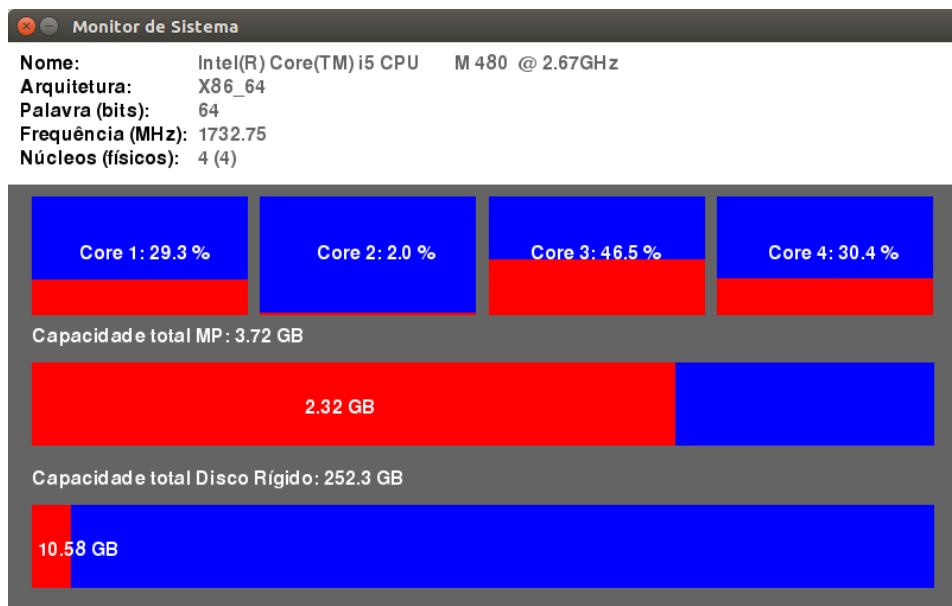
151     # Atualiza o desenho na tela
152     pygame.display.update()
153
154     # 60 frames por segundo
155     clock.tick(60)
156     cont = cont + 1
157
158 # Finaliza a janela
159 pygame.display.quit()
160 pygame.quit()

```

Perr

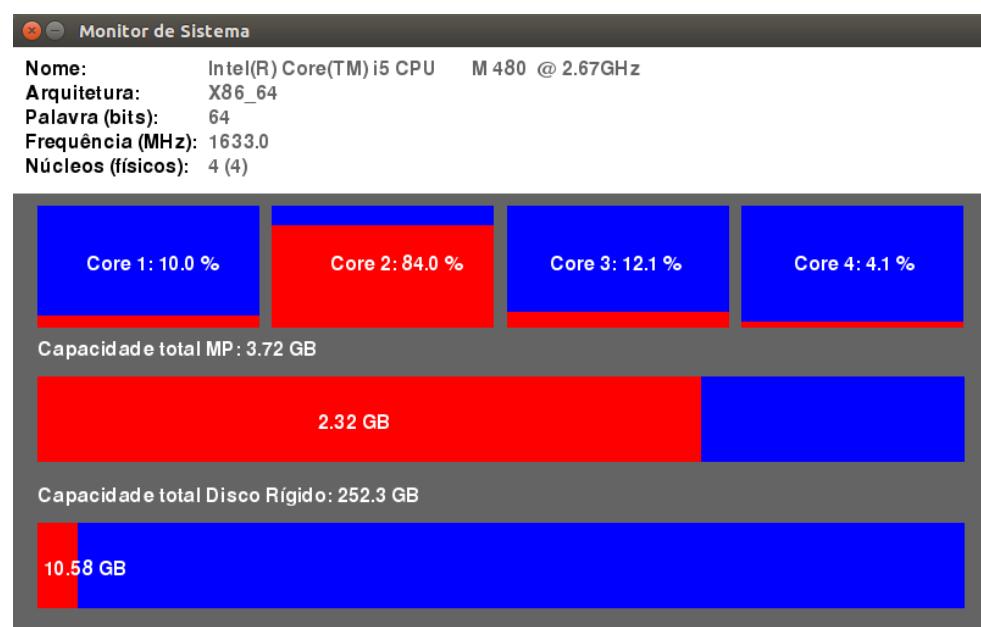
### 3.5.2. Execuções

#### 3.5.2.1. Primeira



Rede local ip: 192.168.3.125

#### 3.5.2.2. Segunda



Rede local ip: 192.168.3.125

### 3.6. Teste de Performance – TP6

Este TP6 corresponde à continuação do TP5. Agora, você irá introduzir no seu programa informações sobre arquivos e diretórios especificados e sobre processos em execução no computador. A partir de então, você terá mais liberdade para criar a forma de visualização das informações da maneira que desejar e de acordo com seu orientador. Certifique-se apenas que está realizando o básico do que este TP requisita.

seguir, os requisitos básicos:

Criar uma ou mais funções que retornem ou apresentem informações sobre diretórios e arquivos. Tais informações podem ser qualquer uma que você achar relevante disponível no módulo ‘os’ e ‘psutil’ de Python, como nome, tamanho, localização, data de criação, data de modificação, tipo, etc.

Usar a função em seu programa para mostrar o resultado. O resultado pode ser em texto formatado impresso na tela ou gráfico, usando o Pygame. Note que o uso do Pygame é opcional.

Criar uma ou mais funções que retornem ou apresentem informações sobre processos do sistema. As informações podem ser: PID, nome do executável, consumo de processamento, consumo de memória, entre outras disponíveis no módulo ‘psutil’ de Python.

Usar a função em seu programa para mostrar o resultado. O resultado pode ser em texto formatado impresso na tela ou gráfico, usando o Pygame. Note que o uso do Pygame é opcional.

### 3.6.1. Código

```
[~/PycharmProjects/infnet] - .../MonitorSistema_v0.6.py [infnet] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
infnet > MonitorSistema_v0.6.py
teste.py x MonitorSistema_v0.5.py x MonitorSistema_v0.6.py x
1 # -*- coding: utf-8 -*-
2 """
3     Created on Mon Jan 11 22:00:19 2018
4     @author: Alexander.Silva
5 """
6 import pygame, psutil, cpuinfo, os, time, datetime
7
8 pygame.init()
9 # Cores:
10 preto = (0, 0, 0)
11 branco = (255, 255, 255)
12 cinza = (100, 100, 100)
13 azul = (0, 0, 255)
14 vermelho = (255, 0, 0)
15 # Iniciando a janela principal
16 larguraTela = 800
17 alturaTela = 600
18 tela = pygame.display.set_mode((larguraTela, alturaTela))
19 pygame.display.set_caption("Monitor de Sistema")
20 pygame.display.init()
21 # Superfícies para mostrar as informações:
22 s0 = pygame.surface.Surface((larguraTela, alturaTela/6))
23 s1 = pygame.surface.Surface((larguraTela, alturaTela/6))
24 s2 = pygame.surface.Surface((larguraTela, alturaTela/6))
25 s3 = pygame.surface.Surface((larguraTela, alturaTela/6))
26 s4 = pygame.surface.Surface((larguraTela, alturaTela/6))
27 s5 = pygame.surface.Surface((larguraTela, alturaTela/6))
28 # Inicializa fonte do sistema
29 pygame.font.init()
30 font = pygame.font.Font(None, 22)
31 # Cria relógio
32 clock = pygame.time.Clock()
33 # Contador de tempo
34 cont = 60
35 # Obtém informações da CPU
36 infoCpu = cpuinfo.get_cpu_info()
37 # Retorna uma lista de percentual de uso de cada CPU:
38 listCpuPercent = psutil.cpu_percent(interval=1, percpu=True)
39
40 # Mostra as informações de CPU escolhidas:
41 def mostraInfoCpu():
42     s0.fill(branco)
43     mostraTexto(s0, "Nome:", "brand", 3)
44     mostraTexto(s0, "Arquitetura:", "arch", 23)
45     mostraTexto(s0, "Palavra (bits):", "bits", 43)
46     mostraTexto(s0, "Frequência (MHz):", "freq", 63)
47     mostraTexto(s0, "Núcleos (físicos):", "nucleos", 83)
48     tela.blit(s0, (0, 0))
```

## Continuação...

```

~/PycharmProjects/infnet] - .../MonitorSistema_v0.6.py [infnet] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
infnet > MonitorSistema_v0.6.py
teste.py x MonitorSistema_v0.5.py x MonitorSistema_v0.6.py x

49
50     # Mostra texto de acordo com uma chave:
51     def mostraTexto(s0, nome, chave, pos_y):
52         text = font.render(nome, True, preto)
53         s0.blit(text, (10, pos_y))
54         if chave == "freq":
55             s1 = str(round(psutil.cpu_freq().current, 2))
56         elif chave == "nucleos":
57             s1 = str(psutil.cpu_count())
58             s1 = s1 + " (" + str(psutil.cpu_count(logical=True)) + ")"
59         else:
60             s1 = str(infoCpu[chave])
61         text = font.render(s1, True, cinza)
62         s0.blit(text, (160, pos_y))
63
64     # Mostra o percentual de uso de CPU:
65     def mostraUsoCpu(s1, listCpuPercent):
66         s1.fill(cinza)
67         numCpu = len(listCpuPercent)
68         x = y = 10
69         desl = 10
70         alt = s0.get_height()-2*y
71         larg = (s0.get_width()-2*x - (numCpu+1)*desl)/numCpu
72         d = x + desl
73         for i in listCpuPercent:
74             pygame.draw.rect(s0, vermelho, (d, y, larg, alt))
75             pygame.draw.rect(s0, azul, (d, y, larg, (1-i/100)*alt))
76             d = d + larg + desl
77             tela.blit(s0, (0, alturaTela/6))
78             textoCpu1 = font.render('Core 1: ' + str(listCpuPercent[0]) + ' %', 1, branco)
79             tela.blit(textoCpu1,(60, alturaTela/6 + 50))
80             textoCpu2 = font.render('Core 2: ' + str(listCpuPercent[1]) + ' %', 1, branco)
81             tela.blit(textoCpu2,(260, alturaTela/6 + 50))
82             textoCpu3 = font.render('Core 3: ' + str(listCpuPercent[2]) + ' %', 1, branco)
83             tela.blit(textoCpu3,(440, alturaTela/6 + 50))
84             textoCpu4 = font.render('Core 4: ' + str(listCpuPercent[3]) + ' %', 1, branco)
85             tela.blit(textoCpu4,(640, alturaTela/6 + 50))
86
87     # Mostra a capacidade de MP em uso e total:
88     def mostraUsoMemoria():
89         mem = psutil.virtual_memory()
90         largBarraMem = larguraTela-2*20
91         s2.fill(cinza)
92         pygame.draw.rect(s2, azul, (20,30, largBarraMem,50))
93         tela.blit(s2, (0,2*alturaTela/6))
94         largBarraMem = largBarraMem*mem.percent/100
95         pygame.draw.rect(s2, vermelho, (20,30, largBarraMem,50))
96         tela.blit(s2,(0,2*alturaTela/6))

```

## Continuação...

### Requisitos I e II:

```

~/PycharmProjects/infnet] - .../MonitorSistema_v0.6.py [infnet] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
infnet > MonitorSistema_v0.6.py
teste.py x MonitorSistema_v0.5.py x MonitorSistema_v0.6.py x
97     total = round(mem.total/(1024*1024*1024),2)
98     totalEmUso = round(mem.used/(1024*1024*1024),2)
99     textoMemTopo = font.render('Capacidade total MP: '
100                               + str(total) + ' GB', 1, branco)
101    textoBarra = font.render(str(totalEmUso) + ' GB', 1, branco)
102    tela.blit(textoMemTopo,(20,2*alturaTela/6 + 10))
103    tela.blit(textoBarra,(250,300))
104
105    # Mostra a capacidade do disco rígido em uso e total:
106    def mostraUsoDisco():
107        lista_info_particao = psutil.disk_partitions()
108        disco = psutil.disk_usage('.')
109        largBarraDisco = larguraTela-2*20
110        s3.fill(cinza)
111        pygame.draw.rect(s3, azul, (20,30, largBarraDisco,50))
112        tela.blit(s3, (0,3*alturaTela/6))
113        largBarraDisco = largBarraDisco*disco.percent/100
114        pygame.draw.rect(s3, vermelho, (20,30, largBarraDisco,50))
115        tela.blit(s3,(0,3*alturaTela/6))
116        total = round(disco.total/(1024*1024*1024),2)
117        discoUso = round(disco.used/(1024*1024*1024),2)
118        textoDiscoTopo = font.render('Capacidade total Disco Rígido: '
119                               + str(total) + ' GB', 1, branco)
120        textoBarra = font.render(str(discoUso) + ' GB', 1, branco)
121        textoInfoParticao = font.render('| Dispositivo: ' + str(lista_info_particao[0].device)
122                                       + ' | Sistema de Arquivo: '
123                                       + str(lista_info_particao[0].fstype), 1, branco)
124        tela.blit(textoInfoParticao, (350, 3*alturaTela/6 + 10))
125        tela.blit(textoDiscoTopo,(20, 3*alturaTela/6 + 10))
126        tela.blit(textoBarra,(25, 3*alturaTela/6 + 50))
127
128    # Mostra o ip da interface de rede local:
129    def mostraIp():
130        dicIpAdress = psutil.net_if_addrs()
131        s4.fill(branco)
132        tela.blit(s4,(0, 4*alturaTela/6))
133        textoIpAdress = font.render('Rede local Ip: '
134                               + dicIpAdress['enp0s7'][0].address, 1, preto)
135        tela.blit(textoIpAdress,(20, 410))
136
137    def mostraInfoDir():
138        #s4.fill(branco)
139        #tela.blit(s4, (0, 4*alturaTela/5))
140
141        lista = os.listdir()
142        dic = {}

```

## Continuação...

### Requisitos III e IV:

**MonitorSistema\_v0.6.py**

