ATIVIDADE LIVRE - TEMA 08

Gráficos de linha com a biblioteca Matplotlib (Python)

CCET | ufs
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

EVANDRO BATISTA CIQUEIRA FILHO

PROGRAMAÇÃO IMPERATIVA

AULA 01

APRESENTANDO A BIBLIOTECA MATPLOTLIB

Uma das criações mais úteis da humanidade para apresentação de dados foi de fato os gráficos. Existem variados tipos gráficos que podem nos auxiliar nas tomadas de decisão do dia a dia. E quando trazemos isso para o mundo da Programação, devemos entender que ela por si só vem para auxiliar em um melhor tratamento dos dados, assim como integrar gráficos de forma automática em alguns processos. Já pensou se os grandes investidores precisassem plotar manualmente cada gráfico que eles precisassem pra avaliação de mercado? Seria muito trabalhoso!

Na programação em Python temos uma **biblioteca** que nos auxilia a plotarmos gráficos partindo de uma lista dada de **variáveis coordenadas**. Sim! Os nossos famosos dados **x** e **y**.

Esse mini curso tem como finalidade ensiná-lo a gerar **gráficos de linha** com o Python por meio da biblioteca **Matplotlib**. Na aula 01 faço uma apresentação breve da biblioteca, de forma a motivá-lo a utilizá-la sempre que desejar plotar algum gráfico.

ESCANEIE O CÓDIGO PARA TER ACESSO AO VÍDEO DA AULA 01 OU **CLIQUE AQUI**!

Para começar a usar a biblioteca Matplotlib temos que instalá-la. Portanto, caso você utilize um ambiente de desenvolvimento **offline** garanta que a biblioteca esteja **devidamente instalada** em seu computador. Durante esse minicurso utilizaremos o ambiente de desenvolvimento online **repl.it**, onde ele instala automaticamente a biblioteca.

Já instalamos a biblioteca, vamos começar a usá-la. Nossa biblioteca é a matplotlib. Nós queremos utilizar ela para criar, isto é, plotar, gráficos. Então, falamos para o Python importar (import) da biblioteca matplotlib a parte de plotar gráficos com o Python (pyplot) como "plt", espécie de apelido que damos a biblioteca para facilitar o seu uso enquanto escrevemos o código.

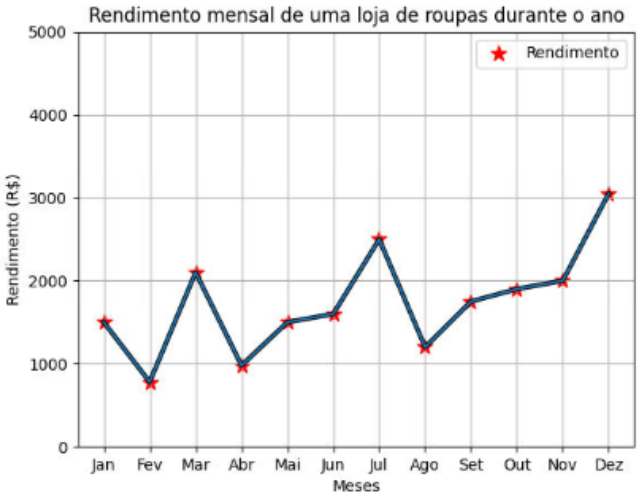
```
>>> import matplotlib.pyplot as plt
```

Nosso minicurso terá como base um exemplo. Confira!

Para analisarmos o rendimento mensal de uma loja de roupas durante o período de um ano podemos criar um gráfico de linhas. Observe o quadro abaixo que relaciona o mês e o respectivo rendimento:

Janeiro	1500
Fevereiro	780
Março	2100
Abril	980
Maio	1500
Junho	1600
Julho	2500
Agosto	1200
Setembro	1750
Outubro	1900
Novembro	2000
Dezembro	3050

Por meio da biblioteca Matplotlib podemos gerar o gráfico abaixo com tranquilidade. Quer aprender como se faz isso? Acompanhe as aulas seguintes!



