1)

Dentre as diversas bibliotecas disponíveis no repositório \_\_\_\_\_\_\_\_\_, pandas é um pacote Python que fornece estruturas de dados projetadas para facilitar o trabalho com dados \_\_\_\_\_\_\_\_\_\_\_ (tabelas) e de séries temporais (https://pandas.pydata.org/docs/getting\_started/overview.html). Esse pacote começou a ser desenvolvido em 2008, tornando-se uma solução open source no final de 2009. Desde 2015, o projeto \_\_\_\_\_\_\_ é patrocinado pela NumFOCUS.

 Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

PyPI; estruturados; pandas

Alternativa assinalada

* b)

Git; estruturados; pandas

* c)

PyPI; não estruturados estruturados; PyPI

* d)

Git; não estruturados estruturados; PyPI

* e)

PyPI; estruturados; Python

2)

Uma biblioteca criada para trabalhar com dados precisa de \_\_\_\_\_\_\_\_\_\_ que suportem o armazenamento e manipulação. A linguagem Python possui algumas estruturas que já fazem parte do seu interpretador padrão, como por exemplo as \_\_\_\_\_\_\_\_\_, capazes de armazenar uma sequência de valores e aumentar ou diminuir sua capacidade de acordo com a necessidade, ou então os \_\_\_\_\_\_\_\_\_\_, que são estruturas que guardam os dados no formato chave-valor.

 Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

estruturas de dados; tuplas; dicionários

* b)

métodos; listas; json

* c)

estruturas de dados; listas; json

* d)

estruturas de dados; listas; dicionários

Alternativa assinalada

* e)

métodos; listas; dicionários

3)

Para utilizar a biblioteca pandas é preciso fazer a instalação, usando o comando: pip install \_\_\_\_\_\_\_\_. Como uma ferramenta de alto nível, pandas possui duas estruturas de dados que são as principais para a análise/manipulação de dados: a Series e o DataFrame. Uma Series é um como \_\_\_\_\_\_\_\_\_\_\_\_\_\_, capaz de armazenar diferentes tipos de dados. Um DataFrame é \_\_\_\_\_\_\_\_\_\_\_, ou como a documentação apresenta, um contêiner para Series.

Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

PyPI; uma matriz; um vetor.

* b)

pandas; uma matriz; um vetor.

* c)

PyPI; um vetor de dados;  conjunto de Series

* d)

pandas; um vetor de dados; uma coluna

* e)

pandas; um vetor de dados;  conjunto de Series

Alternativa assinalada

4)

A biblioteca pandas possui duas estruturas de dados que são as principais. Ambas estruturas, possuem como grande característica, a indexação das linhas, ou seja, cada linha possui \_\_\_\_\_\_\_\_ que o identifica, o qual pode ser uma string, \_\_\_\_\_\_\_\_, um decimal ou uma data. Um DataFrame pode ter uma ou mais colunas e além \_\_\_\_\_\_\_\_\_, também há um rótulo de identificação com o nome da coluna. Podemos comparar um DataFrame como uma \_\_\_\_\_\_\_\_\_\_\_\_\_\_.

Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

um valor; uma lista; das colunas; matriz.

* b)

um rótulo; uma lista; dos índices; planilha eletrônica

* c)

um rótulo; um inteiro; dos índices; planilha eletrônica

* d)

um valor; um inteiro; das colunas; planilha eletrônica.

* e)

um rótulo; um inteiro; dos índices; matriz

Alternativa assinalada

5)

Para construir um objeto do tipo DataFrame, precisamos utilizar o método \_\_\_\_\_\_\_\_\_ do pacote pandas. Dentre todos os parâmetros esperados, somente um é obrigatório para se criar um DataFrame com dados, o parâmetro \_\_\_\_\_\_\_\_. Esse parâmetro pode receber, um objeto iterável, como uma lista, tupla, \_\_\_\_\_\_\_\_ ou um DataFrame.

Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

DataFrame();  data; um dicionário

Alternativa assinalada

* b)

construct(); data; string

* c)

DataFrame();  columns; string

* d)

construct(); columns; um dicionário

* e)

DataFrame();  columns; um dicionário

6)

A biblioteca pandas foi desenvolvida para trabalhar com dados \_\_\_\_\_\_\_\_\_\_\_\_\_, ou seja, dados dispostos em linhas e colunas. Os dados podem estar gravados em arquivos, em páginas web, em APIs, em outros softwares, em \_\_\_\_\_\_\_\_ (sistemas de armazenamento em cloud) ou em bancos de dados. Para todas essas origens (e até mais), a biblioteca possui \_\_\_\_\_\_\_\_\_\_ capazes de fazer a leitura dos dados e carregar \_\_\_\_\_\_\_\_\_\_\_\_.

Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

não estruturados; object stores; métodos; em um DataFrame

* b)

estruturados; SGBD; funções; em um dicionário

* c)

não estruturados; SGBD; funções; em um dicionário

* d)

estruturados; object stores; métodos; em um DataFrame

Alternativa assinalada

* e)

estruturados; object stores; métodos; em um dicionário

7)

Todos os métodos capazes de fazer a leitura dos dados estruturados possuem prefixo \_\_\_\_\_\_, onde pd é o apelido dado no momento da \_\_\_\_\_\_\_\_ da biblioteca e XXX é o restante da sintaxe do método. Além de fazer a leitura a biblioteca possui diversos métodos capazes de \_\_\_\_\_\_\_\_\_ o DataFrame em um arquivo, em um banco ou ainda simplesmente copiar para a área de transferência do sistema operacional.

Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

pandas.pd.read\_XXX; importação; escrever

* b)

pd.read\_XXX; importação; escrever

Alternativa assinalada

* c)

pd.read\_XXX; instalação; transformar

* d)

pandas.pd.read\_XXX; instalação; escrever

* e)

pd.read\_XXX; importação; transformar

8)

**CSV**  (valores separados por \_\_\_\_\_\_\_\_) é um formato de arquivo, nos quais os dados são separados por \_\_\_\_\_\_\_\_\_\_\_\_. Originalmente, esse delimitador é \_\_\_\_\_\_\_\_\_\_, mas na prática um arquivo CSV pode ser criado com qualquer delimitador, por exemplo, por ponto e vírgula (;), por pipe (|), dentre outros. Por ser um arquivo \_\_\_\_\_\_\_\_, é fácil de ser lido em qualquer sistema, por isso se tornou tão democrático.

Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

delimitador; vírgula; tab; chave/valor

* b)

vírgula; vírgula; tab; de texto

* c)

vírgula; um delimitador; uma vírgula; de texto

Alternativa assinalada

* d)

delimitador; tab; uma vírgula; chave/valor

* e)

vírgula; um delimitador; uma vírgula; chave/valor

9)

A leitura de um arquivo JSON deve ser feita com o método \_\_\_\_\_\_\_\_\_\_\_ .  O único parâmetro que é obrigatório para se carregar os dados é o \_\_\_\_\_\_\_, no qual deve ser passado um caminho para o arquivo ou um \_\_\_\_\_\_\_\_\_\_\_\_\_.

Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

pandas.read(); path\_or\_buf; dicionário

* b)

pandas.read\_json(); path\_or\_buf; dicionário

* c)

pandas.read(); data; arquivo como objeto

* d)

pandas.read\_json(); data; arquivo como objeto

* e)

pandas.read\_json(); path\_or\_buf; arquivo como objeto

Alternativa assinalada

10)

A leitura de um arquivo CSV deve ser feita com o método: pandas.read\_csv(parâmetros). Tal método possui um grande número de parâmetros, o que proporciona uma versatilidade incrível para esse método. Dentre todos os parâmetros o único que é obrigatório é o \_\_\_\_\_\_\_\_\_\_\_\_. Outro parâmetro que é bastante importante para a leitura desse arquivo é o \_\_\_\_\_\_\_\_ ou delimiter (ambos fazem a mesma coisa), que por padrão possui o valor \_\_\_\_\_\_\_\_\_, ou seja, caso não seja especificado nenhum valor, então o método fará a leitura dos dados considerando esse valor default. O parâmetro \_\_\_\_\_\_\_\_, tem como valor padrão 'infer', que significa que o método fará a inferência para os nomes das colunas a partir da primeira linha de dados do arquivo.

Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

data; separator; pipe (|); columns

* b)

filepath\_or\_buffer; sep; vírgula (,); columns

* c)

data; sep; vírgula (,); header

* d)

filepath\_or\_buffer; sep; vírgula (,); header

Alternativa assinalada

* e)

filepath\_or\_buffer; separator; vírgula (,); columns

11)

Visualização de dados ou \_\_\_\_\_\_\_\_ é um dos pilares dos profissionais que trabalham com dados. Existem profissionais que se especializam e inclusive tiram certificações para atuar nessa área específica. Após uma análise de dados e extração de informações é interessante que a entrega de resultados para a área de negócios seja feita de maneira visual, ou seja, por meio de gráficos. Um gráfico bem elaborado "fala" por si só e ajuda aos que assistem a entenderem os resultados.

A linguagem Python, conta com uma série de bibliotecas que permitem a criação de gráficos, os quais podem ser \_\_\_\_\_\_\_  ou \_\_\_\_\_\_\_, que apresentam iteração, como por exemplo, responder a \_\_\_\_\_\_\_\_ de clique do mouse.

Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

DataGraph; estáticos; dinâmicos; eventos

* b)

DataViz; funções; métodos; iteração

* c)

DataGraph; funções; métodos; iteração

* d)

DataViz; estáticos; funcionais; iteração

* e)

DataViz; estáticos; dinâmicos; eventos

Alternativa assinalada

12)

Com a utilização da linguagem Python na área científica para trabalhar com dados, após a extração dos resultados, o cientista precisava criar seus gráficos nos outros softwares mencionados, o que se tornava inconveniente, motivando a criação de uma das principais bibliotecas em Python.

Ao se falar em criação de gráficos em Python, o profissional precisa conhecer a biblioteca matplotlib, pois

**Alternativas:**

* a)

diversas outras são construídas a partir desta.

Alternativa assinalada

* b)

é a que contém o maior número de funções.

* c)

é a única que contém função capaz de plotar gráfico de barras.

* d)

é a única que contém função capaz de plotar gráficos estatísticos.

* e)

é a única que já vem instalada com o interpretador padrão.

13)

As principais estruturas de dados da biblioteca pandas \_\_\_\_\_\_\_\_\_\_\_ e \_\_\_\_\_\_\_\_\_\_\_ possuem o método \_\_\_\_\_\_\_\_\_\_, construído com base no \_\_\_\_\_\_\_ e que permite criar gráficos a partir dos dados nas estruturas.

Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

Series; DataFrame; plot(); matplotlib

Alternativa assinalada

* b)

listas; dicionários; plot(); matplotlib

* c)

Series; DataFrame; plotar(); PyPI

* d)

listas; dicionários; plotar(); PyPI

* e)

Series; DataFrame; plotar(); interpretador

14)

A partir de um DataFrame, podemos invocar o método: \_\_\_\_\_\_\_\_\_ para criar os gráficos. Os argumentos dessa função, podem variar, mas existem três que são triviais: o nome da coluna que será atribuída  \_\_\_\_\_\_\_\_\_, o nome da coluna que será atribuído  \_\_\_\_\_\_\_\_\_\_, bem como \_\_\_\_\_\_\_\_\_\_.

Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

df.plotar(\*args, \*\*kwargs); ao rótulo; a legenda; o tamanho do gráfico.

* b)

df.plot(\*args, \*\*kwargs); ao rótulo; a legenda; o tamanho do gráfico.

* c)

df.plotar(\*args, \*\*kwargs); eixo x; eixo y; o tipo de gráfico

* d)

df.plot(\*args, \*\*kwargs); eixo x; eixo y; o tamanho do gráfico

* e)

df.plot(\*args, \*\*kwargs); eixo x; eixo y; o tipo de gráfico

Alternativa assinalada

15)

Seaborn é outra biblioteca Python baseada \_\_\_\_\_\_\_\_\_\_\_, que foi desenvolvida especificamente para criação de gráficos. Seaborn pode ser instalado via pip install: pip install \_\_\_\_\_\_\_\_\_, e para utilizar no projeto existe uma convenção para sintaxe: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. A biblioteca conta com um repositório de datasets que podem ser usados para explorar as funcionalidades. A função load\_dataset(), pode ser usada para carregar \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

 Escolha a opção que completa corretamente as lacunas.

**Alternativas:**

* a)

em pandas; python\_seaborn; import seaborn as sbn; uma Series.

* b)

matplotlib; seaborn; import seaborn as sns; um DataFrame

Alternativa assinalada

* c)

matplotlib; seaborn; import seaborn as sns; uma Series.

* d)

em pandas; seaborn; import seaborn as sns; um DataFrame

* e)

matplotlib; pandas\_seaborn; import seaborn as sns; um DataFrame