# Esquema Gráfico Comparativo Entre Linguagens Compiladas e Linguagens Interpretadas

| Linguagens Compiladas | Linguagens Interpretadas |
|---|---|
| Linguagens de um nível mais baixo, mais próximas da linguagem que o computador compreende e executa, por isso mais longe da linguagem corrente e da compreensão humana. | Linguagens de um nível mais elevado, mais próximas da linguagem corrente, e por isso de mais fácil compreensão para o programador. |
| Linguagens que necessitam de um software compilador para transformar o código fonte completo em código máquina. | Linguagens que, através de um software intérprete, reconhecem o código inserido linha a linha e que vão gerar, naquele momento, à medida que o código é corrido, as instruções para a CPU. |
| O passo de compilação é o que gera um executável, ou seja, existe um passo intermédio entre o código fonte e a execução. | Por não existir o passo intermédio de compilação nestas linguagens, o código é executado mais rapidamente, ou seja, o feedback é imediato. |
| O passo de compilação permite detetar erros no código, sendo que o compilador para a compilação, não gerando logo um executável, alerta o programador do erro, indicando também qual a linha em que este se encontra, para que o mesmo possa ser corrigido. | Como não existe um compilador nestas linguagens, se existir um erro no código, o mesmo para de ser executado, e o software intérprete revela que existe um erro, não indicando normalmente onde é que o mesmo se encontra. |
| Por serem linguagens de mais baixo nível, permitem um maior controlo sobre os recursos do computador, ou seja, permitem ter mais controlo sobre o uso do processador, da RAM e do disco do computador. | O código pode ser corrido imediatamente após ser escrito. |
| São de execução mais rápida. | Desenvolvimento mais rápido e facilitado. |
| São exemplos de linguagens compiladas C, C++ e Go. | São exemplos de linguagens interpretadas JavaScript, VisualScript e Python. |