UNISINOS

DESAFIE
O AMANHÃ.

Automatizando um laboratório de eletrônica com Python (PyVISA)

SAEP 2023

Sobre

Armando Leopoldo Keller

Engenheiro eletricista e professor dos cursos da área eletroeletrônica

Graduação em Engenharia elétrica com ênfase em controle e automação
Mestrado em engenharia elétrica
Doutorado em Computação Aplicada (em andamento)

Agenda

- A linguagem python e o ambiente de desenvolvimento
- Plotando gráficos em python com matplotlib
- Protocolo VISA
- Controlando o osciloscópio através do protocolo VISA
- Controlando o gerador de funções através do protocolo VISA
- Desafio: Automatizando um teste utilizando múltiplos equipamentos

A linguagem python e o ambiente de desenvolvimento

- Linguagem bastante popular e de fácil aprendizado
- Como vamos acessar equipamentos físicos através da USB, é necessário ter a instalação do interpretador em ambiente local e com os privilégios de acesso aos dispositivos USB.
- Para manter a compatibilidade dos códigos em diferentes ambientes é recomendado utilizar algum gerenciador de ambientes virtuais. Os exemplos serão feitos utilizando o Poetry (https://python-poetry.org/).

A linguagem python e o ambiente de desenvolvimento

- Os códigos dos exemplos são baseados nos exemplos oficiais da tektronix, disponíveis em https://github.com/tektronix/Programmatic-Control-Examples

- Será necessário instalar as bibliotecas:
  - pyVisa (Comunicação com os dispositivos)
  - Matplotlib (Geração de gráficos)
  - Numpy (funções matemáticas mais avançadas)

A linguagem python e o ambiente de desenvolvimento

- Os comandos básicos de python serão retomados ao longo dos exemplos.
- Instalador do python disponível em https://www.python.org/
- O editor utilizado será o Visual Studio Code com a extensão para python, mas pode ser utilizado qualquer editor de textos ou IDE como notepad++, pyCharm...
- Os códigos do exemplo, e este material estão disponíveis em https://github.com/armandokeller/SAEP2023-pyvisa

Visualização de dados

Visualização de dados

Algumas bibliotecas do python como a matplotlib, numpy, sympy e pandas (todas disponíveis dentro do pacote scipy ou integradas ao winpython) tornam o python uma ferramenta gratuita alternativa ao Matlab.

Visualização de dados

Um dos principais recursos utilizados do matplotlib é o módulo pyplot, que pode ser importado utilizando o seguinte comando.

```
import matplotlib.pyplot as plt
```

Deste modo, demos um apelido (“plt”) para o módulo e ele está disponível para ser utilizado.

Visualização de dados

O módulo pyplot (que agora está disponível pelo apelido plt), possui a função **plot()** que recebe os dados do gráfico a ser criado, podendo receber como argumentos:
a) Valores do eixo vertical, será considerado o passo de uma unidade no eixo horizontal.

b) Valores do eixo vertical e formatação, além de uma lista com os valores a serem plotados, recebe a string de formatação ou demais parâmetros de formatação.

Visualização de dados

c) Valores do eixo horizontal, Valores do eixo vertical: Ao receber duas listas, a primeira será considerada a do eixo horizontal, e a segunda do eixo vertical.

d) Valores do eixo horizontal, Valores do eixo vertical e formatação, além dos valores dos eixos podem ser passados os parâmetros de formatação.

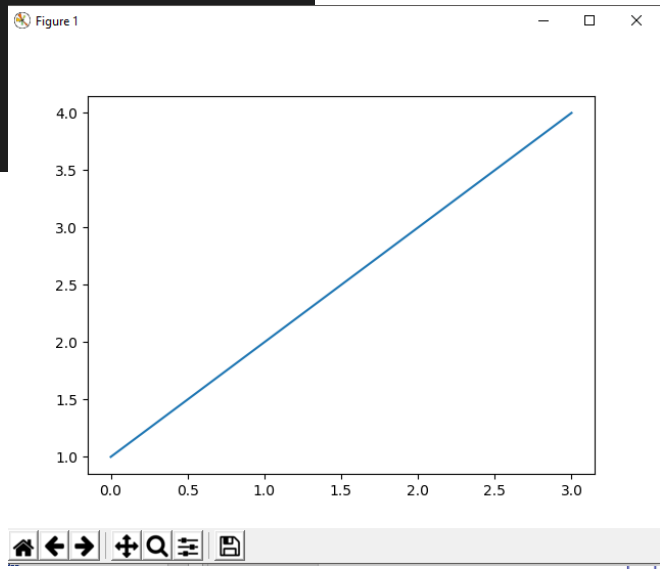
Visualização de dados

Uma vez que a função plot foi chamada, o gráfico já foi criado, mas este só estará visível após a chamada da função show().