OBS: Gráfico plotado com Python no Ambiente de Desenvolvimento Repl.it.

AULA 02

PLOTANDO UM GRÁFICO DE LINHAS COM A BIBLIOTECA MATPLOTLIB

Lembra do nosso exemplo anterior? Vamos dar prosseguimento a ele aprendendo a plotar o gráfico!

Após ter importado a biblioteca Matplotlib, ou melhor, a "plt", apelido carinhoso que demos para evitar repetir uma palavra tão grande, **precisamos formar duas listas** com os dados que queremos utilizar para plotar o gráfico. Fique livre para montar as listas como queira! Usando laços para coletar os dados entrados, lendo algum arquivo com o próprio Python ou até mesmo escrevendo a lista diretamente no código (que será o nosso caso!).

Uma lista corresponderá aos dados do eixo horizontal, também conhecido como eixo das abscissas e a outra corresponderá aos dados do eixo vertical ou eixo das ordenadas. No caso do nosso exemplo:

```
>>> meses = ['Jan', 'Fev', 'Mar', 'Abr', 'Mai', 'Jun', 'Jul', 'Ago', 'Set', 'Out', 'Nov', 'Dez']
>>> receitas = [1500, 780, 2100, 980, 1500, 1600, 2500, 1200, 1750, 1900, 2000, 3050]
```

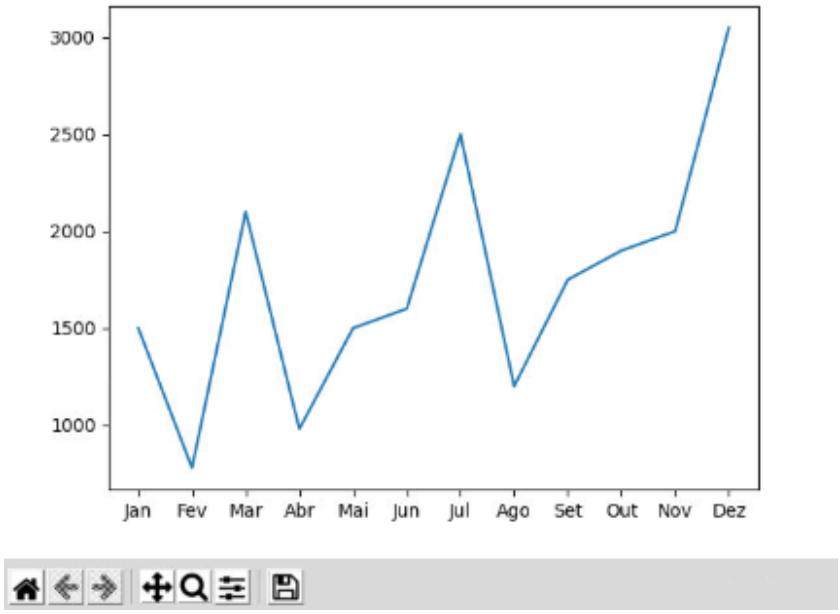
Agora, basta falarmos para o pyplot plotar (plot) o nosso gráfico:

```
>>> plt.plot(meses, receitas)
```

Se você rodar o seu código com toda certeza não verá a imagem do gráfico. Então onde ele se encontra já que você mandou plotar? Quando falamos para o pyplot criar o gráfico, ele constrói o gráfico e o guarda em uma região da memória. Para conseguirmos ver o gráfico, precisamos falar para o pyplot mostrá-lo (show()):

```
>>> plt.show()
```

Agora é só rodar o código!



