## COMPILER vs INTERPRETER vs ASSEMBLER:

Compiler: A compiler is a language translator that converts high level programs into machine understandable machine codes. In this process, the compiler converts the whole program to machine code at a time. If there are any syntactic or semantic error, the compiler will indicate them. It checks the whole program and displays all errors. It is not possible to execute the program without fixing those errors.

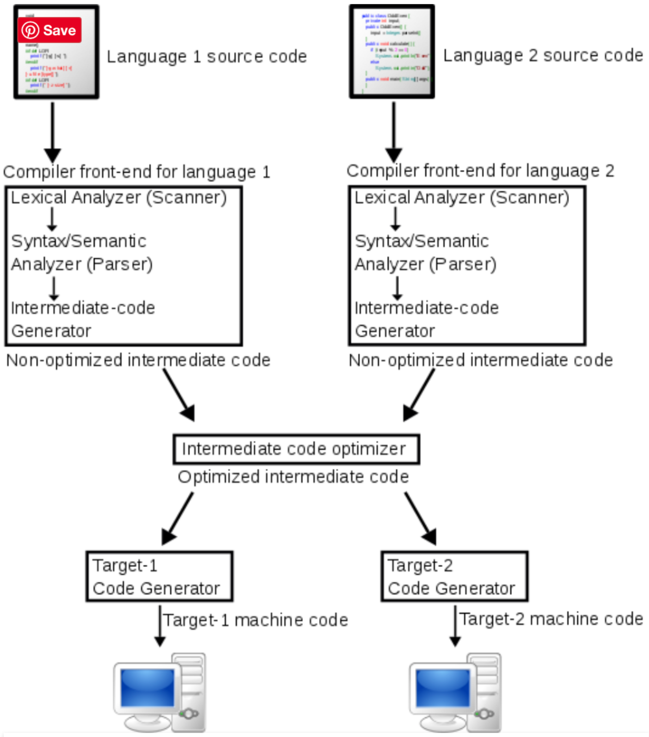

Save
Language 1 source code
Language 2 source code
Compiler front-end for language 1
Compiler front-end for language 2
Lexical Analyzer (Scanner)
Syntax/Semantic Analyzer (Parser)
Intermediate-code Generator
Lexical Analyzer (Scanner)
Syntax/Semantic Analyzer (Parser)
Intermediate-code Generator
Non-optimized intermediate code
Non-optimized intermediate code
Intermediate code optimizer
Optimized intermediate code
Target-1 Code Generator
Target-2 Code Generator
Target-1 machine code
Target-2 machine code

*Figure 1: Compiler*

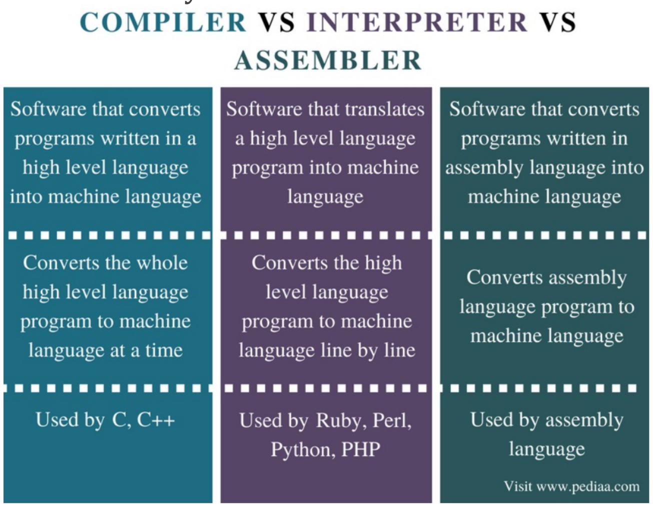
Interpreter: An interpreter is also a language translator that converts high level programs into machine codes. Unlike compilers, interpreters convert the source code to machine code line by line. As it checks line by line, the scanning time is lower. But the overall execution time is higher.

Interpreter displays an error at a time. The programmer should fix that error to interpret the next line. Programming languages such as Python, Ruby, PHP, Perl are some examples of interpreter-based languages.

Assembler: In addition to high level languages and machine language, there is another language called the assembly language. Assembly language is in between the high-level languages and machine language. It is closer to machine language than high level languages. It is also called low level language. This language is not easily readable and understandable by the programmer like a high-level programming language. The assembler works as the translator in converting the assembly language program to machine code.