```

144     for i in lista:
145         if os.path.isfile(i):
146             dic[i] = []
147             dic[i].append(os.stat(i).st_size)
148             dic[i].append(os.stat(i).st_ctime)
149             dic[i].append(os.stat(i).st_mtime)
150
151         tituloInfo = 'Arquivos e Diretórios:'
152         titulo_tamanho = '{:>5}'.format('Tamanho')
153         titulo_data_criacao = '{:>30}'.format('Data de Criação')
154         titulo_data_modificacao = '{:>38}'.format('Data de Modificação')
155         titulo = titulo_tamanho + titulo_data_criacao + titulo_data_modificacao
156         #print(titulo)
157         textoTituloInfo = font.render(tituloInfo, 1, preto)
158         tela.blit(textoTituloInfo, (20, 430))
159         textoTitulo = font.render(titulo, 1, preto)
160         tela.blit(textoTitulo, (20, 450))
161
162         for i in dic:
163             kb = dic[i][0]/1024
164             tamanho = '{:>10}'.format(str('{:.2f}'.format(kb) + 'KB'))
165             time_create = '{:>30}'.format(time.ctime(dic[i][0]))
166             time_mod = '{:>30}'.format(time.ctime(dic[i][1]))
167             nomeArquivo = '{:>30}'.format(i)
168             textoArqDir = font.render(tamanho + time_create + time_mod
169                                     + nomeArquivo, 1, preto)
170             tela.blit(textoArqDir, (20, 470))
171
172     def info_processo():
173         s5.fill(branco)
174         tela.blit(s5, (0, 5*alturaTela/6))
175         p = psutil.Process(pid_selecionado)
176         #print('\n'Nome do processo:', p.name())
177         nameProcess = p.name()
178         #print('Id do processo:', p.pid)

```

**MonitorSistema\_v0.6.py**

```

179     processId = p.pid
180     #print('Nome do usuário proprietário:', p.username())
181     userName = p.username()
182     data_criacao = datetime.datetime.fromtimestamp(p.create_time())\
183                 .strftime("%Y-%m-%d %H:%M:%S")
184     #print('Data de criação do processo:', data_criacao)
185     #print(p.memory_info())
186     memInfo = p.memory_info()
187     mem_usada = round(memInfo.rss/1024, 2)
188     #print('Memória usada:', mem_usada, 'KB')
189     textoNameProcess = font.render('Nome do processo: ' + nameProcess, 1, preto)
190     textoProcessId = font.render('Id do processo: ' + str(processId), 1, preto)
191     textoUserName = font.render('Nome do usuário proprietário: ' + userName, 1, preto)
192     textoDataCriacao = font.render('Data de criação: ' + data_criacao, 1, preto)
193     textoMemInfo = font.render('Memória usada: ' + str(mem_usada), 1, preto)
194     tela.blit(textoNameProcess, (20, 495))
195     tela.blit(textoProcessId, (20, 515))
196     tela.blit(textoUserName, (20, 535))
197     tela.blit(textoDataCriacao, (20, 555))
198     tela.blit(textoMemInfo, (20, 575))
199
200     list_pids = psutil.pids()
201     #print('\n'Lista de processos em execução: ', list_pids)
202
203     for i in list_pids:
204         pid_selecionado = i
205
206     if pid_selecionado in list_pids:
207         info_processo()
208     else:
209         print('PID selecionado não existe!')
210
211     # Repetição para capturar eventos e atualizar tela
212     sair = False
213     while not sair:

```

## Continuação...

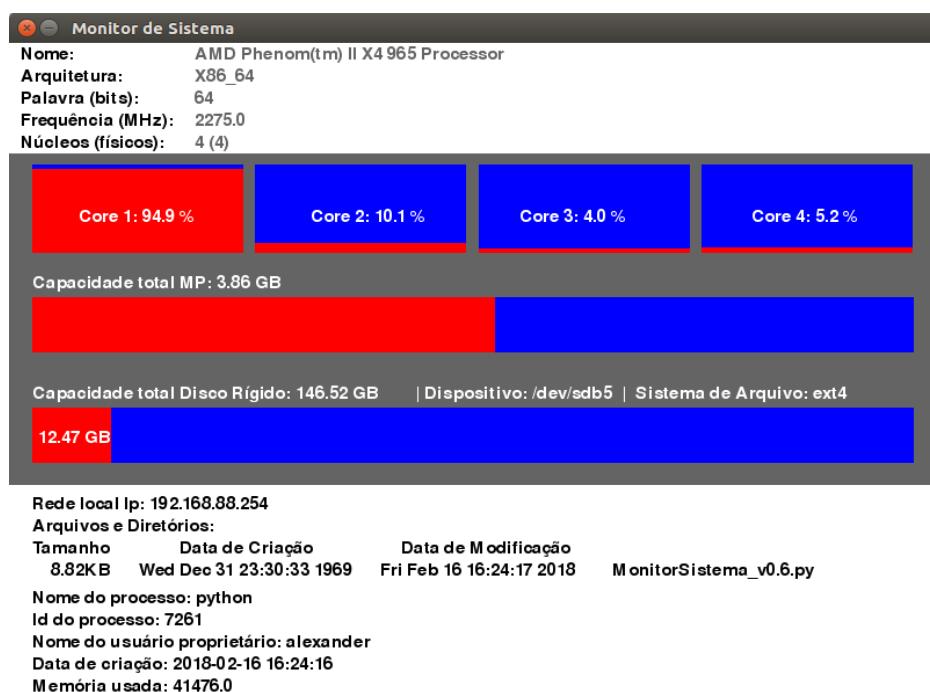
```

210 # Repetição para capturar eventos e atualizar tela
211 sair = False
212 while not sair:
213     # Checar os eventos do mouse aqui:
214     for event in pygame.event.get():
215         if event.type == pygame.QUIT:
216             sair = True
217
218     # Fazer a atualização a cada segundo:
219     if cont == 60:
220         mostraInfoCpu()
221         mostraUsoCpu(s0, listCpuPercent)
222         mostraUsoMemoria()
223         mostraUsoDisco()
224         mostraIp()
225         mostraInfoDir()
226         info_processo()
227         cont = 0
228
229     # Atualiza o desenho na tela
230     pygame.display.update()
231
232     # 60 frames por segundo
233     clock.tick(60)
234     cont = cont + 1
235
236     # Finaliza a janela
237     pygame.display.quit()
238     pygame.quit()
239

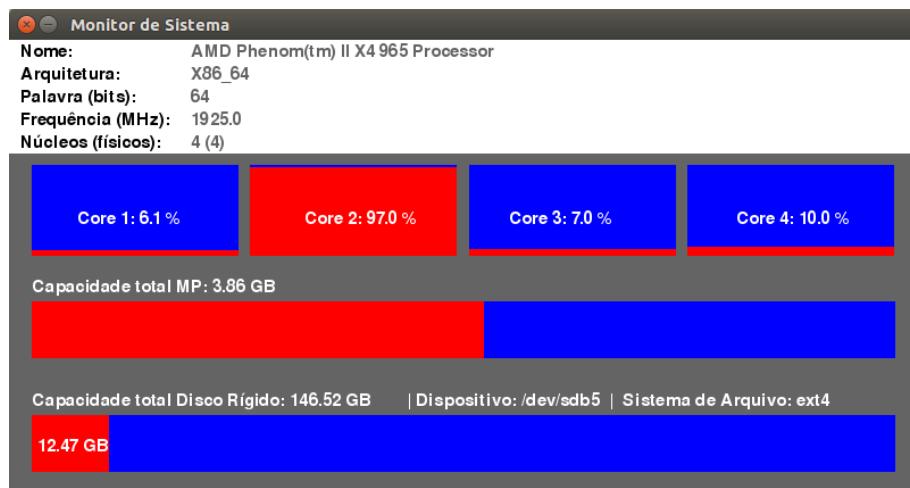
```

## 3.6.2. Execuções

### 3.6.2.1. Primeira



### 3.6.2.2. Segunda



Rede local Ip: 192.168.88.254

Arquivos e Diretórios:

Tamanho	Data de Criação	Data de Modificação	
8.82KB	Wed Dec 31 23:30:33 1969	Fri Feb 16 16:24:17 2018	MonitorSistema_v0.6.py

Nome do processo: python

Id do processo: 7312

Nome do usuário proprietário: alexander

Data de criação: 2018-02-16 16:27:20

Memória usada: 41592.0

### 3.7. Teste de Performance – TP7

O entregável do seu projeto de bloco será:

Um aplicativo simples de apresentação textual do monitoramento e análise de computadores em rede. Ele deverá ser implementado em Python usando módulos como psutil (para capturar dados do sistema computacional) e sockets (para criar cliente e servidor) e será desenvolvido de forma incremental durante o curso. A apresentação gráfica no lado cliente é opcional e pode ser parcial (parte gráfica e parte texto).

Faça todos os Testes de Performance, pois cada um deles é uma parte do seu Projeto de Bloco.

Este TP7 corresponde à continuação do TP6. Agora, você irá introduzir em seu programa informações sobre redes do computador. Escolha a forma de visualização das informações da maneira que desejar e de acordo com seu orientador. Certifique-se apenas que está realizando o básico do que este TP requisita.

A seguir, os requisitos básicos:

Criar uma ou mais funções que retornem ou apresentem ao menos 3 tipos de informações de redes. Tais informações podem ser qualquer uma que você achar relevante e que esteja disponível no módulo 'psutil' de Python, como características das interfaces (IP, gateway, máscara de subrede, etc.), uso de dados de rede por interface, uso de dados de rede por processos, etc.

Usar a função em seu programa para mostrar o resultado. O resultado pode ser em texto formatado impresso na tela ou gráfico, usando o Pygame. Note que o uso do Pygame é opcional.

Escrever um pequeno relatório com o código (e explicações que achar necessárias sobre ele), ao menos 3 exemplos de execução e justificativa das escolhas das informações a serem exibidas.

### 3.7.1. Justificativa

Este relatório descreve a implementação de um programa que consiste em obter informações de arquitetura, percentual de uso da cpu, memória, disco rígido, arquivos, diretórios, processos e rede. Para tal foi implementado 7 funções.

A abordagem deste relatório será sobre a função `mostra_info_rede()`, que através do módulo `psutil` em python é possível extrair do sistema informações relevantes. A primeira função do módulo `psutil` utilizada foi a `psutil.net_if_addrs()`, que retorna os endereços associados a cada NIC (placa de interface de rede) instalada no sistema como um dicionário cujas chaves são os nomes, e os valores do NIC é uma lista de tuplas nomeadas para cada endereço atribuído à NIC. Cada tupla nomeada inclui 5 campos:

**família:** a família de endereços, `AF_INET`, `AF_INET6` ou `sutil.AF_LINK`, que se refere a um endereço MAC.

**endereço:** o endereço NIC primário (sempre definido).

**máscara de rede:** o endereço de máscara de rede (pode ser `None`).

**broadcast:** o endereço de transmissão (pode ser `None`).

**ptp:** significa "ponto a ponto"; É o endereço de destino em uma interface ponto a ponto (normalmente uma VPN). A transmissão e o ptp são mutuamente exclusivos.

Destes campos foram implementados o família, endereço, máscara e broadcast.

Segunda função: `psutil.net_if_stats()`, devolve informações sobre cada NIC (placa de interface de rede) instalada no sistema como um dicionário cujas chaves são os nomes e os valores do NIC é uma tupla nomeada com os seguintes campos:

**isup:** um boolean que indica se a NIC está funcionando.

**duplex:** o tipo de comunicação duplex; pode ser `NIC_DUPLEX_FULL`, `NIC_DUPLEX_HALF` ou `NIC_DUPLEX_UNKNOWN`.

**velocidade:** a velocidade NIC expressa em mega bits (MB), se não puder ser determinada (por exemplo, 'localhost'), será configurado para 0.

**mtu:** unidade de transmissão máxima da NIC expressa em bytes.

Destes foram utilizados todos os campos, tendo em vista que são informações importantes no dia a dia.

Terceira função: psutil.net\_io\_counters(pernic=True), retorna as estatísticas de E / S da rede ao longo do sistema como uma tupla nomeada, incluindo os seguintes atributos:

**bytes\_sent:** número de bytes enviados

**bytes\_recv:** número de bytes recebidos

**packets\_sent:** número de pacotes enviados

**packets\_recv:** número de pacotes recebidos

**errin:** número total de erros ao receber

**Erro:** número total de erros durante o envio

**dropin:** número total de pacotes recebidos que foram descartados

**Desistência:** número total de pacotes de saída que foram descartados (sempre 0 no OSX e BSD)

Destes foram implementados os campos bytes sent/recv, e packets sent/recv, embora os demais também sejam relevantes.

Quarta e última função: psutil.net\_connections( ), retorna as conexões de socket do sistema como uma lista de tuplas com nome. Cada tupla nomeada fornece 7 atributos:

**fd**: o descritor do arquivo de socket. Se a conexão se refere ao processo atual, isso pode ser passado para `socket.fromfd()` para obter um objeto de soquete utilizável. No Windows e SunOS, isso sempre está configurado para **-1**.

**Família de endereços**, AF\_INET,AF\_INET6 ou AF\_UNIX.

**digite**: o tipo de endereço, SOCK\_STREAM ou SOCK\_DGRAM.

**laddr**: o endereço local como uma tupla nomeada ou um no caso de sockets AF\_UNIX. Para soquetes UNIX, veja as notas abaixo. **(ip, port) path**

**raddr**: o endereço remoto como uma tupla nomeada ou um absoluto no caso de soquetes UNIX. Quando o ponto final remoto não está conectado, você obterá uma tupla vazia (AF\_INET \*) ou (AF\_UNIX). Para soquetes UNIX, veja as notas abaixo. **(ip, port)path""**

**status**: representa o status de uma conexão TCP. O valor de retorno é uma **daspsutil.CONN \***constantes (uma string). Para os soquetes UDP e UNIX, isso sempre será **psutil.CONN\_NONE**.

**pid**: o PID do processo que abriu o soquete, se recuperável, de outra forma **None**. Em algumas plataformas (por exemplo, Linux), a disponibilidade desse campo muda de acordo com os privilégios de processo (a raiz é necessária).

Finalizando, os atributos usados foram fd, família, tipo, status e pid.

---

### **3.7.2.1. Código**

## **Requisitos I e II**