Visualização de dados

Exemplos:
Caso a)

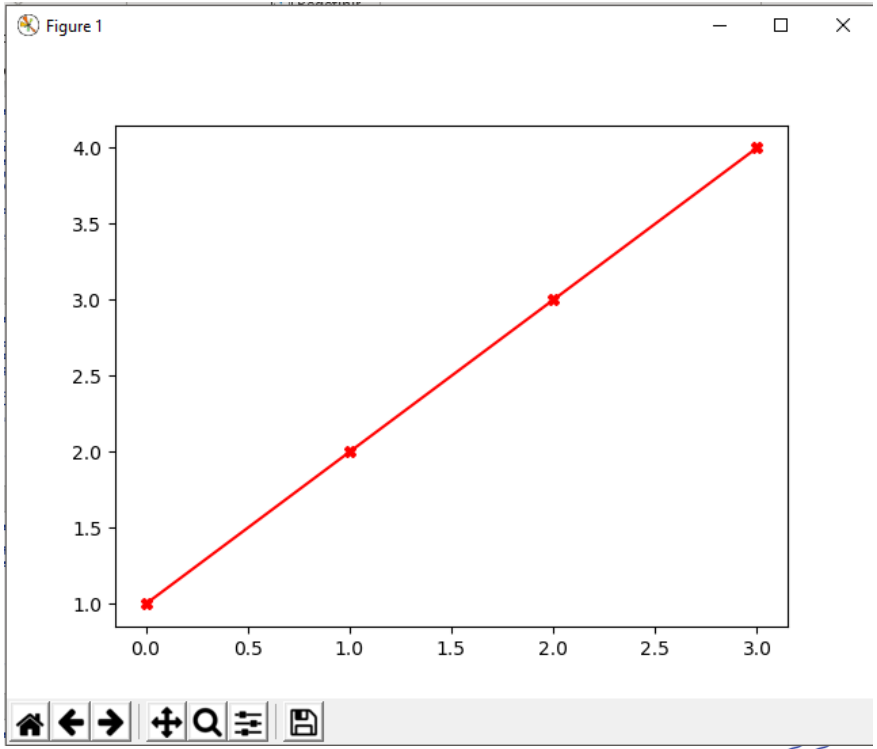
```
import matplotlib.pyplot as plt
valores = [1,2,3,4]
plt.plot(valores)
plt.show()
```



Visualização de dados

Exemplos:
Caso b)

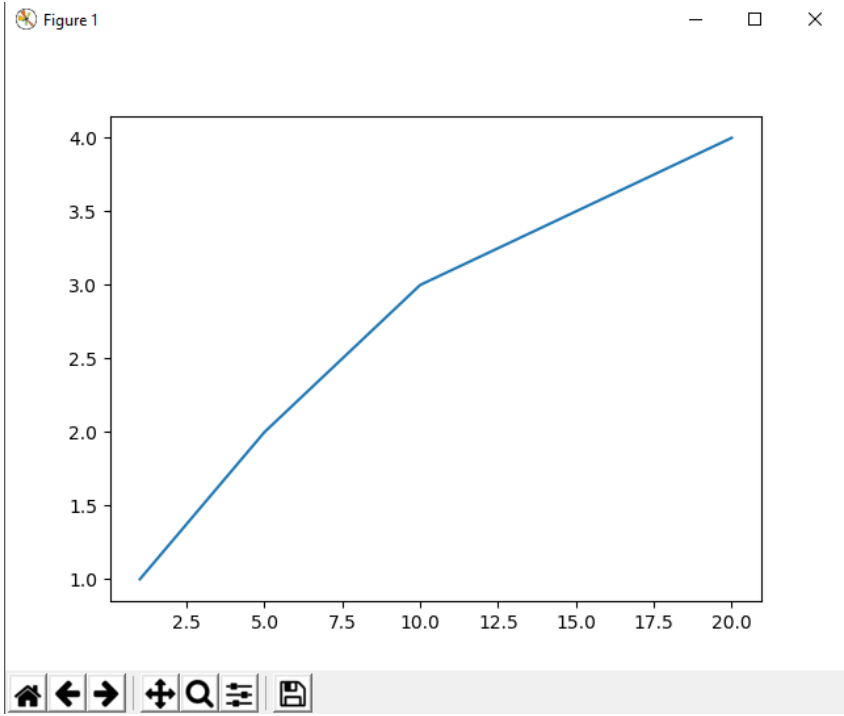
```
import matplotlib.pyplot as plt
valores = [1,2,3,4]
plt.plot(valores,"Xr-")
plt.show()
```



Visualização de dados

Exemplos:
Caso c)

```
import matplotlib.pyplot as plt
valoresX = [1,5,10,20]
valoresY = [1,2,3,4]
plt.plot(valoresX, valoresY)
plt.show()
```