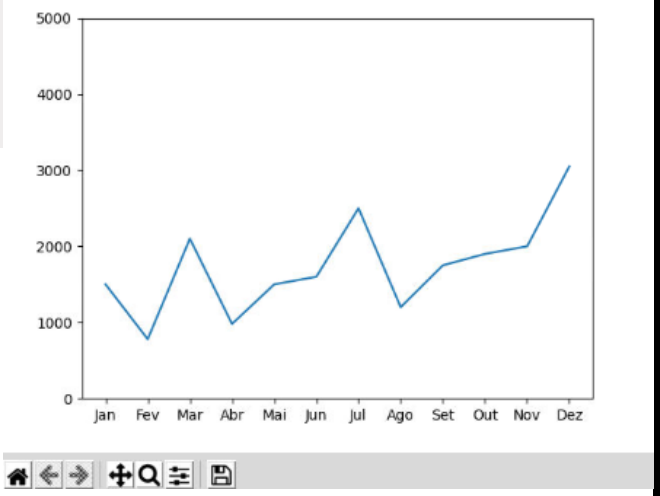
O método **plot** cria um gráfico de linhas. Podemos ver que os meses ficaram na parte horizontal do gráfico, isto é, no eixo X. Enquanto os valores ficaram na parte vertical do gráfico, ou seja, no eixo Y.

Com o gráfico, podemos ver de uma forma mais clara como foram as vendas em cada mês no primeiro semestre.

Nós podemos estabelecer um limite inferior e superior nos eixos x e y. Por exemplo, caso eu queira que meus valores de rendimento estejam em uma faixa de 0 a 5000 no gráfico, basta utilizarmos o limitador (**ylim()**):

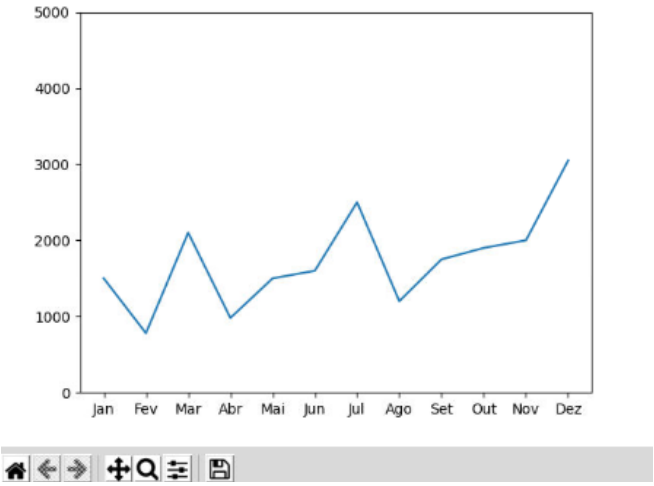
```
>>> plt.plot(meses, receitas)
>>> plt.ylim(0, 5000)
>>> plt.show()
```

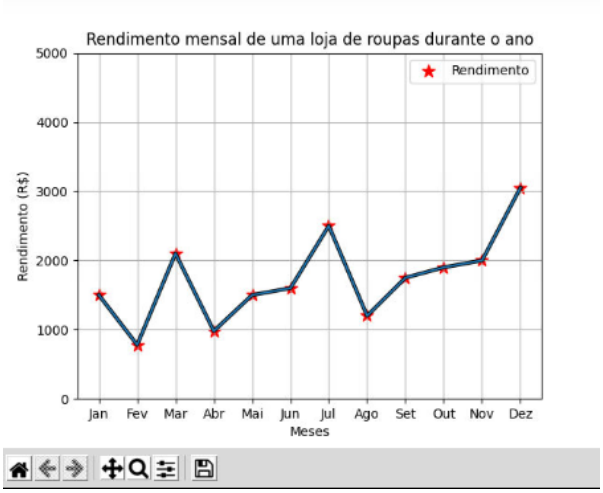
Ao rodar o código o gráfico ficaria como o apresentado ao lado:



Analogamente podemos limitar o eixo x utilizando o limitador **xlim()** desde que estejamos trabalhando com valores numéricos nesse eixo.

Ainda assim o gráfico não está tão apresentável como o que vimos da Aula 01, concorda? Olha a diferença entre eles abaixo!