```
#@author: Alexander.Silva
import psutil, cpuinfo, os, time, datetime

# Obtém informações da CPU
infoCpu = cpuinfo.get_cpu_info()

# Retorna uma lista de percentual de uso de cada CPU
listCpuPercent = psutil.cpu_percent(interval=1, percpu=True)

# Retorna uma lista de informações das partições
lista_info_particao = psutil.disk_partitions()

# Mostra as informações de CPU escolhidas:
def mostra_info_cpu():

    dic = {}
    for i in infoCpu:

        brand = infoCpu['brand']
        arch = infoCpu['arch']
        bits = infoCpu['bits']
        dic['Nome'] = brand
        dic['Arquitetura'] = arch
        dic['Palavra'] = bits
    print()
    print('INFORMAÇÕES DA CPU:\n')
    print('Nome:', dic['Nome'], '\n' 'Arquitetura:', dic['Arquitetura'],
          '\n' 'Palavra (bits):', dic['Palavra'])
    freq = str(round(psutil.cpu_freq().current, 2))
    print('Frequência (MHz):', freq)
    nucleos = str(psutil.cpu_count())
    print('Núcleos (físicos):', nucleos + '(' + str(psutil.cpu_count(logical=True)) + ')')

print()

# Mostra o percentual de uso de CPU:
def mostra_uso_cpu(listCpuPercent):

    print('INFORMAÇÕES POR NÚCLEOS DA CPU:\n')

    for i in listCpuPercent:
        print('Núcleo: ' + str(i) + ' %')
    print()

# Mostra a capacidade de MP em uso e total:
def mostra_uso_memoria():

    mem = psutil.virtual_memory()
    total = round(mem.total/(1024*1024*1024),2)
    totalEmUso = round(mem.used/(1024*1024*1024),2)
    totalLivre = round(mem.free / (1024 * 1024 * 1024), 2)
    memPercent = round(mem.percent, 2)
    print('INFORMAÇÕES MEMÓRIA PRINCIPAL:\n')
    print('Capacidade total: ' + str(total) + ' GB')
    print('Capacidade em uso: ' + str(totalEmUso) + ' GB')
    print('Capacidade livre: ' + str(totalLivre) + ' GB')
    print('Percentual em uso: ' + str(memPercent) + ' %\n')

# Mostra a capacidade do disco rígido em uso e total:
def mostra_uso_disco():

    disco = psutil.disk_usage('.')
    total = round(disco.total/(1024*1024*1024),2)
    discoUso = round(disco.used/(1024*1024*1024),2)
    discoLivre = round(disco.free / (1024 * 1024 * 1024), 2)
    discoPercent = round(disco.percent, 2)
```

## Continuação...

```

MonitorSistema_v0.7.py x
67     print('INFORMAÇÕES DISCO RÍGIDO:\n')
68     print('Capacidade total: ' + str(total) + ' GB')
69     print('Capacidade em uso: ' + str(discoUso) + ' GB')
70     print('Capacidade livre: ' + str(discoLivre) + ' GB')
71     print('Percentual em uso: ' + str(discoPercent) + ' %\n')
72
73     def mostra_info_dir():
74
75         lista = os.listdir()
76         dic = {}
77
78         for i in lista:
79             if os.path.isfile(i):
80                 dic[i] = []
81                 dic[i].append(os.stat(i).st_size)
82                 dic[i].append(os.stat(i).st_ctime)
83                 dic[i].append(os.stat(i).st_mtime)
84
85         print('ARQUIVOS E DIRETÓRIOS:\n')
86         print('Dispositivo: ' + str(lista_info_particao[0].device))
87         print('Sistema de Arquivo: ' + str(lista_info_particao[0].fstype), '\n')
88         titulo_tamanho = '{:^10}'.format('TAMANHO')
89         titulo_data_criacao = '{:^25}'.format('DATA DE CRIAÇÃO')
90         titulo_data_modificacao = '{:^37}'.format('DATA DE MODIFICAÇÃO')
91         titulo_nome = '{:^27}'.format('NOME')
92         titulo = titulo_tamanho + titulo_data_criacao + titulo_data_modificacao + titulo_nome
93         print(titulo)
94
95         for i in dic:
96             kb = dic[i][0]/1024
97             tamanho = '{:^9}'.format(str('{:.2f}'.format(kb)) + ' KB')
98             time_create = '{:^30}'.format(time.ctime(dic[i][0]))

```

```

MonitorSistema_v0.7.py x
99     time_mod = '{:^30}'.format(time.ctime(dic[i][1]))
100    nomeArquivo = '{:^33}'.format(i)
101    print()
102    print(tamanho + time_create + time_mod + nomeArquivo)
103
104    def mostra_info_processo():
105
106        print('\n''PROCESSOS: ')
107        p = psutil.Process(pid_selecionado)
108        print('\n''Id do processo:', p.pid)
109        print('Nome do processo:', p.name())
110        print('Nome do usuário proprietário:', p.username())
111        data_criacao = datetime.datetime.fromtimestamp(p.create_time()).strftime("%Y-%m-%d %H:%M:%S")
112        print('Data de criação do processo:', data_criacao)
113        memInfo = p.memory_info()
114        mem_usada = round(memInfo.rss/1024, 2)
115        print('Memória usada:', mem_usada, 'KB\n')
116
117    # Mostra o ip da interface de rede local:
118    def mostra_info_rede():
119
120        print('CARACTERÍSTICAS:\n')
121        interfaces = psutil.net_if_addrs()
122        nomes = []
123        # Obtém os nomes das interfaces primeiro
124        for i in interfaces:
125            nomes.append(str(i))
126            # print(nomes)
127        if nomes:
128            titulo = '{:12}'.format("INTERFACES")
129            titulo = titulo + '{:>13}'.format("FAMILY")
130            titulo = titulo + '{:>30}'.format("ADDRESS")
131            titulo = titulo + '{:>38}'.format("NETMASK")

```

## Continuação...

```

MonitorSistema_v0.7.py x
132     titulo = titulo + '{:>35}'.format("BROADCAST")
133     print(titulo)
134
135     # Depois, imprimir os valores:
136     for i in nomes:
137         print(i + ":")
138         for j in interfaces[i]:
139             print("\t" + '{:^38}'.format(str(j[0])) + '{:<35}'.format(str(j[1]))
140                         + '{:<42}'.format(str(j[2])) + '{:<38}'.format(str(j[3])))
141     print()
142
143     print('STATUS:\n')
144     nameStatus = []
145     status = psutil.net_if_stats()
146     for i in status:
147         nameStatus.append(str(i))
148         # print(nameStatus)
149
150     if nameStatus:
151         titulo = '{:>15}'.format("ISUP")
152         titulo = titulo + '{:>20}'.format("DUPLEX")
153         titulo = titulo + '{:>25}'.format("SPEED")
154         titulo = titulo + '{:>10}'.format("MTU")
155         print(titulo)
156
157     for i in nameStatus:
158         print(i + ":")
159         print("\t" + '{:^18}'.format(str(status[i][0])) + '{:<35}'.format(str(status[i][1]))
160                         + '{:<10}'.format(str(status[i][2])) + '{:<10}'.format(str(status[i][3])))
161     print()
162
163     print('ENTRADA E SAIDA DE DADOS:\n')
164     io_status = psutil.net_io_counters(bernie=True)

```

```

MonitorSistema_v0.7.py x
165     names_Io_status = []
166     for i in io_status:
167         names_Io_status.append(str(i))
168
169     if names_Io_status:
170         titulo = '{:>20}'.format("BYTES_SENT")
171         titulo = titulo + '{:>16}'.format("BYTES_RECV")
172         titulo = titulo + '{:>15}'.format("PACKETS_SENT")
173         titulo = titulo + '{:>15}'.format("PACKETS_RECV")
174         print(titulo)
175
176     for j in names_Io_status:
177         print(j + ":")
178         print("\t" + '{:^22}'.format(str(io_status[j][0])) + '{:<10}'.format(str(io_status[j][1]))
179                         + '{:<16}'.format(str(io_status[j][2])) + '{:<18}'.format(str(io_status[j][3])))
180     print()
181
182     print('CONEXÕES:\n')
183     connections = psutil.net_connections()
184     list_con = []
185     for i in connections:
186         list_con.append(i)
187         # print(list_con)
188
189     if list_con:
190         titulo = '{:>4}'.format("FD")
191         titulo = titulo + '{:>16}'.format("FAMILY")
192         titulo = titulo + '{:>24}'.format("TYPE")
193         # titulo = titulo + '{:>20}'.format("LADDR")
194         # titulo = titulo + '{:>15}'.format("RADDR")
195         titulo = titulo + '{:>20}'.format("STATUS")
196         titulo = titulo + '{:>11}'.format("PID")
197         print(titulo)

```

## Continuação...

```

198
199     for j in list_con:
200         print('{:^7}'.format(str(j[0])) + '{:<25}'.format(str(j[1])) + '{:<25}'.format(str(j[2]))
201             + '{:<15}'.format(str(j[5])) + '{:<10}'.format(str(j[6])))
202
203
204 list_pids = psutil.pids()
205 #print('\n'+'Lista de processos em execução: ', list_pids)
206
207 for i in list_pids:
208     pid_selecionado = i
209
210 if pid_selecionado in list_pids:
211
212     mostra_info_cpu()
213     mostra_uso_cpu(listCpuPercent)
214     mostra_uso_memoria()
215     mostra_uso_disco()
216     mostra_info_dir()
217     mostra_info_processo()
218     mostra_info_rede()
219
220 else:
221     print('PID selecionado não existe!')

```

### 3.7.3. Execuções

#### Requisito III

##### 3.7.3.1 Primeira

```

infnet [~/desenvolvedor/PycharmProjects/infnet] - .../Projeto_Bloco/TP7/MonitorSistema_v0.7.py [infnet] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
Infnet > Projeto_Bloco > TP7 > MonitorSistema_v0.7.py
Run MonitorSistema_v0.7
INFORMAÇÕES DA CPU:
Nome: Intel(R) Core(TM) i5 CPU M 480 @ 2.67GHz
Arquitetura: X86_64
Palavra (bits): 64
Frequência (MHz): 1199.0
Núcleos (físicos): 4(4)

INFORMAÇÕES POR NÚCLEOS DA CPU:
Núcleo: 6.1 %
Núcleo: 20.0 %
Núcleo: 9.0 %
Núcleo: 4.1 %

INFORMAÇÕES MEMÓRIA PRINCIPAL:
Capacidade total: 3.72 GB
Capacidade em uso: 1.95 GB
Capacidade livre: 0.12 GB
Percentual em uso: 60.6 %

INFORMAÇÕES DISCO RÍGIDO:
Capacidade total: 252.3 GB
Capacidade em uso: 13.01 GB
Capacidade livre: 226.45 GB
Percentual em uso: 5.4 %

ARQUIVOS E DIRETÓRIOS:
Dispositivo: /dev/sda5

```

## Continuação...