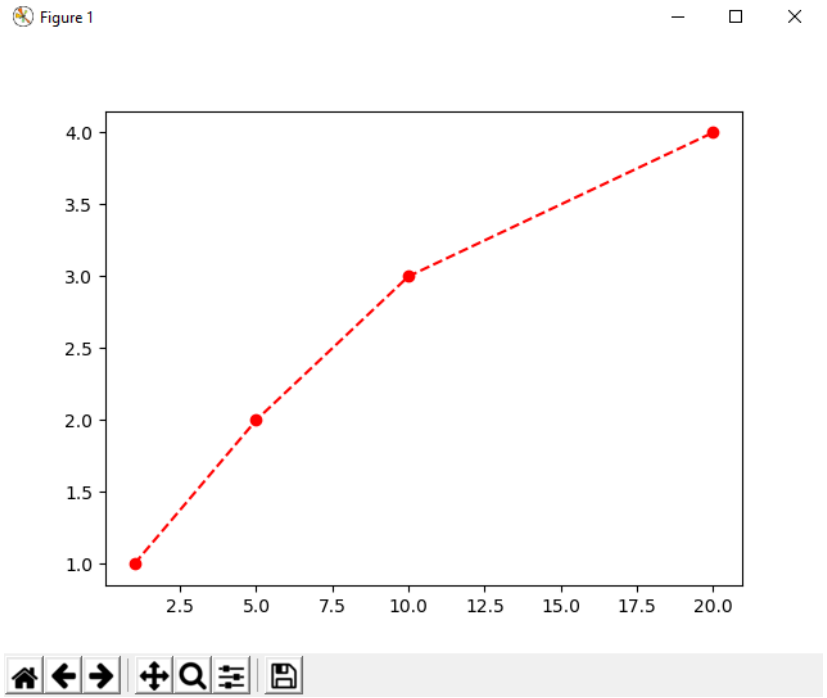


UNISINOS
DESAFIE O AMANHÃ.

Visualização de dados

Exemplos:
Caso d)

```
import matplotlib.pyplot as plt
valoresX = [1,5,10,20]
valoresY = [1,2,3,4]
plt.plot(valoresX, valoresY,"ro--")
plt.show()
```



Formatação básica de gráficos

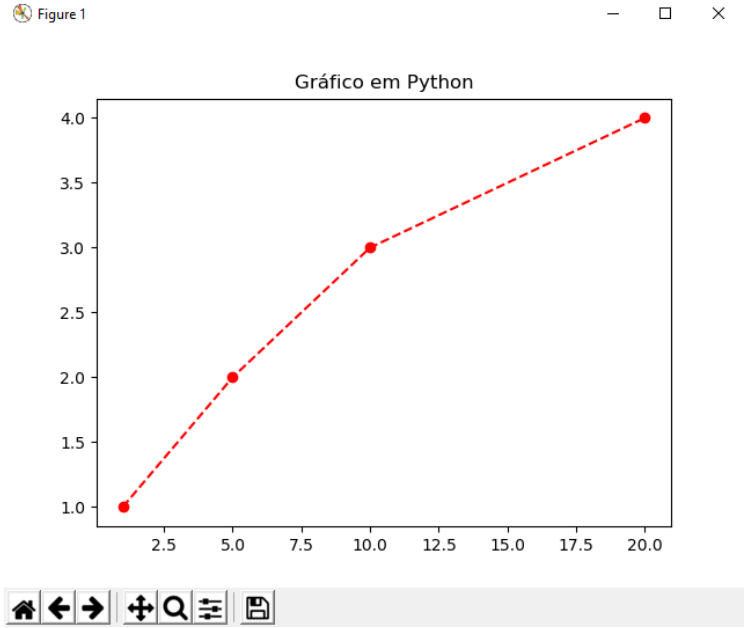
Formatação básica de gráficos

Para inserir um título no nosso gráfico, basta chamar a função **title()** passando como argumento o título a ser exibido.
Este título será posicionado acima do gráfico.

Formatação básica de gráficos

Exemplo:

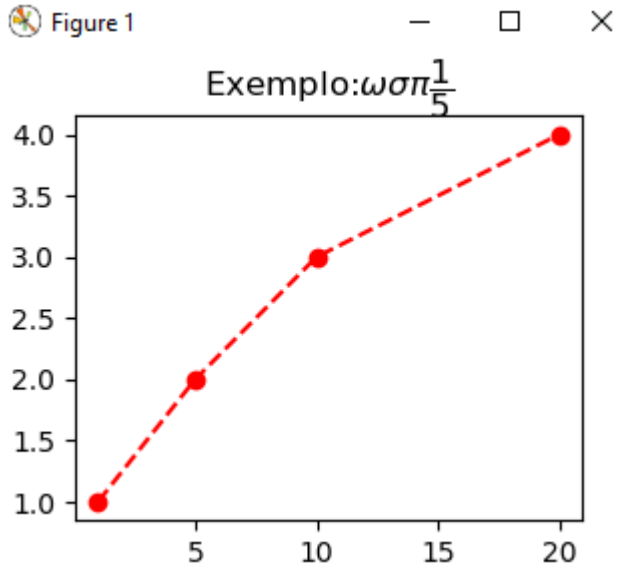
```
import matplotlib.pyplot as plt
valoresX = [1,5,10,20]
valoresY = [1,2,3,4]
plt.plot(valoresX, valoresY,"ro--")
plt.title("Gráfico em Python")
plt.show()
```



Formatação básica de gráficos

Além de títulos comuns, o matplotlib aceita textos em formato LaTeX, quando passado entre cifrões.