Aprenderemos na Aula 03 a deixar o gráfico apresentável! Porém antes disso, tenho uma atividade prática para você! Se tiver dúvidas, basta assistir o vídeo da Aula 02 disponibilizado abaixo!

ESCANEIE O CÓDIGO PARA TER ACESSO AO VÍDEO DA AULA 02 OU **CLIQUE AQUI**!

EXERCITE!

Fazendo uso da biblioteca Matplotlib com Python, elabore um gráfico de linhas de posição em função do tempo de uma partícula em movimento. Você tem a liberdade de criar os seus dados. Use os limites superiores e inferiores nos eixos do seu gráfico para uma melhor visualização dos dados.

AULA 03

DEIXANDO O NOSSO GRÁFICO APRESENTÁVEL

Podemos fazer inúmeros ajustes em nosso gráfico que aprendemos a plotar na Aula 02. Colocar título, legenda, nomear os eixos x e y, adicionar uma grade, escolher o tipo de traço, as cores envolvidas, enfim! Há uma variedade de modificações que a biblioteca Matplotlib nos permite fazer. Vamos aprender por partes?

Título

Para nomear o nosso gráfico, basta falarmos ao pyplot que queremos adicionar um titulo (title):

```
>>> plt.title("Rendimento mensal de uma loja de roupas durante o ano")
```

Nomear os eixos

Para escrever os significados dos eixos do nosso gráfico, pedimos ao pyplot para rotular os dados, seja do eixo x (xlabel), seja do eixo y (ylabel):

```
>>> plt.xlabel("Meses")
>>> plt.ylabel("Rendimento (R$)")
```

Colocar grade de fundo

Para adicionar uma grade (grid) ao fundo do gráfico, basta dizermos ao pyplot para adicioná-la:

```
>>> plt.grid()
```

Definir cor, espessura e estilo da linha

Lembra do plot? É nele que definimos parâmetros que formarão a linha do nosso gráfico. Antes de eu mostrar a você como devemos escrever esses parâmetros é valido ressaltar que existe uma tabela de códigos para cor e estilo de linha de um gráfico que a biblioteca Matplotlib reconhece. Veja a seguir:

Perceba que existem algumas cores básicas que podem ser identificadas apenas com uma letra. No nosso exemplo definiremos a linha na cor preta, por isso o código de cor é a letra **k**. Além de identificar as cores desse modo, você também pode utilizar o código hexadecimal da cor que desejar! Por exemplo, a cor preta em hexadecimal é representada pelo código #000000.

Já para os estilos de linha do nosso gráfico, o Matplotlib reconhece de acordo com a tabela ao lado.

Tipo da linha	Caractere
Linha cheia	'–'
Linha tracejada	'– –'
Linha traço-ponto	'– .'
Linha pontilhada	':'

Se quiser definir a espessura da linha do gráfico, basta adicionar um parâmetro numérico no plot do pyplot. Vamos voltar ao nosso gráfico!

Como eu digo ao pyplot que eu quero o gráfico com uma linha cheia (linestyle) preta (color) de espessura 3 (linewidth)? Veja abaixo!

```
>>> plt.plot(meses, receitas, color = "k", linestyle =  "-", linewidth = 3)
```

Pronto! Estabelecendo esses parâmetros no plot() você pode brincar e muito com a forma que você quer apresentar o seu gráfico.

Adicionar marcador, estilo e legenda

Para que os seus pontos (x,y) sejam representados no gráfico, podemos definir alguns parâmetros no pyplot por meio do método **scatter()**. Esses parâmetros podem ser a cor do marcador (color), o siginifcado do marcador por meio de legenda (label), o estilo do marcador (marker) e o peso (s) dele no gráfico, que seria semelhante a sua proporção dada por um numero inteiro. O estilo do marcador segue alguns modelos reconhecidos pelo Matplotlib. Veja alguns deles:

Tipo do Marcador	Caractere
Ponto	'.'
Pixel	','
Círculo	'o'
Triângulo	'v'
Triângulo	'^'
Triângulo	'<'
Triângulo	'>'
Y para baixo	'1'
Y para cima	'2'

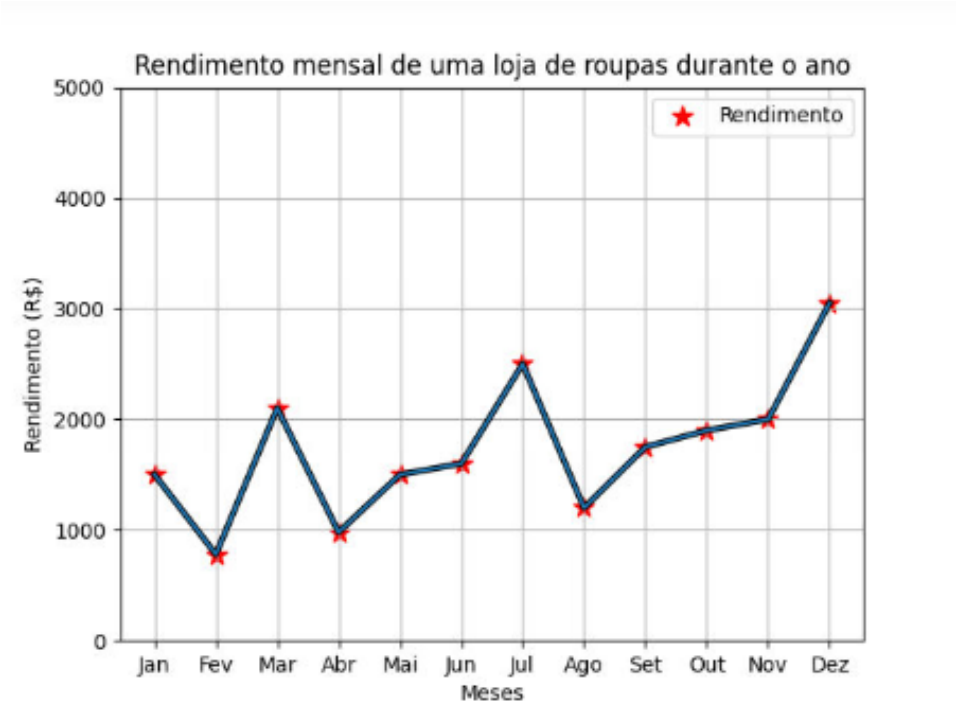
Tipo do Marcador	Caractere
Y para a esquerda	'3'
Y para a direita	'4'
Quadrado	's'
Pentágono	'p'
Estrela	'*'
Hexágono	'h'
Hexágono	'H'
Sinal de mais	'+'
Sinal cruzado	'x'
Losango largo	'D'
Losango estreito	'd'
Linha vertical	'|'
Linha horizontal	'_'

Para garantir que a legenda do seu marcador seja plotada junto ao gráfico, precisamos utilizar o método **legend()**, pois sem ele apenas armazenamos na memória o significado do marcador, porém sem plotar. É semelhante ao método show() discutido anteriormente na Aula 02.

Como aplicamos tudo isso em nosso gráfico? Vamos definir o nosso marcador como "Rendimento" na legenda, a cor como vermelho ('r'), o estilo como uma estrela ('*') e o peso dele como 100. É bem simples, veja:

```
>>> plt.scatter(meses, receitas, label = "Rendimento", color = "r", marker = "*", s = 100)
>>> plt.legend()
```

Não há mais desculpas para deixar o gráfico de forma não apresentável, viu?! Após aplicar todos esses ajustes o nosso gráfico está pronto para uma boa apresentação!



Ainda tem dúvidas? Veja o vídeo da Aula 03 disponível abaixo! Na Aula 04 te ensino como salvar esse gráfico em alguns formatos interessantes, afinal de contas se ele foi gerado é para ser usado!