```

MonitorSistema_v0.7
Sistema de Arquivo: ext4
TAMANHO DATA DE CRIAÇÃO DATA DE MODIFICAÇÃO NOME
1.74KB Wed Dec 31 21:29:41 1969 Sat Feb 24 23:30:42 2018 tp7.py
7.57KB Wed Dec 31 23:09:07 1969 Sun Mar 11 20:58:08 2018 MonitorSistema_v0.7.py
3.09KB Wed Dec 31 21:52:42 1969 Sun Mar 11 20:01:33 2018 info_rede.py

PROCESSOS:
Id do processo: 10661
Nome do processo: python
Nome do usuário proprietário: alex
Data de criação do processo: 2018-03-11 20:58:08
Memória usada: 13844.0 KB

CARACTERÍSTICAS:
INTERFACES FAMILY ADDRESS NETMASK BROADCAST
lo: AddressFamily.AF_INET 127.0.0.1 255.0.0.0 None
     AddressFamily.AF_INET6 ::1 fffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff None
     AddressFamily.AF_PACKET 00:00:00:00:00:00 None
wlp2s0: AddressFamily.AF_INET 192.168.3.125 255.255.255.0 192.168.3.255
     AddressFamily.AF_INET6 fe80::b38a:bd5c:4eca:2a67%wlp2s0 ffff:ffff:ffff:ffff:ffff:ffff:ffff:ff None
     AddressFamily.AF_PACKET b4:74:9f:92:7c:7f None ff:ff:ff:ff:ff:ff
enp3s0: AddressFamily.AF_PACKET 00:e0:91:49:44:d7 None ff:ff:ff:ff:ff:ff

```

```

MonitorSistema_v0.7
STATUS:
↑ ISUP DUPLEX SPEED MTU
↓
wlp2s0: True NicDuplex.NIC_DUPLEX_UNKNOWN 0 1500
lo: True NicDuplex.NIC_DUPLEX_UNKNOWN 0 65536
enp3s0: True NicDuplex.NIC_DUPLEX_HALF 10 1500

ENTRADA E SAIDA DE DADOS:
↑ BYTES_SENT BYTES_RECV PACKETS_SENT PACKETS_RECV
↓
wlp2s0: 1681219 8246717 11277 10326
lo: 816525 816525 3407 3407
enp3s0: 0 0 0 0

CONEXÕES:
FD FAMILY TYPE STATUS PID
-1 AddressFamily.AF_INET SocketKind.SOCK_DGRAM NONE None
129 AddressFamily.AF_INET SocketKind.SOCK_STREAM ESTABLISHED 2668
-1 AddressFamily.AF_INET6 SocketKind.SOCK_STREAM LISTEN None
-1 AddressFamily.AF_INET SocketKind.SOCK_STREAM LISTEN None
-1 AddressFamily.AF_INET SocketKind.SOCK_DGRAM NONE None
-1 AddressFamily.AF_INET SocketKind.SOCK_DGRAM NONE None
96 AddressFamily.AF_INET6 SocketKind.SOCK_DGRAM NONE 2668
220 AddressFamily.AF_INET SocketKind.SOCK_STREAM LISTEN 3815
-1 AddressFamily.AF_INET SocketKind.SOCK_STREAM LISTEN None
86 AddressFamily.AF_INET SocketKind.SOCK_DGRAM NONE 2668
119 AddressFamily.AF_INET SocketKind.SOCK_STREAM LISTEN 3815

```

### 3.7.3.2. Segunda

```
infnet [~/desenvolvedor/PycharmProjects/infnet] - .../Projeto_Bloco/TP7/MonitorSistema_v0.7.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help
infnet > Projeto_Bloco > TP7 > MonitorSistema_v0.7.py
Run MonitorSistema_v0.7

INFORMAÇÕES DA CPU:
Nome: Intel(R) Core(TM) i5 CPU      M 480 @ 2.67GHz
Arquitetura: X86_64
Palavra (bits): 64
Frequência (MHz): 1199.0
Núcleos (físicos): 4(4)

INFORMAÇÕES POR NÚCLEOS DA CPU:
Núcleo: 8.1 %
Núcleo: 24.5 %
Núcleo: 21.0 %
Núcleo: 10.9 %

INFORMAÇÕES MEMÓRIA PRINCIPAL:
Capacidade total: 3.72 GB
Capacidade em uso: 1.96 GB
Capacidade livre: 0.12 GB
Percentual em uso: 60.8 %

INFORMAÇÕES DISCO RÍGIDO:
Capacidade total: 252.3 GB
Capacidade em uso: 13.01 GB
Capacidade livre: 226.45 GB
Percentual em uso: 5.4 %

ARQUIVOS E DIRETÓRIOS:
Dispositivo: /dev/sda5
Sistema de Arquivo: ext4
```

```
MonitorSistema_v0.7

Dispositivo: /dev/sda5
Sistema de Arquivo: ext4

TAMANHO DATA DE CRIAÇÃO DATA DE MODIFICAÇÃO NOME
1.74KB Wed Dec 31 21:29:41 1969 Sat Feb 24 23:30:42 2018 tp7.py
7.50KB Wed Dec 31 23:07:57 1969 Sun Mar 11 21:09:42 2018 MonitorSistema_v0.7.py
3.09KB Wed Dec 31 21:52:42 1969 Sun Mar 11 20:01:33 2018 info_rede.py

PROCESSOS:
Id do processo: 10906
Nome do processo: python
Nome do usuário proprietário: alex
Data de criação do processo: 2018-03-11 21:09:42
Memória usada: 13780.0 KB

CARACTERÍSTICAS:

INTERFACES FAMILY ADDRESS NETMASK BROADCAST
lo: AddressFamily.AF_INET 127.0.0.1 255.0.0.0 None
AddressFamily.AF_INET6 ::1 fffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff None
AddressFamily.AF_PACKET 00:00:00:00:00:00 None
wlp2s0: AddressFamily.AF_INET 192.168.3.125 255.255.255.0 192.168.3.255
AddressFamily.AF_INET6 fe80::b38a:bd5c:4eca:2a67%wlp2s0 fffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff None
AddressFamily.AF_PACKET b4:74:9f:92:7c:7f None ff:ff:ff:ff:ff:ff
enp3s0: AddressFamily.AF_PACKET 00:e0:91:49:44:d7 None ff:ff:ff:ff:ff:ff
```

```

MonitorSistema_v0.7
STATUS:
wlp2s0: ISUP DUPLEX SPEED MTU
True NicDuplex.NIC_DUPLEX_UNKNOWN 0 1500
lo: True NicDuplex.NIC_DUPLEX_UNKNOWN 0 65536
enp3s0: True NicDuplex.NIC_DUPLEX_HALF 10 1500

ENTRADA E SAIDA DE DADOS:
wlp2s0: BYTES_SENT BYTES_RECV PACKETS_SENT PACKETS_RECV
1706487 8261881 11474 10443
lo: 845391 845391 3526 3526
enp3s0: 0 0 0 0

CONEXÕES:
FD FAMILY TYPE STATUS PID
-1 AddressFamily.AF_INET SocketKind.SOCK_STREAM LISTEN None
-1 AddressFamily.AF_INET SocketKind.SOCK_STREAM LISTEN None
220 AddressFamily.AF_INET SocketKind.SOCK_STREAM LISTEN 3815
-1 AddressFamily.AF_INET6 SocketKind.SOCK_STREAM LISTEN None
-1 AddressFamily.AF_INET SocketKind.SOCK_DGRAM NONE None
-1 AddressFamily.AF_INET SocketKind.SOCK_DGRAM NONE None
-1 AddressFamily.AF_INET6 SocketKind.SOCK_STREAM LISTEN None
119 AddressFamily.AF_INET SocketKind.SOCK_STREAM LISTEN 3815
-1 AddressFamily.AF_INET SocketKind.SOCK_DGRAM NONE None
-1 AddressFamily.AF_INET6 SocketKind.SOCK_DGRAM NONE None
86 AddressFamily.AF_INET SocketKind.SOCK_DGRAM NONE 2668

```

### 3.7.3.3. Terceira

```

infnet [~/desenvolvedor/PycharmProjects/infnet] - .../Projeto_Bloco/TP7/MonitorSistema_v0.7.py [ir]
File Edit View Navigate Code Refactor Run Tools VCS Window Help
infnet > Projeto_Bloco > TP7 > MonitorSistema_v0.7.py >
Run MonitorSistema_v0.7
1:Project Z-Structure Favorites
INFORMAÇÕES DA CPU:
Nome: Intel(R) Core(TM) i5 CPU M 480 @ 2.67GHz
Arquitetura: X86_64
Palavra (bits): 64
Frequência (MHz): 1199.0
Núcleos (físicos): 4(4)

INFORMAÇÕES POR NÚCLEOS DA CPU:
Núcleo: 0.0 %
Núcleo: 1.0 %
Núcleo: 0.0 %
Núcleo: 1.0 %

INFORMAÇÕES MEMÓRIA PRINCIPAL:
Capacidade total: 3.72 GB
Capacidade em uso: 2.0 GB
Capacidade livre: 0.15 GB
Percentual em uso: 61.7 %

INFORMAÇÕES DISCO RÍGIDO:
Capacidade total: 252.3 GB
Capacidade em uso: 13.02 GB
Capacidade livre: 226.45 GB
Percentual em uso: 5.4 %

ARQUIVOS E DIRETÓRIOS:
Dispositivo: /dev/sda5
Sistema de Arquivo: ext4

```

MonitorSistema_v0.7				
Dispositivo: /dev/sda5 Sistema de Arquivo: ext4				
TAMANHO	DATA DE CRIAÇÃO	DATA DE MODIFICAÇÃO	NOME	
1.74KB	Wed Dec 31 21:29:41 1969	Sat Feb 24 23:30:42 2018	tp7.py	
7.50KB	Wed Dec 31 23:07:57 1969	Sun Mar 11 21:09:42 2018	MonitorSistema_v0.7.py	
3.09KB	Wed Dec 31 21:52:42 1969	Sun Mar 11 20:01:33 2018	info_rede.py	
<b>PROCESSOS:</b>				
Id do processo: 11316 Nome do processo: python Nome do usuário proprietário: alex Data de criação do processo: 2018-03-11 21:23:14 Memória usada: 13856.0 KB				
<b>CARACTERÍSTICAS:</b>				
INTERFACES	FAMILY	ADDRESS	NETMASK	BROADCAST
lo:	AddressFamily.AF_INET	127.0.0.1	255.0.0.0	None
	AddressFamily.AF_INET6	::1	ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff	None
	AddressFamily.AF_PACKET	00:00:00:00:00:00	None	None
wlp2s0:	AddressFamily.AF_INET	192.168.3.125	255.255.255.0	192.168.3.255
	AddressFamily.AF_INET6	fe80::b38a:bd5c:4eca:2a67%wlp2s0	ffff:ffff:ffff:ffff:ffff:ffff::	None
	AddressFamily.AF_PACKET	b4:74:f9:92:7c:7f	None	ff:ff:ff:ff:ff:ff
enp3s0:	AddressFamily.AF_PACKET	00:e0:91:49:44:d7	None	ff:ff:ff:ff:ff:ff

MonitorSistema_v0.7					
STATUS:					
	ISUP	DUPLEX	SPEED	MTU	
wlp2s0:	True	NicDuplex.NIC_DUPLEX_UNKNOWN	0	1500	
lo:	True	NicDuplex.NIC_DUPLEX_UNKNOWN	0	65536	
enp3s0:	True	NicDuplex.NIC_DUPLEX_HALF	10	1500	
ENTRADA E SAIDA DE DADOS:					
	BYTES_SENT	BYTES_RECV	PACKETS_SENT	PACKETS_RECV	
wlp2s0:	1764988	8296933	11806	10701	
lo:	875336	875336	3642	3642	
enp3s0:	0	0	0	0	
CONEXÕES:					
FD	FAMILY	TYPE	STATUS	PID	
-1	AddressFamily.AF_INET6	SocketKind.SOCK_DGRAM	NONE	None	
-1	AddressFamily.AF_INET	SocketKind.SOCK_STREAM	TIME_WAIT	None	
-1	AddressFamily.AF_INET	SocketKind.SOCK_STREAM	LISTEN	None	
-1	AddressFamily.AF_INET	SocketKind.SOCK_STREAM	LISTEN	None	
119	AddressFamily.AF_INET	SocketKind.SOCK_STREAM	LISTEN	3815	
-1	AddressFamily.AF_INET6	SocketKind.SOCK_STREAM	LISTEN	None	
-1	AddressFamily.AF_INET	SocketKind.SOCK_DGRAM	NONE	None	
86	AddressFamily.AF_INET	SocketKind.SOCK_DGRAM	NONE	2668	
-1	AddressFamily.AF_INET	SocketKind.SOCK_DGRAM	NONE	None	
-1	AddressFamily.AF_INET	SocketKind.SOCK_DGRAM	NONE	None	
220	AddressFamily.AF_INET	SocketKind.SOCK_STREAM	LISTEN	3815	

### 3.8. Teste de Performance - TP8

O entregável do seu projeto de bloco será:

Um aplicativo simples de apresentação textual do monitoramento e análise de computadores em rede. Ele deverá ser implementado em Python usando módulos como psutil (para capturar dados do sistema computacional) e sockets (para criar cliente e servidor) e será desenvolvido de forma incremental durante o curso. A apresentação gráfica no lado cliente é opcional e pode ser parcial (parte gráfica e parte texto).

Certifique-se apenas que está realizando o básico do que este TP requisita.

A seguir, os requisitos básicos:

1. Implementar ao menos 2 tipos de obtenção de informação de um computador, conforme feito nos TPs anteriores, mas agora no servidor. Você pode implementar todas que foram requisitadas nos TPs anteriores, mas não é necessário no momento. No entanto, será necessário na etapa 09.

Alguns exemplos de informações:

- a. uso de processamento
- b. uso/capacidade de memória
- c. sobre arquivos e diretórios
- d. sobre processos executando
- e. sobre redes/interfaces de redes

2. Implemente um programa cliente que requisite tais informações ao programa servidor e exiba os dados usando texto formatado ou de forma visual (com PyGame, por exemplo).

3. Implemente um programa servidor que receba tais requisições do cliente, as obtenha na própria máquina e as envie ao cliente.

4. Escrever um pequeno relatório com o código (e explicações que achar necessárias sobre ele), ao menos 3 exemplos de execução e justificativa das escolhas das informações a serem exibidas.

### 3.8.1. Código Socket Cliente

#### Requisitos I, II e III

---

