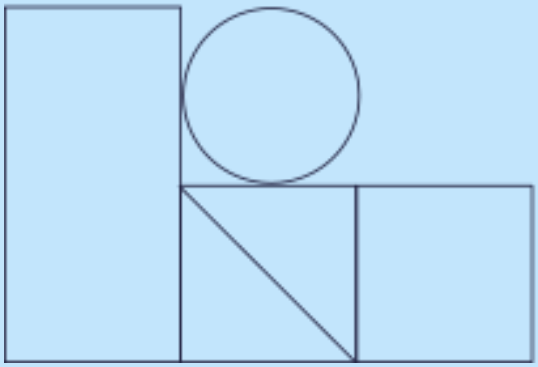
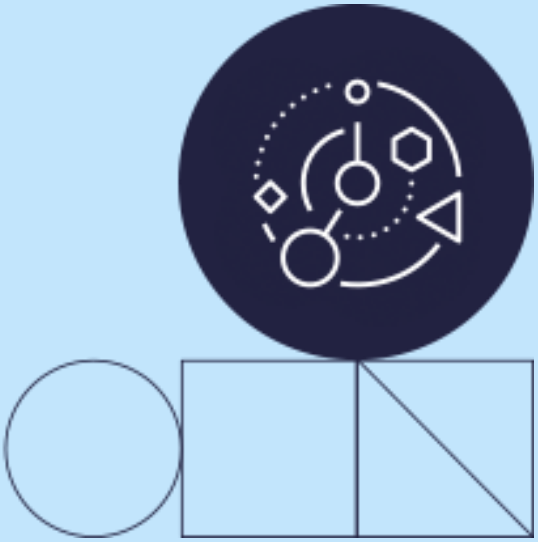
IBM SkillsBuild

IBM

Noções básicas de programação

Linguagem compilada *vs.* linguagem interpretada





Índice

Introdução

A linguagem sempre foi o meio de comunicação mais eficaz. A boa comunicação é sempre uma ponte da confusão para a clareza. Não apenas os humanos, cada elemento desta natureza também tem sua própria linguagem para compartilhar suas emoções e sentimentos com os outros. Às vezes até uma brisa suave nos diz uma centena de coisas.

Hoje, nesta 21ª era, a era da tecnologia, os computadores têm linguagens para se comunicar conosco, humanos... linguagem de programação.

A linguagem de programação pode ser classificada em dois tipos:

- Linguagem compilada
- Língua interpretada

Qual é a linguagem compilada?

Linguagens de programação que a máquina alvo pode ler sem a ajuda de outro programa. Geralmente, os humanos não conseguem entender o código desta linguagem.

Compilador

É um programa que toma como entrada um texto escrito em um idioma fonte, e como saída fornece outro texto em um idioma objeto. Sua função é traduzir o idioma fonte para o idioma objeto.

Um compilador não funciona isoladamente.

Ela precisa de outros programas para atingir seu objetivo: obter um programa executável de um programa fonte em uma linguagem de alto nível. Alguns destes programas são o pré-processador, o linker, o depurador e o assembler.

O pré-processador se encarrega de incluir arquivos, expandir macros, remover comentários e outras tarefas similares.

O linker constrói o arquivo executável adicionando ao arquivo objeto gerado pelo compilador os cabeçalhos necessários e as funções de biblioteca utilizadas pelo programa fonte.

funções de biblioteca utilizadas pelo programa fonte para o arquivo objeto gerado pelo compilador.

O depurador permite, se o compilador tiver gerado o programa objeto corretamente, acompanhar a execução de um programa passo a passo.

Muitos compiladores, ao invés de gerar código objeto, geram um programa de linguagem assembly que deve então ser convertido em um executável por um programa assembly.

executável por meio de um programa de montagem.

Vantagens

- Rápida execução.
- O código compilado não pode ser "aberto" por outros.
- Não há necessidade de transmitir o código fonte.
- O código compilado é compactado em um único arquivo.

Desvantagens

- O código compilado normalmente ocupa muito espaço em disco, pois incorpora algumas bibliotecas do sistema no próprio código.
- A depuração de um programa envolve a recompilação após as mudanças.

Qual é a linguagem interpretada?

Estas são linguagens de programação que a máquina alvo não consegue ler por si só. As máquinas sempre precisam de outro programa para a conversão das instruções em sua forma legível.

