Atividade Python

Histórico

A linguagem Python foi criada por Guido van Rossum em meados dos anos 90. A motivação para criar a linguagem foi a necessidade de uma linguagem de programação fácil de ler e escrever, com sintaxe clara e intuitiva. O nome "Python" foi inspirado no grupo de comédia britânico Monty Python, que van Rossum era um grande fã.

O desenvolvimento do Python começou em 1989, quando van Rossum trabalhava no Centrum Voor Wiskunde en Informatica (CWI) em Amsterdã, Holanda. Ele começou a procurar por uma linguagem de script que tivesse sintaxe semelhante ao ABC, mas que tivesse acesso às chamadas de sistema do Amoeba. Após procurar e não encontrar nenhuma linguagem que atendesse às suas necessidades, Rossum decidiu projetar uma linguagem de script simples que pudesse superar as inadequações do ABC.

A primeira versão do Python foi lançada em 1991, e desde então a linguagem tem evoluído e se tornado uma das linguagens de programação mais populares e amplamente utilizadas no mundo.

Quais são os principais IDEs?

PyCharm: Lançado em 2010 pela JetBrains, é uma das IDEs mais utilizadas por quem programa em Python. Oferece análise de código, suporte ao desenvolvimento web com Django, depurador gráfico e recursos de desenvolvimento remoto.

Eclipse: Lançada em 2001 pela IBM, é uma IDE multiplataforma que suporta diversas linguagens, incluindo Python. Oferece tecnologia baseada em plugins e pacotes de desenvolvimento para facilitar a codificação de aplicações Python.

Jupyter Notebook: Criada em 2014, é uma ferramenta de ciência de dados que permite a manipulação de documentos e suporta diversas linguagens, incluindo Python. Oferece ambiente interativo de ciência de dados e é uma opção excelente para quem inicia na ciência de dados.

Sua aplicação:

Inteligência Artificial: Python é amplamente utilizado em machine learning e aprendizado de máquina para automatizar tarefas e analisar grandes conjuntos de dados.

Ciência de Dados: Python é usado para análise de dados, visualização de dados e manipulação de grandes conjuntos de dados.

Desenvolvimento de Web: Python é utilizado para desenvolver aplicações web com frameworks como Django e Flask.

Desenvolvimento de Aplicativos: Python é usado para criar aplicativos desktop e mobile.

Automação de Scripts: Python é utilizado para automatizar operações do sistema de arquivos no Windows e gerenciar rotinas de backupt.

Fintechs: Python é aplicado em tecnologia financeira, como blockchain, criptomoedas e bitcoins.

Georreferenciamento: Python é usado para tratar dados geográficos com bibliotecas como ArcMap e ArcGIS.

Oque são bibliotecas? Como são utilizadas?

De acordo com a linguagem Python, bibliotecas são conjuntos de módulos e funções úteis que reduzem o uso de código no programa. São mais de 137 mil bibliotecas Python que facilitam a programação dos desenvolvedores, com diversas finalidades. As bibliotecas Python são usadas para automatizar tarefas, manipular e processar dados, criar gráficos e visualizações, desenvolver aplicações web e muito mais.

Quais os principais Framework?

Frameworks Python, como Django e Flask, fornecem uma estrutura e ferramentas para facilitar o desenvolvimento web.

Questões:

1.2: Porque o Python é melhor que o Java?

O Python é considerado melhor que o Java em muitos casos devido às seguintes razões:

Sintaxe mais simples: A sintaxe do Python é mais concisa e fácil de ler, tornando mais rápido o desenvolvimento de aplicações.

Flexibilidade: Python é uma linguagem multiplataforma, suporta orientação a objetos e é fácil de aprender.

Desenvolvimento mais rápido: Python é uma linguagem de alto nível, com sintaxe clara e concisa, permitindo um desenvolvimento mais rápido e eficiente.

Comunidade mais ativa: A comunidade de desenvolvedores Python é mais ativa e colaborativa, com uma ampla variedade de bibliotecas e ferramentas disponíveis.

Desenvolvimento de aplicações web: Python é mais fácil e rápido para desenvolver aplicações web com frameworks como Django e Flask.

Desenvolvimento de aplicações desktop: Python é mais fácil e rápido para desenvolver aplicações desktop com bibliotecas como Tkinter e PyQt.

1.3: Quantos tipos de dados existem no Python?

