|  |  |
| --- | --- |
| Interpreter translates just one statement of the program at a time into machine code . |  Compiler scans the entire program and translates the whole of it into machine code at once . |
|  An interpreter takes very less time to analyze the source code . However, the overall time to execute the process is much slower. |  A compiler takes a lot of time to analyze the source code. However, the overall time taken to execute the process is much faster. |
|  An interpreter does not generate an intermediary code. Hence, an interpreter is highly efficient in terms of its memory. |  A compiler always generates an intermediary object code. It will need further linking. Hence more memory is needed. |
|  Keeps translating the program continuously till the first error is confronted. If any error is spotted, it stops working and hence debugging becomes easy. |  A compiler generates the error message only after it scans the complete program and hence debugging is relatively harder while working with a compiler. |
|  Interpreters are used by programming languages like Ruby and Python for example. |  Compliers are used by programming languages like C and C++ for example. |