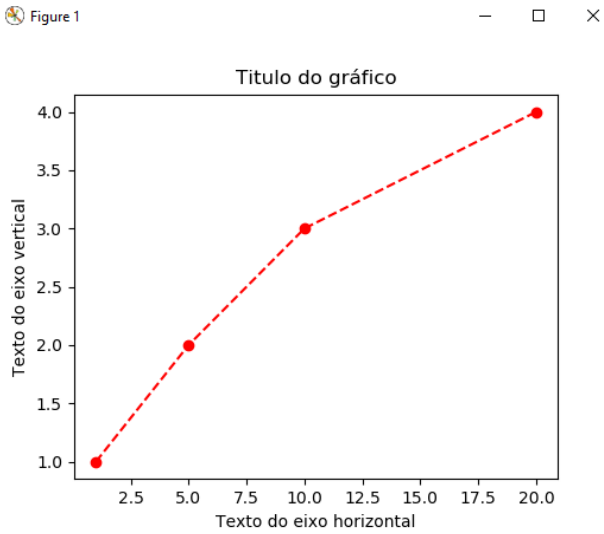
```
import matplotlib.pyplot as plt
valoresX = [1,5,10,20]
valoresY = [1,2,3,4]
plt.plot(valoresX, valoresY,"ro--")
plt.title("Exemplo:$\omega \sigma \pi \dfrac{1}{5}$")
plt.show()
```



Formatação básica de gráficos

Os títulos dos eixos horizontal e vertical podem ser modificados com o comando plt.xlabel() e plt.ylabel()

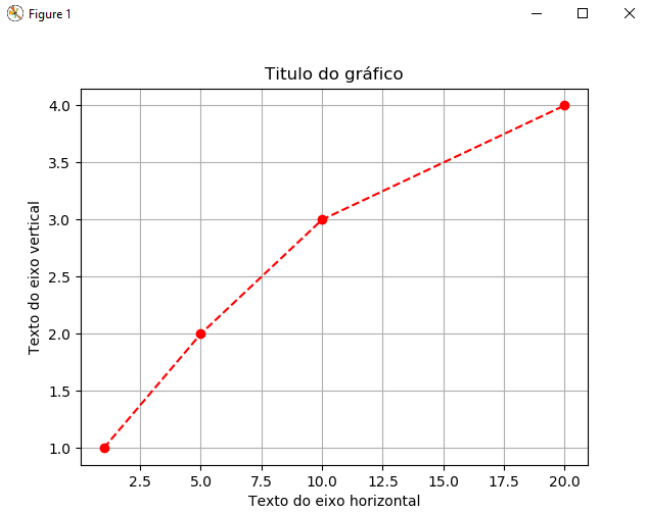
```
import matplotlib.pyplot as plt
valoresX = [1,5,10,20]
valoresY = [1,2,3,4]
plt.plot(valoresX, valoresY,"ro--")
plt.title("Titulo do gráfico")
plt.xlabel("Texto do eixo horizontal")
plt.ylabel("Texto do eixo vertical")
plt.show()
```



Formatação básica de gráficos

O grid pode ser exibido ou ocultado através da função plt.grid() que recebe como argumento um valor booleano referente ao status do grid.

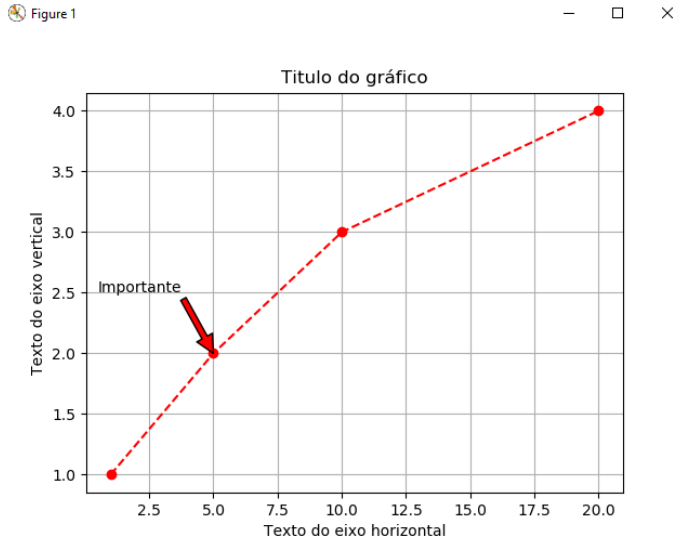
```
import matplotlib.pyplot as plt
valoresX = [1,5,10,20]
valoresY = [1,2,3,4]
plt.plot(valoresX, valoresY,"ro--")
plt.title("Titulo do gráfico")
plt.xlabel("Texto do eixo horizontal")
plt.ylabel("Texto do eixo vertical")
plt.grid(True)
plt.show()
```