```

import socket, time, pickle, os
GBYTES = (1024 * 1024 * 1024)
# Mostra a informações da CPU:
def print_info_cpu(list):
    dic = {}
    for j in list[1]:
        # print(str(j))
        brand = list[1]['brand']
        arch = list[1]['arch']
        bits = list[1]['bits']
        freq = list[1]['hz_actual']
        count = list[1]['count']
        dic['Nome'] = brand
        dic['Arquitetura'] = arch
        dic['Palavra'] = bits
        dic['Frequencia'] = freq
        dic['Nucleos'] = count
    print()
    print('ARQUITETURA DA CPU:\n')
    print('Nome:', dic['Nome'], '\n' 'Arquitetura:', dic['Arquitetura'],
          '\n' 'Palavra (bits):', dic['Palavra'], '\n' 'Frequência: ',
          dic['Frequencia'],
          '\n' 'Nucleos: ', dic['Nucleos'], ' (' + str(list[2]) + ')')
    print()
    print('PERCENTUAL DE USO DA CPU POR NÚCLEO:\n')
    for i, elem in enumerate(list[0]):
        # print(str(i))
        if len(list[0]) == 1:
            print('Core:' + str(i))
            load("|", "|", 35, elem)
        else:
            print('Core:' + str(i))
            load("|", "|", 35, elem)
    print()
    # Mostra a capacidade de MP em uso e total:
def print_info_memory(list):
    for mem in list:
        # print(i)

```

```

total = round(mem.total/GBYTES, 2)
totalEmUso = round(mem.used/GBYTES, 2)
totalLivre = round(mem.free/GBYTES, 2)
memPercent = round(mem.percent, 2)
print('INFORMAÇÕES MEMÓRIA PRINCIPAL:' '\n')
print('Capacidade total: ' + str(total) + ' GB '\n')
print('Capacidade em uso: ' + str(totalEmUso) + ' GB')
load("|", "|", 35, memPercent)
print('\n' 'Capacidade livre: ' + str(totalLivre) + ' GB' '\n')

# Mostra a capacidade do disco rígido em uso e total:
def print_info_disk(list):
    for disco in list:
        #print(disco)
        total = round(disco.total/GBYTES, 2)
        discoUso = round(disco.used/GBYTES, 2)
        discoLivre = round(disco.free/GBYTES, 2)
        discoPercent = round(disco.percent, 2)
        print('INFORMAÇÕES DISCO RÍGIDO:' '\n')
        print('Capacidade total: ' + str(total) + ' GB '\n')
        print('Capacidade em uso: ' + str(discoUso) + ' GB')
        load("|", "|", 35, discoPercent)
        print('\n' 'Capacidade disponível: ' + str(discoLivre) + ' GB\n')

# Mostra informações de diretórios, arquivos e sistema de arquivos
def print_info_dir(list):
    print('ARQUIVOS E DIRETÓRIOS:' '\n')
    list_partition = []
    for i in list[0]:
        list_partition.append(i)
    # print(list_partition)
    if list_partition:
        titulo_partition = '{:^15}'.format("DEVICE")
        titulo_partition = titulo_partition + '{:^20}'.format("POINT MOUNT")
        titulo_partition = titulo_partition + '{:^35}'.format("FILE SYSTEM")
        print(titulo_partition)
        for j in list_partition:
            # print(j)
            device = j[0]
            mount = j[1]
            sys_file = j[2]
            print("\t" + '{:<15}'.format(device) + '{:<28}'.format(mount)
                  + '{:<10}'.format(sys_file))
    print()
    if list[1]:
        titulo = '{:^13}'.format('TAMANHO')

```

```

    titulo = titulo + '{:^25}'.format('DATA DE CRIAÇÃO')
    titulo = titulo + '{:^33}'.format('DATA DE MODIFICAÇÃO')
    titulo = titulo + '{:^27}'.format('NOME')
    print(titulo)
for key in list[1]:
    #print(list[1][key])
    file_name = '{:<30}'.format(key)
    for i in list[1][key]:
        kb = list[1][key][0]/1024
        tam = '{:>10}'.format(str('{:.2f}'.format(kb) + ' KB'))
        c_time = '{:^32}'.format(time.ctime((list[1][key][1])))
        m_time = '{:<30}'.format(time.ctime((list[1][key][2])))
        print(str(tam + c_time + m_time + file_name))
    print()
# Mostra informações de processos em execução
def print_info_process(list):
    print('PROCESSOS:\n')
    percentual = 2
    if list:
        titulo = '\t''{:<6}'.format("PID")
        titulo = titulo + '{:^20}'.format("Name")
        titulo = titulo + '{:^6}'.format("%CPU")
    print(titulo)
    lista_ordenada = sorted(list, key=func_comparation, reverse=True)
    for i in lista_ordenada:
        if i['cpu_percent'] >= (percentual):
            texto = '\t''{:<6}'.format(i['pid'])
            texto = texto + '{:^20}'.format(i['name'])
            texto = texto + '{:>6}'.format(i['cpu_percent'])
            print(texto)
            #time.sleep(0.2)
    print()
# Mostra informações de rede
def print_info_net(list, list_st):
    #print(list)
    print('INTERFACES DE REDE:\n')
    nomes = []
    # Obtém os nomes das interfaces
    for key in list[0]:
        nomes.append(str(key))
    if nomes:
        titulo = '{:13}'.format("INTERFACES")
        titulo = titulo + '{:>13}'.format("FAMILY")
        titulo = titulo + '{:>20}'.format("ADDRESS")

```

```

    titulo = titulo + '{:>20}'.format("NETMASK")
    titulo = titulo + '{:>20}'.format("BROADCAST")
    print(titulo)
for i in nomes:
    print(i + ':')
    values_tuple = []
    for a, b, c, d, _, in list[0][i]:
        #print(a)
        values_tuple.append(a)
        values_tuple.append(b)
        values_tuple.append(c)
        values_tuple.append(d)
        #print(list_fam)
    for j in values_tuple:
        family = str(values_tuple[0])
        address = str(values_tuple[1])
        netmask = str(values_tuple[2])
        broadcast = str(values_tuple[3])
        print('\t' + '{:^29}'.format(family) + '{:<20}'.format(address)
              + '{:<17}'.format(netmask) + '{:<20}'.format(broadcast))
    print()
    input('Pressione qualquer tecla para continuar...')

print()
print('STATUS:\n')
nameStatus = []
# Obtém os nomes das interfaces
for k in list_st[0]:
    nameStatus.append(str(k))
if nameStatus:
    titulo = '{:>18}'.format("ISUP")
    titulo = titulo + '{:>20}'.format("DUPLEX")
    titulo = titulo + '{:>20}'.format("SPEED")
    titulo = titulo + '{:>8}'.format("MTU")
    print(titulo)
for l in nameStatus:
    print(l + ':')
    isup = str(list_st[0][l][0])
    duplex = str(list_st[0][l][1])
    speed = str(list_st[0][l][2])
    mtu = str(list_st[0][l][3])
    print("\t" + '{:^14}'.format(isup) + '{:<33}'.format(duplex)
          + '{:<7}'.format(speed) + '{:<8}'.format(mtu))
print()
input('Pressione qualquer tecla para continuar...')

```

```

print()
print('ENTRADA E SAIDA DE DADOS:\n')
names_Io_status = []
for m in list_st[1]:
    names_Io_status.append(str(m))
if names_Io_status:
    titulo = '{:>20}'.format("BYTES_SENT")
    titulo = titulo + '{:>14}'.format("BYTES_RECV")
    titulo = titulo + '{:>15}'.format("PACKETS_SENT")
    titulo = titulo + '{:>15}'.format("PACKETS_RECV")
    print(titulo)
for n in names_Io_status:
    print(n + ':')
    bytes_sent = str(list_st[1][n][0])
    bytes_recv = str(list_st[1][n][1])
    packets_sent = str(list_st[1][n][2])
    packets_recv = str(list_st[1][n][3])
    print("\t" + '{:^4}'.format(' ') + '{:<12}'.format(bytes_sent) +
          '{:<15}'.format(bytes_recv) + '{:<15}'.format(packets_sent) +
          '{:<10}'.format(packets_recv))
print()
input('Pressione qualquer tecla para continuar...')

print()
print('CONEXÕES:\n')
list_con = []
for i in list_st[2]:
    list_con.append(i)
# print(list_con)
if list_con:
    titulo = '{:>4}'.format("FD")
    titulo = titulo + '{:>16}'.format("FAMILY")
    titulo = titulo + '{:>24}'.format("TYPE")
    # titulo = titulo + '{:>20}'.format("LADDR")
    # titulo = titulo + '{:>15}'.format("RADDR")
    titulo = titulo + '{:>20}'.format("STATUS")
    titulo = titulo + '{:>11}'.format("PID")
    print(titulo)
for j in list_con:
    fd = str(j[0])
    family = str(j[1])
    type = str(j[2])
    status = str(j[5])
    pid = str(j[6])
    print('{:^7}'.format(fd) + '{:<25}'.format(family) + '{:<25}'.format(type))

```

```

        + '{:<15}'.format(status) + '{:<10}'.format(pid))
print()

def load(left_side, right_side, length, percent):
    x = 0
    y = ""
    p = length*percent/100
    #print("\r")
    while x <= p:
        space = length - len(y)
        space = " " * space
        z = left_side + y + space + right_side
        y += "#"
        time.sleep(0.1)
        x += 1
    print("\r", 'Percentual:', str(percent) + '%', z)

def clear():
    os.system("cls" if os.name == "nt" else "clear")

def func_comparation(l):
    return(l['cpu_percent'])

def print_menu(): # Função que exibe menu de exibição
    print(80 * '*')
    print('*', '{:^76}'.format(''), '*' * '\n***', '{:^76}'.format('MONITOR DO
SISTEMA')),
    '*' * '\n***', '{:^76}'.format(''), '*')
    print(80 * '*')
    print(37 * "**", "MENU", 37 * "**")
    print("1. CPU")
    print("2. MEMÓRIA")
    print("3. DISCO")
    print("4. DIRETÓRIOS")
    print("5. PROCESSOS")
    print("6. REDES")
    print("7. SAIR")
    print(80 * "**")

serverHost = socket.gethostname()
serverPort = 30000
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
try:
    # Conecta-se ao servidor

```

```

client.connect((serverHost, serverPort))
data = ''
loop = True
while loop: # Enquanto loop = True continuará até loop = False
    print_menu() # Menu de exibição
    choice = int(input("Digite uma opção [1-7]: "))
    print()
    if choice == 1:
        clear()
        sair = True
        while sair:
            data = '1'
            # Envia mensagem vazia apenas para indicar a requisição
            client.send(data.encode('utf-8'))
            # Recebe reposta da requisição
            bytes = client.recv(2048)
            # Converte os bytes para lista
            list = pickle.loads(bytes)
            print_info_cpu(list)
            #time.sleep(1)
            voltaMenu = int(input('Digite zero[0] para voltar ao Menu: '))
            if voltaMenu == 0:
                sair = False
    if choice == 2:
        clear()
        sair = True
        while sair:
            data = '2'
            # Envia mensagem vazia apenas para indicar a requisição
            client.send(data.encode('utf-8'))
            # Recebe reposta da requisição
            bytes = client.recv(1024)
            # Converte os bytes para lista
            list = pickle.loads(bytes)
            print_info_memory(list)
            # time.sleep(1)
            voltaMenu = int(input('Digite zero[0] para voltar ao Menu: '))
            if voltaMenu == 0:
                sair = False
    if choice == 3:
        clear()
        sair = True
        while sair:
            data = '3'

```

```

# Envia mensagem vazia apenas para indicar a requisição
client.send(data.encode('utf-8'))
# Recebe reposta da requisição
bytes = client.recv(1024)
# Converte os bytes para lista
list = pickle.loads(bytes)
print_info_disk(list)
# time.sleep(1)
voltaMenu = int(input('Digite zero[0] para voltar ao Menu: '))
if voltaMenu == 0:
    sair = False
if choice == 4:
    clear()
    sair = True
while sair:
    data = '4'
    # Envia mensagem vazia apenas para indicar a requisição
    client.send(data.encode('utf-8'))
    # Recebe reposta da requisição
    bytes = client.recv(2048)
    # Converte os bytes para listamostraInfoDir(list)
    list = pickle.loads(bytes)
    print_info_dir(list)
    # time.sleep(1)
    voltaMenu = int(input('Digite zero[0] para voltar ao Menu: '))
    if voltaMenu == 0:
        sair = False
if choice == 5:
    clear()
    sair = True
while sair:
    data = '5'
    # Envia mensagem vazia apenas para indicar a requisição
    client.send(data.encode('utf-8'))
    # Recebe reposta da requisição
    bytes = client.recv(50000)
    # Converte os bytes para lista
    list = pickle.loads(bytes)
    print_info_process(list)
    # time.sleep(1)
    voltaMenu = int(input('Digite zero[0] para voltar ao Menu: '))
    if voltaMenu == 0:
        sair = False
if choice == 6:

```

```

        clear()
        sair = True
        while sair:
            data = '6'
            # Envia mensagem vazia apenas para indicar a requisição
            client.send(data.encode('utf-8'))
            # Recebe resposta da requisição
            bytes1 = client.recv(50000)
            # Converte os bytes para lista
            list1 = pickle.loads(bytes1)
            # Recebe resposta da requisição
            bytes2 = client.recv(50000)
            # Converte os bytes para lista
            list2 = pickle.loads(bytes2)
            print_info_net(list1, list2)
            voltaMenu = int(input('Digite zero[0] para voltar ao Menu: '))
            if voltaMenu == 0:
                sair = False
            if choice == 7:
                data = '7'
                client.send(data.encode('utf-8'))
                print("Saindo do programa...")
                loop = False # Isso fará com que o loop do while termine
            else:
                # Qualquer entrada inteira diferente dos valores 1-7, imprimirá
                # uma mensagem de erro
                print("Opção inválida!. Tente novamente ...")