Basicamente, as palavras-chave de qualquer linguagem de programação devem ser convertidas para linguagem de máquina, para que o código possa ser executado e nos dê a saída através do sistema.

A metodologia da linguagem da máquina está sempre além da inteligência humana.

Portanto, é necessária a mediação de um intérprete, que, como vimos em lições anteriores, são programas de tradutores que transformam programas fonte escritos em linguagem de alto nível em programas objeto escritos em linguagem de máquina. Nesses programa de intérprete a tradução é feita de tal forma que, após transformar uma instrução do programa fonte em uma ou várias instruções em linguagem de máquina. instruções em uma ou mais instruções de linguagem da máquina. Eles não esperam que a próxima instrução seja traduzida, mas a executam imediatamente.

Vantagens

- Depuração rápida do código, já que não há necessidade de recompilar após uma mudança.
- Manutenção fácil e rápida do código fonte.

Desvantagens

- A execução é desacelerada, já que a interpretação linha por linha é necessária cada vez que o programa é executado.
- O usuário tem acesso ao código e pode copiá-lo ou modificá-lo.

Exemplos de linguagens de código interpretadas são: JavaScript, Python, BASIC, PHP, etc.

Software de tradução

O software de tradução converte as palavras-chave da linguagem de programação em linguagem de máquina e a executa para produzir o resultado.

Normalmente, o software do tradutor funciona de duas maneiras:

- Como um compilador
- Como um intérprete

Como um compilador, o que o tradutor de software faz é o seguinte:

Ele considera todas as instruções do Programa e converte o código fonte em código de máquina equivalente.

Por exemplo: C, C++, CLEO, C#... use um compilador para que estes se enquadrem na categoria de idiomas compilados.

No papel de intérprete, o software do tradutor toma uma instrução de cada vez e a executa em tempo real. Esta é na verdade uma forma interativa de executar um programa.

Por exemplo: JavaScript, Python, BASIC, etc., use um intérprete para ser incluído na categoria de linguagens interpretadas.

Esquematicamente, as diferenças entre os idiomas compilados e interpretados podem parecer como se segue:

Linguagem compilada *vs.* interpretada

Idiomas compilados	**Idiomas interpretados**
O compilador trabalha em todo o programa de uma só vez. Ele leva o programa inteiro como entrada.	Um intérprete leva os programas linha por linha. É necessário uma declaração de cada vez como entrada.
Ele gera o código intermediário conhecido como código de máquina ou código de objeto.	Um intérprete não gera um código intermediário conhecido como código de máquina.
Os idiomas dos compiladores são mais eficientes, mas difíceis de depurar.	Os idiomas interpretados são menos eficientes, mas fáceis de depurar. Esta especialidade a torna uma escolha ideal para novos alunos.
O compilador não permite a execução de um programa até que ele esteja completamente livre de erros.	O intérprete executa o programa desde a primeira linha e pára a execução somente se encontrar um erro.
Compilar uma vez e executar a qualquer momento. O programa compilado não precisa ser compilado a cada vez.	Cada vez que você executa o programa, ele é interpretado linha por linha.
Os erros são relatados depois que o programa inteiro é verificado quanto à sintáctica e outros erros.	O erro é relatado quando o primeiro erro é encontrado. O resto do programa permanece sem controle até que o erro seja resolvido.
O programa compilado ocupa mais memória porque todo código objeto tem que residir na memória.	O intérprete não gera código de memória intermediário. Portanto, os programas de intérpretes são eficientes em termos de memória.
Por exemplo: C, C++, CLEO, C#.	Por exemplo: JavaScript, Python, BASIC, etc.

Conclusão

No resultado do processo de interpretação ou compilação está a diferença entre a linguagem interpretada e compilada. Um intérprete sempre produz uma saída de programa; um compilador, por outro lado, produz um programa escrito em linguagem de montagem.

Cabe então ao montador da arquitetura converter o programa resultante em código binário. Para cada computador individual, dependendo de sua arquitetura, a linguagem de montagem varia. Consequentemente, somente em computadores que tenham a mesma arquitetura que o computador em que foram compilados, os programas compilados podem ser executados.

Em idiomas compilados não temos acesso ao código diretamente porque ele está em um arquivo e podemos detectar erros em tempo de compilação e de execução, enquanto em idiomas interpretados temos acesso ao código porque ele é traduzido em tempo real e podemos detectar erros em tempo de execução.