Formatação básica de gráficos

Podem ser inseridas anotações no gráfico para destacar algum ponto, utilizando o comando plt.annotate()

```
import matplotlib.pyplot as plt
valoresX = [1,5,10,20]
valoresY = [1,2,3,4]
plt.plot(valoresX, valoresY,"ro--")
plt.title("Titulo do gráfico")
plt.xlabel("Texto do eixo horizontal")
plt.ylabel("Texto do eixo vertical")
plt.grid(True)
plt.annotate(    'Importante',
                 xy=(5, 2),
                 xytext=(0.5, 2.5),
                 arrowprops=dict(facecolor='red'),
             )
plt.show()
```



Formatação básica de gráficos

Para inserir legendas no gráfico, é necessário dar um label a cada linha plotada, isto pode ser feito com o parâmetro label que pode ser passado no momento da criação da linha ou posteriormente na inserção das legendas (que seguirá a ordem na qual as linhas foram plotadas).

Para exibir as legendas, usa-se o comando plt.legend(), que pode receber a lista de legendas, e até mesmo a posição na qual estas devem ser posicionadas.

Formatação básica de gráficos

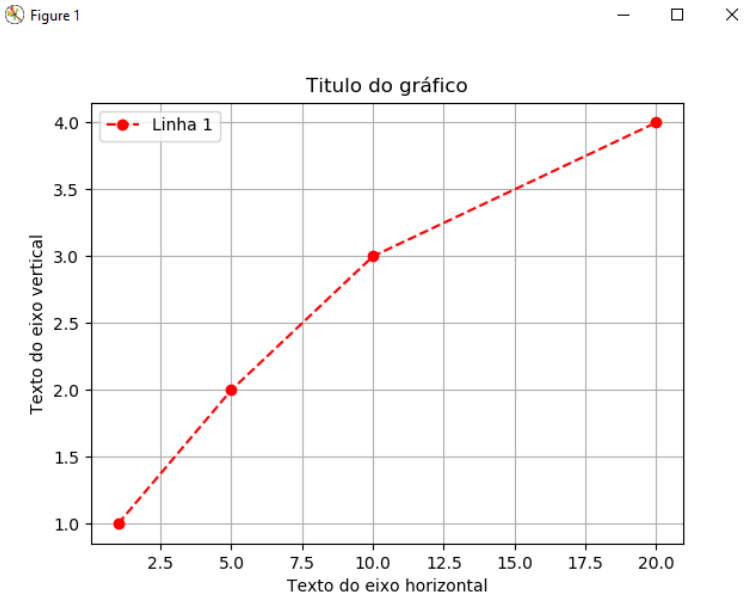
As posições possíveis são passadas no argumento loc

Location String	Location Code
'best'	0
'upper right'	1
'upper left'	2
'lower left'	3
'lower right'	4
'right'	5
'center left'	6
'center right'	7
'lower center'	8
'upper center'	9
'center'	10

Formatação básica de gráficos

Exemplo com posicionamento padrão

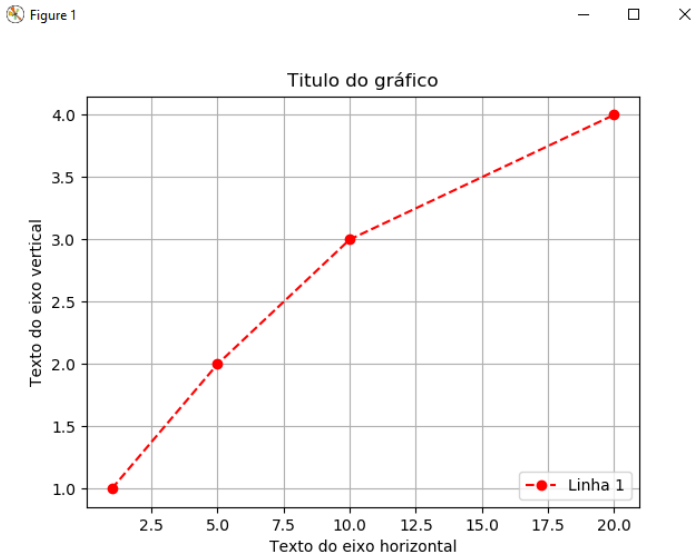
```
import matplotlib.pyplot as plt
valoresX = [1,5,10,20]
valoresY = [1,2,3,4]
plt.plot(valoresX, valoresY,"ro--", label="Linha 1")
plt.title("Titulo do gráfico")
plt.xlabel("Texto do eixo horizontal")
plt.ylabel("Texto do eixo vertical")
plt.grid(True)
plt.legend()
plt.show()
```