        except Exception as error:
            print(str(error))
        # Fecha o socket
        client.close()
        input('Pressione qualquer tecla para sair...')


```

---

### 3.8.3. Código Socket Servidor

```

import socket, os, psutil, cpuinfo, pickle
# Obtem nome e porta da máquina
myHost = socket.gethostname()
myPort = 30000

```

```

# Cria o socket
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
# Associa ip e porta
server.bind((myHost, myPort))
# Servidor esperando por uma conexão
server.listen()
print('Servidor:', myHost, 'esperando conexão na porta:', myPort)
# Aceita alguma conexão
(conexao, address) = server.accept()
print('Conectado a:', str(address))
while True:
    # Recebe requisição do cliente
    data = ''
    data = conexao.recv(2048)
    if data.decode('utf-8') == '1':
        request = data.decode('utf-8')
        print('Requisição da opção', request, 'recebida com sucesso!')
        print('Processando...')

        list_percent_cpu=psutil.cpu_percent(interval=1,percpu=True)
        dict_info_cpu = cpuinfo.get_cpu_info()
        num_cores = psutil.cpu_count(logical=True)
        # Cria a lista de resposta
        response = []
        response.append(list_percent_cpu)
        response.append(dict_info_cpu)
        response.append(num_cores)
        # Prepara a lista para o envio
        bytes_resp = pickle.dumps(response)
        # Envia os dados
        conexao.send(bytes_resp)
        print('Resposta da opção', request, 'enviada com sucesso!\n')
    if data.decode('utf-8') == '2':
        request = data.decode('utf-8')
        print('Requisição da opção', request, 'recebida com sucesso!')
        print('Processando...')
        info_mem = psutil.virtual_memory()
        # Cria a lista de resposta
        response = []
        response.append(info_mem)

```

```

# Prepara a lista para o envio
bytes_resp = pickle.dumps(response)
# Envia os dados
conexao.send(bytes_resp)
print('Resposta da opção', request, 'enviada com sucesso!\n')
if data.decode('utf-8') == '3':
    request = data.decode('utf-8')
    print('Requisição da opção', request, 'recebida com sucesso!')
    print('Processando...')
    info_disco = psutil.disk_usage('.')
# Cria a lista de resposta
response = []
response.append(info_disco)
# Prepara a lista para o envio
bytes_resp = pickle.dumps(response)
# Envia os dados
conexao.send(bytes_resp)
print('Resposta da opção', request, 'enviada com sucesso!\n')
if data.decode('utf-8') == '4':
    request = data.decode('utf-8')
    print('Requisição da opção', request, 'recebida com sucesso!')
    print('Processando...')
    list = os.listdir()
    dict_file = {}
    for file in list:
        if os.path.isfile(file):
            dict_file[file] = []
            dict_file[file].append(os.stat(file).st_size)
            dict_file[file].append(os.stat(file).st_ctime)
            dict_file[file].append(os.stat(file).st_mtime)
    info_partitions = psutil.disk_partitions()
# Cria a lista de resposta
response = []
response.append(info_partitions)
response.append(dict_file)
# Prepara a lista para o envio
bytes_resp = pickle.dumps(response)
# Envia os dados
conexao.send(bytes_resp)

```

```

print('Resposta da opção', request, 'enviada com sucesso!\n')
if data.decode('utf-8') == '5':
    request = data.decode('utf-8')
    print('Requisição da opção', request, 'recebida com sucesso!')
    print('Processando...')
    for i in range(2):
        response = []
        list_proc = psutil.process_iter()
        for proc in list_proc:
            try:
                pinfo=proc.as_dict(attrs=['pid','name','cpu_percent'])
                response.append(pinfo)
            # Prepara a lista para o envio
        except psutil.NoSuchProcess:
            pass
        bytes_resp = pickle.dumps(response)
        # Envia os dados
        conexao.send(bytes_resp)
        print('Resposta da opção', request, 'enviada com sucesso!\n')
if data.decode('utf-8') == '6':
    request = data.decode('utf-8')
    print('Requisição da opção', request, 'recebida com sucesso!')
    print('Processando...')
    dict_interface = psutil.net_if_addrs()
    # Cria a lista de resposta
    response = []
    response.append(dict_interface)
    # Prepara a lista para o envio
    bytes_respl = pickle.dumps(response)
    # Envia os dados
    conexao.send(bytes_respl)
    status = psutil.net_if_stats()
    io_status = psutil.net_io_counters(pernic=True)
    conn = psutil.net_connections()
    # Cria a lista de resposta
    response_2 = []
    response_2.append(status)
    response_2.append(io_status)
    response_2.append(conn)

```

```

# Prepara a lista para o envio
bytes_resp2 = pickle.dumps(response_2)
# Envia os dados
conexao.send(bytes_resp2)
print('Resposta da opção', request, 'enviada com sucesso!\n')
if data.decode('utf-8') == '7':
    print('Conexão encerrada.')
    break
# Fecha socket do servidor e cliente
conexao.close()
server.close()

```

### 3.8.4. Relatório

## Requisito IV

O projeto abordado neste TP, baseia-se no modelo cliente-servidor utilizando módulos em python como o os, psutil, time, pickle, socket e afins. Sendo assim, para iniciar a abordagem sobre a aplicação desenvolvida será apresentada algumas funções implementadas para efeito de exemplo.

Primeira função *print\_uso\_cpu()* - o objeto desta função é imprimir informações da arquitetura da CPU e o percentual de processamento usado por cada núcleo. Para isso, ela recebe como argumento uma lista contendo um dicionário e uma lista de percentual da CPU, requisitada pelo cliente e respondida pelo servidor, através de uma lógica de programação que verifica a opção solicitada e executa a função *psutil.cpu\_percent()* e *psutil.get\_cpu\_info()*. Para tornar mais amigável a interação com o usuário foi implementado nas linhas 33 e 36 a função *load()*, que carrega uma barra de progresso do processamento da CPU. Quanto as escolhas sobre as informações a serem exibidas optou-se pelas seguintes: Nome da CPU, arquitetura, palavras, núcleos e percentual da CPU, por considerar as mais relevantes. Segunda função *print\_uso\_memory()* - destaca-se por imprimir as informações da capacidade usada, livre, total e o percentual de uso da MP. Sendo assim, ela recebe como atributo uma lista que retorna as estatísticas sobre o uso da memória do sistema como uma tupla nomeada, incluindo os seguintes campos, expressos em bytes por exemplo, *total*, *used*, *free* e *percent*. Nesta função foi

implementado nas linhas 43-46 a função *round()* que recebe dois argumentos para formatar os campos da tupla em GB e um *float* com duas casas decimais. Conta ainda com a função *load()* já citada a cima. Optou-se pelas informações citadas a cima por considerá-las essencial.

Terceira e ultima função, embora existam outras, mas será abordada para finalizar o exemplos a função *print\_info\_net()* que tem como finalidade imprimir informações sobre as interfaces de rede, status, status de entrada e saída de bytes/pacotes, processos e conexões. Esta função diferentemente das demais recebe dois atributos, primeiro uma lista de dicionário e o segundo uma lista com lista de tuplas. Na linha 149 foi implementado uma estrutura de repetição utilizando um *for* para iterar nos campos de cada tupla e retornar os valores interessados. Nesta função escolheu-se exibir as informações de rede como as famílias, ip, mascara, broadcast, bytes enviados/recebidos, pacotes enviados/recebidos, mtu e conexões ativas.

---

### 3.8.5. Execuções

#### 3.8.5.1. Primeira

**Menu:**

```
*****
*
*          MONITOR DO SISTEMA
*
*****
***** MENU *****
1. CPU
2. MEMÓRIA
3. DISCO
4. DIRETÓRIOS
5. PROCESSOS
6. REDES
7. SAIR
*****
Digite uma opção [1-7]:
```

Opção 1:

```
ARQUITETURA DA CPU:
Nome: Intel(R) Core(TM) i5 CPU      M 480 @ 2.67GHz
Arquitetura: X86_64
Palavra (bits): 64
Frequência: 1.3330 GHz
Núcleos: 4 (4)

PERCENTUAL DE USO DA CPU POR NÚCLEO:

Core:0
Percentual: 6.0% |■
Core:1
Percentual: 6.9% |■
Core:2
Percentual: 3.0% |■
Core:3
Percentual: 99.0% |███████████|■

Digite zero[0] para voltar ao Menu:
```

Opção 2:

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_B
INFORMAÇÕES MEMÓRIA PRINCIPAL:

Capacidade total: 3.72 GB
Capacidade em uso: 1.91 GB
Percentual: 59.9% |███████████|■

Capacidade livre: 0.22 GB
Digite zero[0] para voltar ao Menu: ■
```

Opção 3:

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_B
INFORMAÇÕES DISCO RÍGIDO:

Capacidade total: 931.51 GB
Capacidade em uso: 535.69 GB
Percentual: 57.5% |███████████|■

Capacidade disponível: 395.82 GB
Digite zero[0] para voltar ao Menu: ■
```

## Opção 4:

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/Apresentação

DEVICE      POINT MOUNT          FILE SYSTEM
/dev/sda5    /           ext4
/dev/sdb1   /media/alex/Hitachi  fuseblk

TAMANHO      DATA DE CRIAÇÃO          DATA DE MODIFICAÇÃO        NOME
12021.20 KB  Fri Mar 30 18:33:47 2018  Mon Mar 19 02:16:03 2018  a.pdf
  0.96 KB   Fri Mar 30 12:38:31 2018  Fri Mar 30 12:38:30 2018  barra_progresso_1.py
  7.07 KB   Thu Mar 22 09:24:44 2018  Thu Mar 22 09:23:31 2018  cliente_socket_tcp_v1.0.py
  7.47 KB   Thu Mar 22 15:20:48 2018  Thu Mar 22 15:20:08 2018  cliente_socket_tcp_v1.1.py
  7.69 KB   Sat Mar 24 08:25:57 2018  Sat Mar 24 08:24:37 2018  cliente_socket_tcp_v1.2.py
12.97 KB    Mon Mar 26 09:14:58 2018  Sun Mar 25 13:39:25 2018  cliente_socket_tcp_v1.3.py
  7.69 KB   Sat Mar 24 08:22:28 2018  Fri Mar 23 02:36:42 2018  cliente_socket_tcp_v1.3.py
13.77 KB    Sat Mar 31 18:58:44 2018  Sat Mar 31 18:58:44 2018  cliente_socket_tcp_v1.4.py
15.49 KB    Fri Mar 30 20:54:31 2018  Fri Mar 30 20:52:15 2018  cliente_socket_tcp_v1.4.2.py
13.03 KB    Mon Mar 26 09:14:58 2018  Sun Mar 25 18:17:17 2018  cliente_socket_tcp_v1.4.py
  3.46 KB   Thu Mar 22 09:24:44 2018  Thu Mar 22 08:13:01 2018  servidor_socket_tcp_v1.0.py
  3.08 KB   Thu Mar 22 15:21:28 2018  Thu Mar 22 15:21:09 2018  servidor_socket_tcp_v1.1.py
  3.46 KB   Sat Mar 24 08:25:57 2018  Sat Mar 24 08:24:43 2018  servidor_socket_tcp_v1.2.py
  4.70 KB   Mon Mar 26 09:14:58 2018  Sun Mar 25 13:42:59 2018  servidor_socket_tcp_v1.3.py
  3.76 KB   Sat Mar 24 08:22:20 2018  Fri Mar 23 21:53:51 2018  servidor_socket_tcp_v1.3.py
  5.07 KB   Sat Mar 31 00:38:45 2018  Sat Mar 31 00:38:45 2018  servidor_socket_tcp_v1.4.1.py
  6.15 KB   Fri Mar 30 20:54:42 2018  Fri Mar 30 20:04:14 2018  servidor_socket_tcp_v1.4.2.py
  4.70 KB   Mon Mar 26 09:14:58 2018  Sun Mar 25 13:42:59 2018  servidor_socket_tcp_v1.4.py
1.20 KB     Thu Mar 22 09:24:44 2018  Wed Mar 21 01:42:42 2018  teste.py
  1.94 KB   Fri Mar 23 00:19:06 2018  Fri Mar 23 00:19:06 2018  teste_dir.py
  63.43 KB  Fri Mar 30 20:32:08 2018  Sun Mar 11 23:43:58 2018  x.pdf

Digite zero[0] para voltar ao Menu: ■
```

## Opção 5:

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/Apresentação

PROCESSOS:

  PID      Name      %CPU
  6428     python    102.9
  5616     java      26.0

Digite zero[0] para voltar ao Menu: ■
```

## Opção 6:

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/Apresentação

INTERFACES DE REDE:

INTERFACES      FAMILY      ADDRESS      NETMASK      BROADCAST
lo:            AddressFamily.AF_INET  127.0.0.1    255.0.0.0    None
wlp2s0:        AddressFamily.AF_INET  192.168.3.125  255.255.255.0  192.168.3.255
enp3s0:        AddressFamily.AF_PACKET 00:e0:91:49:44:d7  None          ff:ff:ff:ff:ff:ff

Pressione qualquer tecla para continuar...

STATUS:

  wlp2s0:      ISUP      DUPLEX      SPEED      MTU
  lo:          True     NicDuplex.NIC_DUPLEX_UNKNOWN  0       1500
  enp3s0:      True     NicDuplex.NIC_DUPLEX_UNKNOWN  0       65536
               True     NicDuplex.NIC_DUPLEX_HALF      10      1500

Pressione qualquer tecla para continuar...■
```

```

alex@alex-ubuntu-pc:/media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_E$ 
ENTRADA E SAIDA DE DADOS:
          BYTES_SENT    BYTES_RECV   PACKETS_SENT   PACKETS_RECV
wlp2s0:        912486     7948666       6731           7839
lo:          683843     683843       2207           2207
enp3s0:         0             0           0             0
Pressione qualquer tecla para continuar...
CONEXÕES:
      FD   FAMILY          TYPE      STATUS      PID
154  AddressFamily.AF_INET  SocketKind.SOCK_STREAM  ESTABLISHED  3780
-1   AddressFamily.AF_INET  SocketKind.SOCK_DGRAM   NONE        None
-1   AddressFamily.AF_INET  SocketKind.SOCK_DGRAM   NONE        None
-1   AddressFamily.AF_INET  SocketKind.SOCK_STREAM  LISTEN      None
119  AddressFamily.AF_INET  SocketKind.SOCK_STREAM  LISTEN      5616
-1   AddressFamily.AF_INET  SocketKind.SOCK_DGRAM   NONE        None
-1   AddressFamily.AF_INET  SocketKind.SOCK_DGRAM   NONE        None
-1   AddressFamily.AF_INET6 SocketKind.SOCK_DGRAM   NONE        None
-1   AddressFamily.AF_INET6 SocketKind.SOCK_STREAM  LISTEN      None
-1   AddressFamily.AF_INET6 SocketKind.SOCK_DGRAM   NONE        None
86   AddressFamily.AF_INET  SocketKind.SOCK_DGRAM   NONE        3780
5    AddressFamily.AF_INET  SocketKind.SOCK_STREAM  ESTABLISHED  6762
-1   AddressFamily.AF_INET  SocketKind.SOCK_DGRAM   NONE        None
-1   AddressFamily.AF_INET6 SocketKind.SOCK_STREAM  LISTEN      None
-1   AddressFamily.AF_INET  SocketKind.SOCK_DGRAM   NONE        None
-1   AddressFamily.AF_INET  SocketKind.SOCK_DGRAM   NONE        None
220  AddressFamily.AF_INET  SocketKind.SOCK_STREAM  LISTEN      5616
87   AddressFamily.AF_INET6 SocketKind.SOCK_DGRAM   NONE        3780
4    AddressFamily.AF_INET  SocketKind.SOCK_STREAM  LISTEN      6762
3    AddressFamily.AF_INET  SocketKind.SOCK_STREAM  ESTABLISHED  6769

Digite zero[0] para voltar ao Menu: ■

```

## Opção 7:

Servidor: alex-ubuntu-pc esperando conexão na porta: 30000  
Conectado a: ('127.0.0.1', 53930)

Requisição da opção 1 recebida com sucesso!

Processando...

Resposta da opção 1 enviada com sucesso!

Requisição da opção 2 recebida com sucesso!

Processando...

Resposta da opção 2 enviada com sucesso!

Requisição da opção 3 recebida com sucesso!

Processando...

Resposta da opção 3 enviada com sucesso!

Requisição da opção 4 recebida com sucesso!

Processando...

Resposta da opção 4 enviada com sucesso!

Requisição da opção 5 recebida com sucesso!

Processando...

Resposta da opção 5 enviada com sucesso!

Requisição da opção 6 recebida com sucesso!

Processando...

Resposta da opção 6 enviada com sucesso!

Conexão encerrada.

Process finished with exit code 0

### 3.8.5.2. Segunda

Menu:

Opção 1:

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_CG
ARQUITETURA DA CPU:
Nome: Intel(R) Core(TM) i5 CPU      M 480 @ 2.67GHz
Arquitetura: X86_64
Palavra (bits): 64
Frequência: 2.2660 GHz
Núcleos: 4 (4)

PERCENTUAL DE USO DA CPU POR NÚCLEO:
Core:0
Percentual: 53.0% |██████████| 
Core:1
Percentual: 2.0% | 
Core:2
Percentual: 44.4% |██████████| 
Core:3
Percentual: 11.0% |██| 

Digite zero[0] para voltar ao Menu: ■
```

Opção 2:

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_
INFORMAÇÕES MEMÓRIA PRINCIPAL:
Capacidade total: 3.72 GB
Capacidade em uso: 1.91 GB
Percentual: 59.8% |██████████| 
Capacidade livre: 0.36 GB
Digite zero[0] para voltar ao Menu: ■
```

### Opção 3:

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/Apresentação
INFORMAÇÕES DISCO RÍGIDO:

Capacidade total: 931.51 GB

Capacidade em uso: 535.69 GB
Percentual: 57.5% |██████████| 57.5

Capacidade disponível: 395.82 GB

Digite zero[0] para voltar ao Menu: ■
```

### Opção 4:

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/Apresentação
DEVICE      POINT MOUNT          FILE SYSTEM
/dev/sda5      /           ext4
/dev/sdb1  /media/alex/Hitachi  fuseblk

TAMANHO        DATA DE CRIAÇÃO          DATA DE MODIFICAÇÃO      NOME
12021.20 KB   Fri Mar 30 18:33:47 2018  Mon Mar 19 02:16:03 2018  a.pdf
0.96 KB       Fri Mar 30 12:38:31 2018  Fri Mar 30 12:38:30 2018  barra_progresso_1.py
7.07 KB       Thu Mar 22 09:24:44 2018  Thu Mar 22 09:23:31 2018  cliente_socket_tcp_v1.0.py
7.47 KB       Thu Mar 22 15:20:48 2018  Thu Mar 22 15:20:08 2018  cliente_socket_tcp_v1.1.py
7.69 KB       Sat Mar 24 08:25:57 2018  Sat Mar 24 08:24:37 2018  cliente_socket_tcp_v1.2.py
12.97 KB      Mon Mar 26 09:14:58 2018  Sun Mar 25 13:39:25 2018  cliente_socket_tcp_v1.3.py
7.69 KB       Sat Mar 24 08:22:28 2018  Fri Mar 23 02:36:42 2018  cliente_socket_tcp_v1.3.py
13.77 KB      Sat Mar 31 18:58:44 2018  Sat Mar 31 18:58:44 2018  cliente_socket_tcp_v1.4.1.py
15.49 KB      Fri Mar 30 20:54:31 2018  Fri Mar 30 20:52:15 2018  cliente_socket_tcp_v1.4.2.py
13.03 KB      Mon Mar 26 09:14:58 2018  Sun Mar 25 18:17:17 2018  cliente_socket_tcp_v1.4.py
3.46 KB       Thu Mar 22 09:24:44 2018  Thu Mar 22 08:13:01 2018  servidor_socket_tcp_v1.0.py
3.08 KB       Thu Mar 22 15:21:28 2018  Thu Mar 22 15:21:09 2018  servidor_socket_tcp_v1.1.py
3.46 KB       Sat Mar 24 08:25:57 2018  Sat Mar 24 08:24:43 2018  servidor_socket_tcp_v1.2.py
4.70 KB       Mon Mar 26 09:14:58 2018  Sun Mar 25 13:42:59 2018  servidor_socket_tcp_v1.3.1.py
3.76 KB       Sat Mar 24 08:22:20 2018  Fri Mar 23 21:53:51 2018  servidor_socket_tcp_v1.3.py
5.07 KB       Sat Mar 31 00:38:45 2018  Sat Mar 31 00:38:45 2018  servidor_socket_tcp_v1.4.1.py
6.15 KB       Fri Mar 30 20:54:42 2018  Fri Mar 30 20:04:14 2018  servidor_socket_tcp_v1.4.2.py
4.70 KB       Mon Mar 26 09:14:58 2018  Sun Mar 25 13:42:59 2018  servidor_socket_tcp_v1.4.py
1.20 KB       Thu Mar 22 09:24:44 2018  Wed Mar 21 01:42:42 2018  teste.py
1.94 KB       Fri Mar 23 00:19:06 2018  Fri Mar 23 00:19:06 2018  teste_dir.py
63.43 KB     Fri Mar 30 20:32:08 2018  Sun Mar 11 23:43:58 2018  x.pdf

Digite zero[0] para voltar ao Menu: ■
```

### Opção 5:

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/Apresentação
PROCESSOS:

PID      Name          %CPU
6428    python        102.9
5616    java          26.0

Digite zero[0] para voltar ao Menu: ■
```

## Opção 6:

```
alex@alex-ubuntu-pc:/media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Proj
INTERFACES DE REDE:

INTERFACES      FAMILY          ADDRESS        NETMASK        BROADCAST
lo:            AddressFamily.AF_INET    127.0.0.1     255.0.0.0      None
wlp2s0:         AddressFamily.AF_INET    192.168.3.125 255.255.255.0  192.168.3.255
enp3s0:         AddressFamily.AF_PACKET  00:e0:91:49:44:d7  None           ff:ff:ff:ff:ff:ff

Pressione qualquer tecla para continuar...

STATUS:

wlp2s0:        ISUP          DUPLEX        SPEED        MTU
enp3s0:         True          NicDuplex.NIC_DUPLEX_UNKNOWN  0       1500
lo:             True          NicDuplex.NIC_DUPLEX_HALF    10      1500
                True          NicDuplex.NIC_DUPLEX_UNKNOWN  0       65536

Pressione qualquer tecla para continuar...■
```

```
alex@alex-ubuntu-pc:/media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Proj
ENTRADA E SAIDA DE DADOS:

          BYTES_SENT   BYTES_RECV  PACKETS_SENT  PACKETS_RECV
wlp2s0:      9692223    46863783    65964        66049
enp3s0:        0          0          0          0
lo:          126824429   126824429   24690        24690

Pressione qualquer tecla para continuar...

CONEXÕES:

FD      FAMILY          TYPE        STATUS        PID
108    AddressFamily.AF_INET6  SocketKind.SOCK_DGRAM  NONE        10810
89     AddressFamily.AF_INET   SocketKind.SOCK_DGRAM  NONE        10810
3      AddressFamily.AF_INET   SocketKind.SOCK_STREAM ESTABLISHED 24057
-1     AddressFamily.AF_INET   SocketKind.SOCK_STREAM LISTEN      None
119    AddressFamily.AF_INET   SocketKind.SOCK_STREAM LISTEN      4015
-1     AddressFamily.AF_INET   SocketKind.SOCK_STREAM LISTEN      None
-1     AddressFamily.AF_INET   SocketKind.SOCK_DGRAM  NONE        None
5      AddressFamily.AF_INET   SocketKind.SOCK_STREAM ESTABLISHED 24029
4      AddressFamily.AF_INET   SocketKind.SOCK_STREAM LISTEN      24029
-1     AddressFamily.AF_INET   SocketKind.SOCK_DGRAM  NONE        None
-1     AddressFamily.AF_INET   SocketKind.SOCK_DGRAM  NONE        None
155    AddressFamily.AF_INET   SocketKind.SOCK_STREAM ESTABLISHED 10810
-1     AddressFamily.AF_INET   SocketKind.SOCK_DGRAM  NONE        None
-1     AddressFamily.AF_INET   SocketKind.SOCK_DGRAM  NONE        None
-1     AddressFamily.AF_INET   SocketKind.SOCK_DGRAM  NONE        None
-1     AddressFamily.AF_INET   SocketKind.SOCK_STREAM LISTEN      None
-1     AddressFamily.AF_INET6  SocketKind.SOCK_STREAM LISTEN      None
-1     AddressFamily.AF_INET6  SocketKind.SOCK_DGRAM  NONE        None
-1     AddressFamily.AF_INET6  SocketKind.SOCK_STREAM LISTEN      None
-1     AddressFamily.AF_INET6  SocketKind.SOCK_DGRAM  NONE        None
220    AddressFamily.AF_INET   SocketKind.SOCK_STREAM LISTEN      4015
-1     AddressFamily.AF_INET6  SocketKind.SOCK_DGRAM  NONE        None
-1     AddressFamily.AF_INET6  SocketKind.SOCK_STREAM LISTEN      None
-1     AddressFamily.AF_INET6  SocketKind.SOCK_STREAM LISTEN      None
```

Opção 7:

```
Servidor: alex-ubuntu-pc esperando conexão na porta: 30000
Conectado a: ('127.0.0.1', 53926)

Requisição da opção 1 recebida com sucesso!
Processando...
Resposta da opção 1 enviada com sucesso!

Requisição da opção 2 recebida com sucesso!
Processando...
Resposta da opção 2 enviada com sucesso!

Requisição da opção 3 recebida com sucesso!
Processando...
Resposta da opção 3 enviada com sucesso!

Requisição da opção 4 recebida com sucesso!
Processando...
Resposta da opção 4 enviada com sucesso!

Requisição da opção 5 recebida com sucesso!
Processando...
Resposta da opção 5 enviada com sucesso!