ESCANEIE O CÓDIGO PARA TER ACESSO AO VÍDEO DA AULA 03 OU **CLIQUE AQUI!**

EXERCITE!

Escreva um programa para gerar um gráfico de linhas com as temperaturas máximas e mínimas em Florianópolis nos primeiros sete dias de setembro deste ano. Atenção às exigências impostas a seguir e use os dados da tabela abaixo,

Data	Max/Min
sex 01/09	23°/14°
sáb 02/09	25°/14°
dom 03/09	27°/13°
seg 04/09	28°/13°
ter 05/09	28°/12°
qua 06/09	27°/15°
qui 07/09	28°/14°

Exigido:

a) O gráfico deverá contar com as evoluções de temperaturas máxima e mínima em linhas separadas, porém em um único gráfico.
b) O gráfico deve ter o título "Temperaturas Máximas e Mínimas em Florianópolis ao longo da semana".
c) Os eixos deverão estar devidamente nomeados, explicitando a unidade de medida caso seja necessário.
d) A linhas do gráfico devem ser cheias e com um marcado para temperaturas máximas e outro diferente para temperaturas mínimas.
e) Use cores para as linhas e os marcadores, diferenciando-as.
f) Use uma legenda para diferenciar os marcadores de máxima e mínima temperaturas.
g) Adicione uma grade ao fundo do seu gráfico.
h) Limite os valores nos eixos caso julgue necessário para uma melhor visualização dos dados.

AULA 04

SALVANDO O NOSSO GRÁFICO

Uma vez que temos o nosso gráfico pronto, o ideal é que salvemos para utilizá-lo onde quisermos. Como sabemos, existem vários formatos de arquivos e quando se trata no formato de saída do nosso arquivo com o gráfico não seria diferente. Porém é bem simples, veja!

```
>>> plt.savefig('nome_da_imagem.png')
```

Caso deseje em pdf ou até mesmo jpeg, basta mudar o 'png' do código para o formato desejado! Simples não é?

Inclusive, caso quisesse um png transparente bastava definir isso em seu código, da seguinte forma:

```
>>> plt.savefig('nome_da_imagem.png', transparent = True)
```

Normalmente com esse comando o arquivo será salvo na mesma pasta do arquivo do código, portanto basta procurá-lo lá após executar o código.

Pronto! Espero que tenham aprendido muito até aqui! Confira abaixo o vídeo da a Aula 04 caso deseje e não esqueça de exercitar!

ESCANEIE O CÓDIGO PARA TER ACESSO AO VÍDEO DA AULA 04 OU **CLIQUE AQUI**!

EXERCITE!

Salve em seu computador o gráfico do problem anterior das Temperaturas Máximas e Mínimas em Florianópolis nos formatos png, png transparente, jpeg e pdf.

REFERÊNCIAS

1. **Visualização de gráficos 2D usando matplotlib.** Material disponibilizado pela Universidade São Paulo. (https://panda.ime.usp.br/algoritmos/static/algoritmos/10-matplotlib.html). Último acesso em 09/01/2021

2. **Matplotlib uma biblioteca Python para gerar gráficos interessantes.** Material disponibilizado pela Alura. (https://www.alura.com.br/artigos/criando-graficos-no-python-com-a-matplotlib). Último acesso em 09/01/2021

3. **Matplotlib e Scipy.** Material de apoio da UFRJ elaborado pelos professores João C. P. da Silva e Carla A. D. M. Delgado (https://dcc.ufrj.br/~pythonufrj/aulas-python2_37/aula8_teorica.pdf). Último acesso em 09/01/2021

4. **Introdução aos gráficos utilizando Python.** Material do Prof. Donald Mark Santee (https://www.comp.uems.br/~ojacques/NumericPython/introducaoAosGraficos.pdf). Último acesso em 09/01/2021

CCET | ufs
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

ATIVIDADE LIVRE - TEMA 08

EVANDRO BATISTA CIQUEIRA FILHO

PROGRAMAÇÃO IMPERATIVA