Formatação básica de gráficos

Exemplo com posicionamento definido

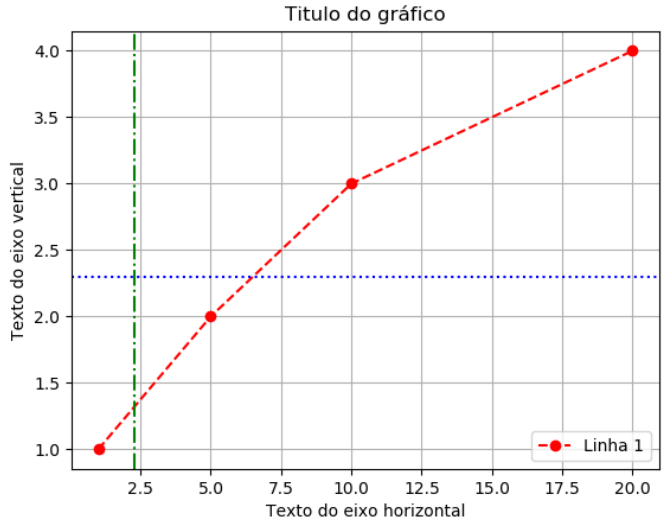
```
import matplotlib.pyplot as plt
valoresX = [1,5,10,20]
valoresY = [1,2,3,4]
plt.plot(valoresX, valoresY,"ro--", label="Linha 1")
plt.title("Titulo do gráfico")
plt.xlabel("Texto do eixo horizontal")
plt.ylabel("Texto do eixo vertical")
plt.grid(True)
plt.legend(loc="lower right")
plt.show()
```



Formatação básica de gráficos

Para adicionar linhas horizontais e/ou verticais no gráfico, podem ser utilizados os comandos plt.axhline() ou plt.axvline()

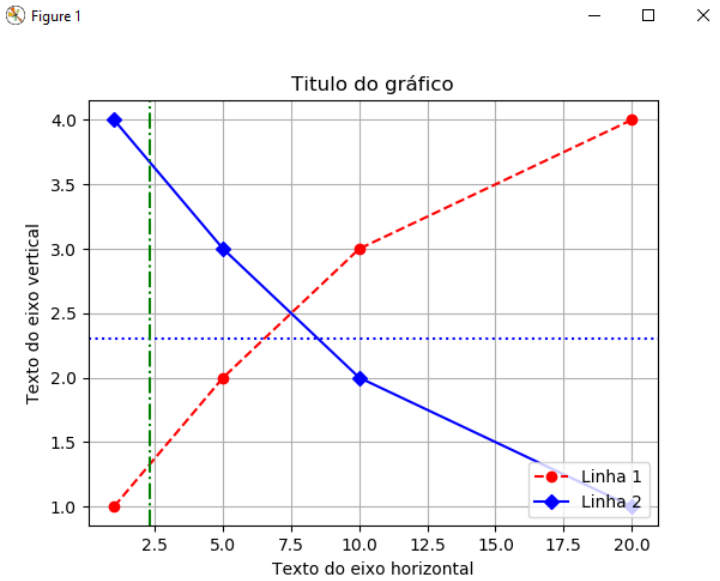
```
import matplotlib.pyplot as plt
valoresX = [1,5,10,20]
valoresY = [1,2,3,4]
plt.plot(valoresX, valoresY,"ro--", label="Linha 1")
plt.title("Titulo do gráfico")
plt.xlabel("Texto do eixo horizontal")
plt.ylabel("Texto do eixo vertical")
plt.axhline(2.3, ls=':', color='b')
plt.axvline(2.3, ls="-.", color='g')
plt.grid(True)
plt.legend(loc="lower right")
plt.show()
```



Formatação básica de gráficos

Podem ser plotadas mais de uma linha de dados no mesmo gráfico, basta utilizar o comando plot novamente.

