**Relatório Python**

**História**

**Python foi criado por Guido van Rossum e a primeira versão foi lançada em 1991. A motivação para a criação de Python surgiu durante o final dos anos 1980, quando Guido estava trabalhando no CWI (Centrum Wiskunde & Informatica) em Amsterdã. Ele estava envolvido no projeto ABC, uma linguagem de programação que, embora inovadora, tinha várias limitações e não ganhou muita popularidade.**

**Observando essas limitações, Guido decidiu criar uma nova linguagem que incorporasse as melhores ideias de ABC, mas que fosse mais prática e fácil de usar. Ele desejava criar uma linguagem com uma sintaxe clara e intuitiva que pudesse melhorar a legibilidade do código, permitindo que os programadores fossem mais produtivos. Assim nasceu Python, uma linguagem projetada para ser fácil de aprender e usar, ao mesmo tempo em que era poderosa e flexível o suficiente para ser usada em uma ampla variedade de aplicações.**

**Desde o seu lançamento inicial, Python tem crescido significativamente em popularidade e se tornou uma das linguagens de programação mais utilizadas no mundo. Seu design simples e legibilidade, combinados com uma vasta coleção de bibliotecas e frameworks, fizeram de Python uma escolha popular tanto para iniciantes quanto para desenvolvedores experientes. A comunidade ativa de desenvolvedores que contribui para o ecossistema Python também desempenha um papel crucial em sua evolução contínua e sucesso global.**

# **Quais são os principais IDEs?**

* **PyCharm: Uma das IDEs mais populares para Python, desenvolvida pela JetBrains, oferece recursos avançados como depuração, teste, análise de código e suporte a frameworks.**
* **Visual Studio Code (VS Code): Um editor de código-fonte gratuito e de código aberto desenvolvido pela Microsoft, amplamente utilizado para Python com extensões como Python Extension Pack.**
* **Jupyter Notebook: Uma aplicação web que permite a criação e compartilhamento de documentos que contêm código executável, visualizações e texto narrativo.**
* **Spyder: Uma IDE poderosa para cientistas de dados, que inclui um console interativo, explorador de variáveis, e um depurador.**

# **Sua aplicação (Onde é utilizada?)**

**Python é amplamente utilizado em várias áreas:**

* **Desenvolvimento Web: Frameworks como Django e Flask.**
* **Ciência de Dados e Aprendizado de Máquina: Bibliotecas como Pandas, NumPy, SciPy, TensorFlow, e scikit-learn.**
* **Automação de Tarefas: Scripts para automatizar tarefas administrativas e de sistemas.**
* **Desenvolvimento de Software: Ferramentas e aplicações diversas.**
* **Educação: Muito popular no ensino de programação devido à sua sintaxe simples e clara.**

# **O que são bibliotecas?**

**Definição: Bibliotecas são coleções de módulos ou pacotes que contêm funções, classes e rotinas reutilizáveis que facilitam o desenvolvimento de software ao fornecer funcionalidades prontas para uso.**

**Como são utilizadas?: Elas são importadas e utilizadas dentro dos projetos para realizar tarefas específicas.**

# **Quais os principais frameworks? (Exemplos)**

* **Django: Um framework de alto nível para o desenvolvimento de aplicações web em Python, que promove o desenvolvimento rápido e o design limpo e pragmático.**

**Exemplo: Criação de sites complexos com autenticação de usuários, formulários dinâmicos e um painel administrativo.**

* **Flask: Um micro framework web para Python, que fornece as ferramentas básicas para construir aplicações web com flexibilidade e simplicidade.**

**Exemplo: Desenvolvimento de APIs RESTful e pequenos aplicativos web.**

* **TensorFlow: Uma biblioteca de código aberto para computação numérica e aprendizado de máquina, que permite construir e treinar modelos de aprendizado profundo.**

**Exemplo: Implementação de redes neurais convolucionais para reconhecimento de imagem.**

* **NumPy: Uma biblioteca fundamental para a computação científica com Python, que oferece suporte a arrays multidimensionais e funções matemáticas avançadas.**

**Exemplo: Processamento de grandes volumes de dados numéricos e operações de álgebra linear.**

Perguntas

# Por que o Python é melhor que o Java?

O Python (quando comparado com o Java) é mais fácil de usar e possui velocidades de codificação muito melhores. Além disso, quando se trata de dados, o Java é estaticamente digitado, enquanto o Python oferece digitação dinâmica. Isso é considerado um grande avanço. Mas só para ter certeza, prepare-se para as perguntas e respostas da entrevista em Python.

# Quantos tipos de dados existem no Python?

O Python possui cinco tipos de dados diferentes: string, lista, número, dicionário e tupla.

# Qual é a diferença entre uma 'tupla' e uma 'lista'?

A principal diferença é que as listas são mais lentas, mas podem ser editadas, enquanto as tuplas funcionam mais rápido, mas não podem ser modificadas.

# O que é 'decapagem' e 'desinteressante'?

A decapagem acontece quando um módulo dentro do Python é aceito e convertido em um módulo de string, e depois é despejado no arquivo. Ao contrário disso, o desengatamento é quando você recupera o módulo de string do arquivo.

# O que é 'lambda'?

Lambda é uma função anonimamente executada com apenas uma expressão única.

# Como a memória é gerenciada no Python?

O espaço de heap privado do Python é responsável pelo gerenciamento de memória. É acessível apenas por um intérprete - se você for um programador Python, não poderá alcançá-lo. A linguagem também possui uma recicladora embutida que é responsável por criar mais espaço de pilha livre (isso é feito através da reciclagem de memória não usada).

# O que é 'passar'?

Passe simplesmente indica um espaço que deve ser deixado em branco dentro da declaração composta.

# Você pode copiar um objeto em Python?

Sim, você pode usar o comando `copy.copy()`.

# Como deletar um arquivo dentro do Python?

Para deletar algo no Python, use o comando `os.remove(name\_of\_the\_file)`.

# O que é um 'dicionário'?

Dicionários são compostos de chaves e os valores correspondentes da chave. Aqui está um exemplo:

```python

dict = {'Car': 'Ford', 'Type': 'Mustang', 'Year': '1967'}

print(dict['Car']) # Ford

print(dict['Type']) # Mustang

print(dict['Year']) # 1967

```

# Python é uma linguagem interpretada?

Sim, o Python é uma linguagem de programação interpretada. O que isto significa? É um processo de três vias - você escreve o código-fonte, o Python o converte em uma linguagem intermediária (para facilitar o entendimento) e depois é novamente transformado em códigos de máquina que são então executados.

# Como o Python é orientado a objeto?

Linguagens de programação orientada a objetos (OOPs) são baseadas em classes e objetos dessas classes. Python é exatamente isso. Mais ainda, o Python possui os recursos que são creditados aos OOPs - herança, polimorfismo, etc.

# O que é 'fatiar'?

No Python, o fatiamento é quando você seleciona vários itens de lugares como listas, sequências de caracteres e assim por diante.