Existem 6 principais tipos de dados em Python:

Números (int, float, complex)

Strings (str)

Listas (list)

Tuplas (tuple)

Dicionários (dict)

Conjuntos (set)

1.4: Qual é a diferença entre uma “tupla” e uma “lista”?

Listas são mutáveis, ou seja, seus elementos podem ser modificados após a criação.

Tuplas são imutáveis, ou seja, seus elementos não podem ser alterados após a criação.

1.5: Oque é “decapagem” e “disinteressante”?

Decapagem (duck typing): é a capacidade de uma linguagem de programação verificar se um objeto tem um determinado método ou atributo, sem se preocupar com a sua origem ou tipo. Isso permite que objetos de diferentes tipos sejam tratados de forma semelhante.

Desinteressante (uninteresting): é um termo utilizado por Guido van Rossum, o criador do Python, para descrever a falta de interesse em uma linguagem de programação em determinados detalhes técnicos.

1.6: Oque é “lambda”?

Em Python, uma "lambda" é uma função anônima, ou seja, uma função sem nome. Ela é utilizada para definir funções breves e simples, que são utilizadas em lugares específicos do código. A lambda é representada por uma expressão que começa com a palavra-chave lambda, seguida de parênteses que contêm os argumentos da função e uma expressão que define o corpo da função.

1.7: Como a memória é gerenciada no Python?

No Python, a memória é gerenciada automaticamente pelo sistema de gerenciamento de memória, conhecido como "Garbage Collector". Isso significa que o desenvolvedor não precisa se preocupar com a alocação e liberação manual de memória, como é necessário em outras linguagens de programação, como C ou C++.

O Garbage Collector do Python monitora os objetos em uso e libera automaticamente a memória ocupada por objetos que não são mais necessários. Isso é feito através de técnicas como contagem de referências e coleta de lixo, que identificam e removem objetos que não são mais acessíveis pelo programa.

Essa abordagem torna o desenvolvimento em Python mais simples e menos propenso a erros de gerenciamento de memória, permitindo que os desenvolvedores se concentrem mais na lógica do programa do que em detalhes de baixo nível.

1.8: Oque é “passar”?

Em Python, "passar" é uma palavra-chave utilizada para indicar que uma instrução é necessária para que o compilador não lance um erro de sintaxe, mas não tem efeito real no código. É comumente utilizada em estruturas de controle, como loops e condicionais, quando não há nenhuma ação específica a ser executada.

1.9: Você pode copiar um objeto em Python?

Sim, é possível copiar objetos em Python. Existem diferentes maneiras de fazer isso, dependendo do tipo de objeto que você deseja copiar.

Cópia rasa (shallow copy): Você pode usar o método copy() para criar uma cópia rasa do objeto. Nesse caso, a nova cópia aponta para os mesmos objetos internos que a original.

Cópia profunda (deep copy): Você pode usar o método deepcopy() para criar uma cópia profunda do objeto. Nesse caso, a nova cópia cria novos objetos internos, independentes da original.

Fatiamento (slicing): Para objetos mutáveis, como listas, você pode usar o fatiamento para criar uma cópia.

1.10: Como deletar um arquivo dentro do Python?

Para deletar um arquivo dentro do Python, você pode usar a função os.remove() do módulo os. Essa função remove o arquivo especificado e retorna None se o arquivo for encontrado e deletado com sucesso.

1.11: O que é um 'dicionário?

Em Python, um dicionário é uma estrutura de dados que armazena pares de chave-valor. É uma coleção de pares de chave-valor, onde cada chave é única e é associada a um valor específico.

1.12: Python é uma linguagem interpretada?

Sim, Python é uma linguagem interpretada. Isso significa que o código Python é executado linha por linha, sem a necessidade de compilação prévia. O intérprete Python lê o código, interpreta as instruções e executa as ações correspondentes.

1.14: Como o Python é orientado a objeto?

O Python é uma linguagem orientada a objetos (OOP), que significa que ele é projetado para organizar e estruturar o código em torno de objetos e classes. Isso permite que os desenvolvedores criem programas mais organizados, escaláveis e fáceis de manter.

Em Python, as classes são definidas com a palavra-chave class e são usadas para criar objetos que têm atributos (ou propriedades) e métodos (ou comportamentos). Os objetos são instâncias das classes e podem ser usados para representar entidades do mundo real, como pessoas, carros, animais, etc.

1.15: O que é 'fatiar'?

Em Python, "fatiar" é um termo que se refere à técnica de dividir uma lista em sub-listas, conhecida como slicing. Isso permite que você obtenha uma sub-lista específica de uma lista original.