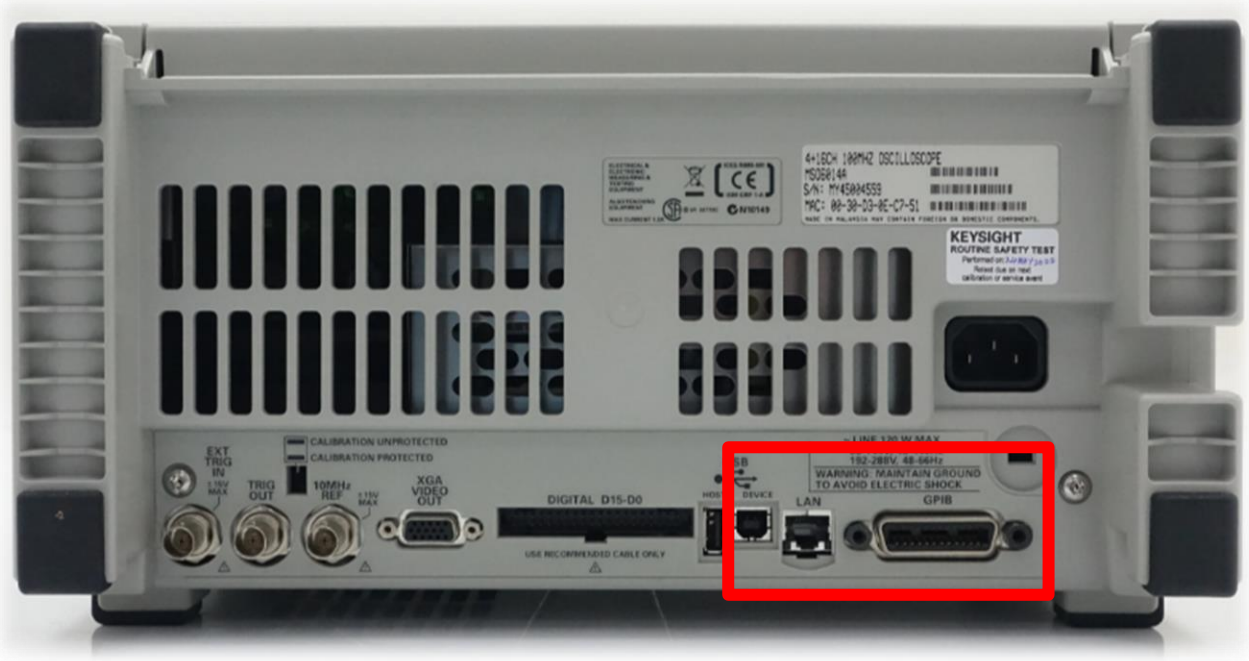
```
import matplotlib.pyplot as plt
valoresX = [1,5,10,20]
valoresY = [1,2,3,4]
valoresY2 = [4,3,2,1]
plt.plot(valoresX, valoresY,"ro--", label="Linha 1")
plt.plot(valoresX, valoresY2, "bD-", label="Linha 2")
plt.title("Titulo do gráfico")
plt.xlabel("Texto do eixo horizontal")
plt.ylabel("Texto do eixo vertical")
plt.axhline(2.3, ls=':', color='b')
plt.axvline(2.3, ls="-.", color='g')
plt.grid(True)
plt.legend(loc="lower right")
plt.show()
```



Protocolo VISA

- VISA (Virtual Instrument Software Architecture)
- Especificação oficial do protocolo: https://www.ivifoundation.org/specifications/default.aspx
- Permite a comunicação por diferentes interfaces com GPIB, USB e Ethernet

Protocolo VISA



UNISINOS
DESAFIE O AMANHÃ.

Protocolo VISA

- Apesar de ser um protocolo bem especificado, é necessário enviar comandos específicos para os equipamentos, os quais dependem da implementação do fabricante.

- O ideal é consultar o manual do programador de cada equipamento, para os exemplos apresentados foram utilizados os manuais de programador "AFG1000 Series Arbitrary Function Generator Programmer Manual" e "TBS2000 Series Digital Oscilloscopes Programmer"

Protocolo VISA

- Os comandos seguem a notação do formalismo de Backus-Naur (BNF)

Símbolo	Significado
< >	Elemento definido
:=	É definido como
|	Ou exclusivo
{ }	Grupo; Um elemento é requerido
[ ]	Opcional; Pode ser omitido
...	Elemento(s) anterior(es) podem ser repetidos
( )	Comentário

Protocolo VISA

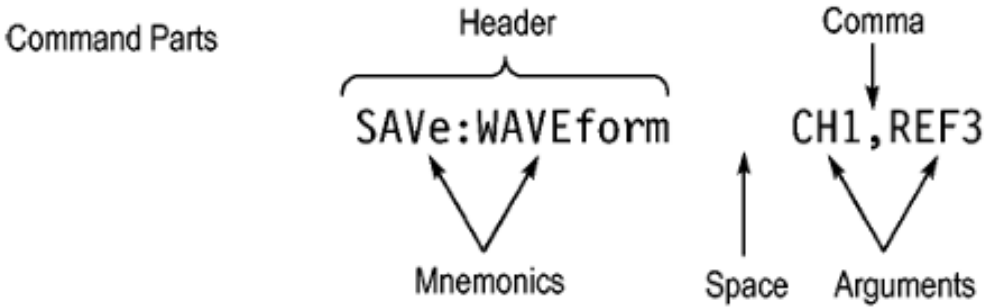
- Os comandos podem ser divididos em comandos de definição ou de consulta
- Os comandos de definição (set commands ou só commands) definem um valor, enquanto as consultas (query commands ou só query) retornam algum valor do equipamento.
- Alguns comandos executam as duas funções.

Protocolo VISA

Símbolo	Significado
<Cabeçalho>	O nome do comando básico. Se o cabeçalho terminar com uma interrogação, é uma consulta. O cabeçalho pode começar com dois pontos (:); Se o comando for concatenado com outro comando é nessesário utilizar uma vírgula. A vírgula de início nunca pode ser utilizada com comandos iniciados com asterisco (*).
<Mnemônico>	Uma subfunção do cabeçalho. Alguns cabeçalhos de comandos possuem somente um mnemônico. Se o cabeçalho tiver múltiplos mnemônicos, eles deve ser sempre separados entre si por dois pontos (:).
<Argumento>	Uma quantidade, qualidade, restrição ou limite associado ao cabeçalho. Nem todos os comandos possuem um argumento, enquanto outros comandos possuem múltiplos argumentos. Argumentos são separados do cabeçalho por um espaço. Argumentos são separados entre si por uma vírgula.