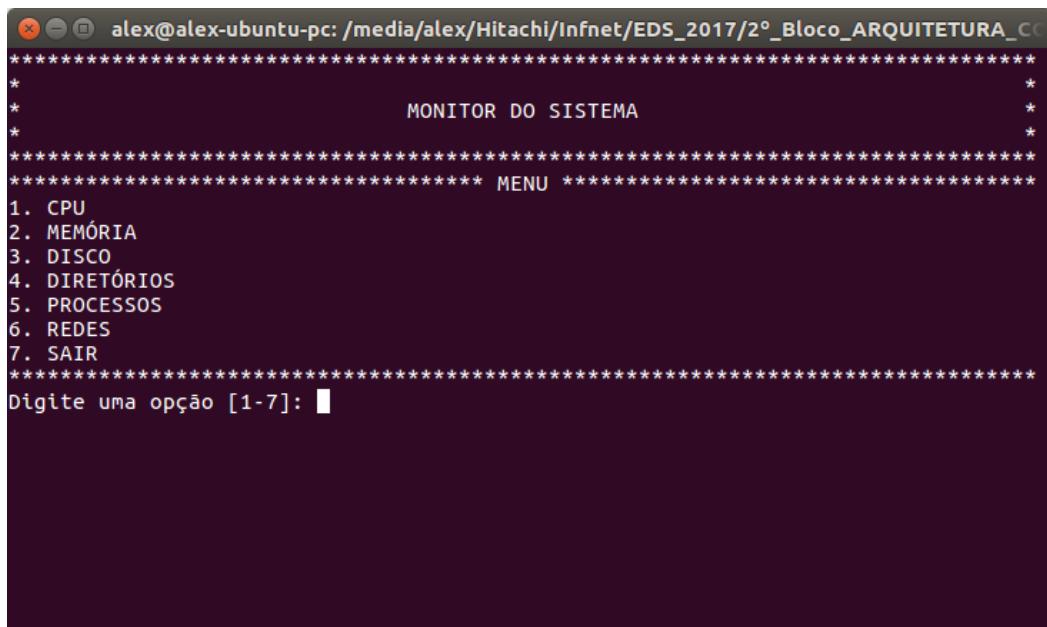
Requisição da opção 6 recebida com sucesso!
Processando...
Resposta da opção 6 enviada com sucesso!

Conexão encerrada.

Process finished with exit code 0
```

### 3.8.5.3. Terceira

Menu:



```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_C
*****
*                               MONITOR DO SISTEMA
*
*****
***** MENU *****
1. CPU
2. MEMÓRIA
3. DISCO
4. DIRETÓRIOS
5. PROCESSOS
6. REDES
7. SAIR
*****
Digite uma opção [1-7]:
```

Opção 1:

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_CO
ARQUITETURA DA CPU:
Nome: Intel(R) Core(TM) i5 CPU      M 480 @ 2.67GHz
Arquitetura: X86_64
Palavra (bits): 64
Frequência: 1.1990 GHz
Núcleos: 4 (4)

PERCENTUAL DE USO DA CPU POR NÚCLEO:
Core:0
Percentual: 1.0% |
Core:1
Percentual: 2.0% |
Core:2
Percentual: 1.0% |
Core:3
Percentual: 3.0% |

Digite zero[0] para voltar ao Menu: |
```

Opção 2:

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2
INFORMAÇÕES MEMÓRIA PRINCIPAL:
Capacidade total: 3.72 GB
Capacidade em uso: 1.93 GB
Percentual: 60.3% | |
Capacidade livre: 0.34 GB
Digite zero[0] para voltar ao Menu: |
```

Opção 3:

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2
INFORMAÇÕES DISCO RÍGIDO:
Capacidade total: 931.51 GB
Capacidade em uso: 535.69 GB
Percentual: 57.5% | |
Capacidade disponível: 395.82 GB
Digite zero[0] para voltar ao Menu: |
```

## Opção 4:

```

alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/Apresentação

ARQUIVOS E DIRETÓRIOS:

DEVICE      POINT MOUNT          FILE SYSTEM
/dev/sda5    /                   ext4
/dev/sdb1    /media/alex/Hitachi  fuseblk

TAMANHO        DATA DE CRIAÇÃO          DATA DE MODIFICAÇÃO          NOME
12021.20 KB   Fri Mar 30 18:33:47 2018 Mon Mar 19 02:16:03 2018 a.pdf
0.96 KB      Fri Mar 30 12:38:31 2018 Fri Mar 30 12:38:30 2018 barra_progresso_1.py
7.07 KB      Thu Mar 22 09:24:44 2018 Thu Mar 22 09:23:31 2018 cliente_socket_tcp_v1.0.py
7.47 KB      Thu Mar 22 15:20:48 2018 Thu Mar 22 15:20:08 2018 cliente_socket_tcp_v1.1.py
7.69 KB      Sat Mar 24 08:25:57 2018 Sat Mar 24 08:24:37 2018 cliente_socket_tcp_v1.2.py
12.97 KB     Mon Mar 26 09:14:58 2018 Sun Mar 25 13:39:25 2018 cliente_socket_tcp_v1.3.py
7.69 KB      Sat Mar 24 08:22:28 2018 Fri Mar 23 02:36:42 2018 cliente_socket_tcp_v1.3.1.py
13.77 KB     Sat Mar 31 18:58:44 2018 Sat Mar 31 18:58:44 2018 cliente_socket_tcp_v1.4.py
15.49 KB     Fri Mar 30 20:54:31 2018 Fri Mar 30 20:52:15 2018 cliente_socket_tcp_v1.4.1.py
13.03 KB     Mon Mar 26 09:14:58 2018 Sun Mar 25 18:17:17 2018 cliente_socket_tcp_v1.4.2.py
3.46 KB      Thu Mar 22 09:24:44 2018 Thu Mar 22 08:13:01 2018 cliente_socket_tcp_v1.4.3.py
3.08 KB      Thu Mar 22 15:21:28 2018 Thu Mar 22 15:21:09 2018 servidor_socket_tcp_v1.0.py
3.46 KB      Sat Mar 24 08:25:57 2018 Sat Mar 24 08:24:43 2018 servidor_socket_tcp_v1.1.py
4.70 KB      Mon Mar 26 09:14:58 2018 Sun Mar 25 13:42:59 2018 servidor_socket_tcp_v1.2.py
3.76 KB      Sat Mar 24 08:22:20 2018 Fri Mar 23 21:53:51 2018 servidor_socket_tcp_v1.3.py
5.07 KB      Sat Mar 31 00:38:45 2018 Sat Mar 31 00:38:45 2018 servidor_socket_tcp_v1.3.1.py
6.15 KB      Fri Mar 30 20:54:42 2018 Fri Mar 30 20:04:14 2018 servidor_socket_tcp_v1.4.py
4.70 KB      Mon Mar 26 09:14:58 2018 Sun Mar 25 13:42:59 2018 servidor_socket_tcp_v1.4.1.py
1.20 KB      Thu Mar 22 09:24:44 2018 Wed Mar 21 01:42:42 2018 teste.py
1.94 KB      Fri Mar 23 00:19:06 2018 Fri Mar 23 00:19:06 2018 teste_dir.py
63.43 KB    Fri Mar 30 20:32:08 2018 Sun Mar 11 23:43:58 2018 x.pdf

Digite zero[0] para voltar ao Menu: ■

```

## Opção 5:

```

alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/Apresentação

PROCESSOS:

          PID          Name          %CPU
        7472        python       102.3

Digite zero[0] para voltar ao Menu: ■

```

```

alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto_Bloco/Apresentação

INTERFACES      FAMILY          ADDRESS          NETMASK          BROADCAST
lo:            AddressFamily.AF_INET  127.0.0.1      255.0.0.0        None
wlp2s0:         AddressFamily.AF_INET  192.168.3.125  255.255.255.0   192.168.3.255
enp3s0:         AddressFamily.AF_PACKET 00:e0:91:49:44:d7  None             ff:ff:ff:ff:ff:ff

Pressione qualquer tecla para continuar...

STATUS:
          ISUP          DUPLEX          SPEED      MTU
wlp2s0:   True   NicDuplex.NIC_DUPLEX_UNKNOWN    0      1500
enp3s0:   True   NicDuplex.NIC_DUPLEX_HALF      10     1500
lo:      True   NicDuplex.NIC_DUPLEX_UNKNOWN    0      65536

Pressione qualquer tecla para continuar...■

```

## Opção 6:

---

```
alex@alex-ubuntu-pc: /media/alex/Hitachi/Infnet/EDS_2017/2º_Bloco_ARQUITETURA_COMP_OS/Projeto
ENTRADA E SAIDA DE DADOS:

      BYTES_SENT    BYTES_RECV   PACKETS_SENT   PACKETS_RECV
wlp2s0:        9600387     46822377       65439         65691
enp3s0:          0            0             0             0
lo:           126763408     126763408      24466         24466

Pressione qualquer tecla para continuar...

CONEXÕES:

   FD      FAMILY          TYPE      STATUS      PID
  89 AddressFamily.AF_INET  SocketKind.SOCK_DGRAM  NONE  10810
 -1 AddressFamily.AF_INET  SocketKind.SOCK_DGRAM  NONE  None
 108 AddressFamily.AF_INET6 SocketKind.SOCK_DGRAM  NONE  10810
 -1 AddressFamily.AF_INET  SocketKind.SOCK_DGRAM  NONE  None
 -1 AddressFamily.AF_INET6 SocketKind.SOCK_STREAM LISTEN  None
  3 AddressFamily.AF_INET  SocketKind.SOCK_STREAM ESTABLISHED 23394
 -1 AddressFamily.AF_INET6 SocketKind.SOCK_DGRAM  NONE  None
 -1 AddressFamily.AF_INET  SocketKind.SOCK_STREAM LISTEN  None
 -1 AddressFamily.AF_INET  SocketKind.SOCK_DGRAM  NONE  None
 88 AddressFamily.AF_INET  SocketKind.SOCK_STREAM ESTABLISHED 10810
 -1 AddressFamily.AF_INET  SocketKind.SOCK_DGRAM  NONE  None
 -1 AddressFamily.AF_INET  SocketKind.SOCK_DGRAM  NONE  None
 -1 AddressFamily.AF_INET  SocketKind.SOCK_STREAM LISTEN  None
 157 AddressFamily.AF_INET  SocketKind.SOCK_STREAM ESTABLISHED  10810
 -1 AddressFamily.AF_INET  SocketKind.SOCK_STREAM LISTEN  None
 -1 AddressFamily.AF_INET  SocketKind.SOCK_DGRAM  NONE  None
 -1 AddressFamily.AF_INET6 SocketKind.SOCK_STREAM LISTEN  None
```

## Opção 7:

```
Servidor: alex-ubuntu-pc esperando conexão na porta: 30000
Conectado a: ('127.0.0.1', 53930)

Requisição da opção 1 recebida com sucesso!
Processando...
Resposta da opção 1 enviada com sucesso!

Requisição da opção 2 recebida com sucesso!
Processando...
Resposta da opção 2 enviada com sucesso!

Requisição da opção 3 recebida com sucesso!
Processando...
Resposta da opção 3 enviada com sucesso!

Requisição da opção 4 recebida com sucesso!
Processando...
Resposta da opção 4 enviada com sucesso!

Requisição da opção 5 recebida com sucesso!
Processando...
Resposta da opção 5 enviada com sucesso!

Requisição da opção 6 recebida com sucesso!
Processando...
Resposta da opção 6 enviada com sucesso!

Conexão encerrada.

Process finished with exit code 0
```

#### **4. CONSIDERAÇÕES FINAIS**

Portanto, com o advento da ARPANET do Departamento de Defesa dos Estados Unidos, a primeira rede operacional de computadores à base de pacotes, precursora da internet, inicialmente, criada para fins militares. Entretanto, no início da década de 1970, universidades e outras instituições tiveram permissão para se conectarem. Então, com o crescimento extraordinário da rede computadores em uma escala global, a informação tornou-se o bem mais precioso das organizações.

Assim, para estudar a arquitetura de sistemas e rede de computadores foi proposto desenvolver uma aplicação com base no modelo cliente-servidor usando o módulo socket e entre outros com apoio da linguagem de programação python, com o objetivo de capturar dados estatísticos. Certamente, que os exercícios e treinamentos propostos pelos Tps, realizados durante o desenvolvimento do projeto foram importantíssimos para agragar conhecimento.

Por conseguinte, o objetivo foi alcançado, pois os resultados obtidos foram satisfatórios e atenderam as expectativas.

## REFERÊNCIAS

PYTHON.ORG. `datetime` Basic date and time types. Disponível em:

<https://docs.python.org/3/library/datetime.html> . Acesso em: 30 mar 2018.

PYTHON.ORG. `os.path` Common pathname manipulations. Disponível em:

<https://docs.python.org/3/library/os.path.html> . Acesso em: 30 mar 2018.

PYTHON.ORG. `psutil` documentation. Disponível em:

<http://psutil.readthedocs.io/en/latest/> . Acesso em: 30 mar 2018.

PYTHON.ORG. `time` Time access and conversions. Disponível em:

<https://docs.python.org/3/library/time.html> . Acesso em: 30 mar 2018.

STACKOVERFLOW.COM.Text Progress Bar in the Console. Disponível em:

<https://stackoverflow.com/questions/3173320/text-progress-bar-in-the-console>.

Acesso em: 31 mar 2018.

PYTHONHELP.wordpress.com. Desempacotamento de Tuplas. Disponível em:

<https://python help.wordpress.com/2013/01/10/desempacotamento-de-tupla/>.

Acesso em: 31 mar 2018.

PYGAME.ORG. Pygame Documentation. Disponível em:

<https://www.pygame.org/docs/> . Acesso em 04 de fevereiro de 2018

PYTHON.ORG. `subprocess` Subprocess management. Disponível em:

<https://docs.python.org/3/library/subprocess.html?highlight=subprocess> .

Acesso em: 10 fev 2018.