Protocolo VISA

- Elementos de uma mensagem de comando



Protocolo VISA

- Os comandos podem ser arranjados em uma forma hierárquica ou em estrutura de árvore, onde o primeiro mnemônico é a base ou raiz da árvore.

[:]<Header>

[:]<Header>[<Space><Argument>[<Comma><Argument>]...]

Protocolo VISA

- Exemplos de comandos:
  - MEASUrement:MEAS<x>:UNIts? Retorna a unidade de medida
  - MEASUrement:MEAS<x>:TYPe? Retorna o tipo de medida selecionada
  - MEASUrement:MEAS<x>? Retorna todos os parâmetros de medida

Protocolo VISA

- O comando HEADer pode ser utilizado para habilitar ou desabilitar o envio dos cabeçalhos nas repostas da consulta. Isto pode facilitar a interpretação das respostas.

- Exemplos:

Consulta	Resposta com cabeçalho desabilitado	Resposta com o cabeçalho habilitado
ACQuire:NUMAVg	64	ACQUIRE:NUMAVG 64
CHx1:COUPling	DC	CH1:COUPLING DC

Protocolo VISA

- Muitos comandos podem ser abreviados para a forma destacada em letras maiúsculas. O comando ACQuire:NUMAvg pode ser inserido somente como ACQ:NUMA ou até mesmo acq:numa.

- É possível concatenar qualquer combinação de comandos utilizando um ponto e vírgula (;) Os comandos são executados na ordem em que o equipamento os recebe. No entanto algumas regras devem ser seguidas ao concatenar comandos.

Protocolo VISA

- Cabeçalhos completamente diferentes devem ser separados tanto pelo ponto e vírgula quanto pelo dois pontos no início de cada comando exceto o primeiro.
    - Exemplo: TRIGger:MODe NORMal;:ACQuire:NUMAVg 16
- Se os cabeçalhos a serem concatenados diferirem somente pelo último mnemônico, o segundo comando pode ser abreviado e o dois pontos pode ser removido. No entanto a versão completa também funciona.
    - Exemplo: ACQuire:MODe AVErage; NUMAVg 16
    - Exemplo completo: ACQuire:MODe AVErage;:ACQuire:NUMAVg 16

Protocolo VISA

- Nunca utilizar ponto e vírgula ou dois pontos antes de comandos iniciados por asterisco. O equipamento vai ignorar os comandos.
- Consultas concatenadas possuem seus resultados concatenados.
- Comandos de definição e consulta podem ser concatenados.
- Qualquer consulta que retorne dados arbitrários como ID, devem ser o último item da concatenação, caso contrário o equipamento retornará o erro 440.

Protocolo VISA

- Instalando a biblioteca PyVISA
- https://pyvisa.readthedocs.io/projects/pyvisa-py/en/latest/installation.html

- pip install pyvisa-py

Protocolo VISA

- O primeiro passo é importar a biblioteca utilizando o comando

```
import pyvisa
```

- A comunicação com o dispositivo é realizada através de um gerenciador de recursos. Ele pode ser instanciado com o seguinte comando:

```
rm = pyvisa.ResourceManager('@py')
```

Protocolo VISA

- Para saber quais equipamentos estão conectados, é chamado o método list_resources do objeto ResourceManager. Ele devolve uma lista de identificadores únicos dos equipamentos, assim como a interface pela qual ele está acessível.

- Exemplo:

```
rm.list_resources()
('USB0::0x1AB1::0x0588::DS1K00005888::INSTR')
```

Protocolo VISA

- De posse dos identificadores, é possível abrir o recurso (equipamento), para isto é criada uma instância de recurso para cada equipamento através do gerenciador de recursos.

- Exemplo:

```
inst = rm.open_resource('USB0::0x1AB1::0x0588::DS1K00005888::INSTR’)
print(inst.query("*IDN?"))
```

Controlando o Osciloscópio

Objetivo:

- Executar autoset
- Adquirir forma de onda
- Plotar a forma de onda com o matplotlib

Controlando o gerador de funções

Objetivo exemplo 1:

- Alterar o tipo de forma de onda e os seus parâmetros

Objetivo exemplo 2:

- Enviar uma forma de onda arbitrária

Desafio - Integração de equipamentos

Crie um script que:

- modifique a saída do gerador de funções para enviar senoides de diferentes frequências.
- Receba os dados dos canais 1 e 2 do osciloscópio e meça a amplitude dos sinais e a defasagem (pode ser utilizada a medida do osciloscópio, ou pode ser feita com os dados extraídos).

Aplique o sinal em um filtro RC e plote um gráfico da variação da amplitude entre a saída e a entrada pela frequência.

OBRIGADO.

UNISINOS

DESAFIE
O AMANHÃ.