### Difference Between Compiler and Interpreter and Assembler:

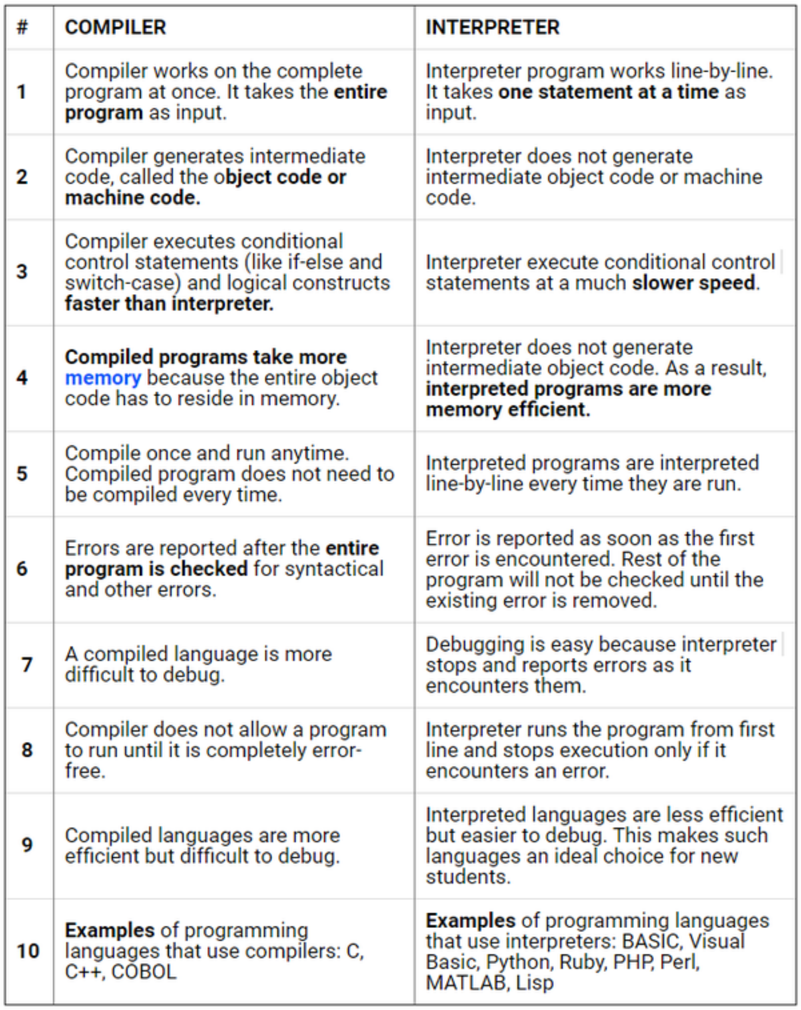
- A compiler is a software that converts programs written in a high-level language into executable machine code for a CPU or low-level language. An interpreter is a software that takes a source program and runs it line by line, translating each line as it comes to it. An assembler is a software that converts programs written in assembly language into machine language.
- Compiler converts the whole high-level language program to machine language or low level language at a time. Interpreter converts the high-level language program to machine language line by line. In contrast, assembler converts assembly language program to machine language.
- Languages such as C, C++ use compilers to convert the code. Languages such as Ruby, Perl, Python, PHP uses an interpreter and assembly language uses an assembler.
- Compiler, Interpreter and Assembler are language translators. The difference between compiler interpreter and assembler is that compiler converts whole high-level language programs to machine language at a time while interpreter converts high level language programs to machine language line by line and assembler converts assembly language programs to machine language.
- Compiler takes large amount of time to analyze the entire source code, but the overall execution time of the program is comparatively faster. Interpreter takes less amount of time to analyze the source code, but the overall execution time of the program is slower.
- Compiler generates the error message only after scanning the whole program, so debugging is comparatively hard as the error can be present anywhere. Interpreter's debugging is easier as it continues translating the program until the error is met in the program.
- Compiler generates intermediate object code. No intermediate code is generated by interpreter.
- Interpreter -> No intermediate object code is generated, hence are memory efficient. Compiler -> Generates intermediate object code which further requires linking, hence requires more memory.



COMPILER VS INTERPRETER VS ASSEMBLER

| | | |
|---|---|---|
| Software that converts programs written in a high level language into machine language | Software that translates a high level language program into machine language | Software that converts programs written in assembly language into machine language |
| Converts the whole high level language program to machine language at a time | Converts the high level language program to machine language line by line | Converts assembly language program to machine language |
| Used by C, C++ | Used by Ruby, Perl, Python, PHP | Used by assembly language |

Visit www.pediaa.com



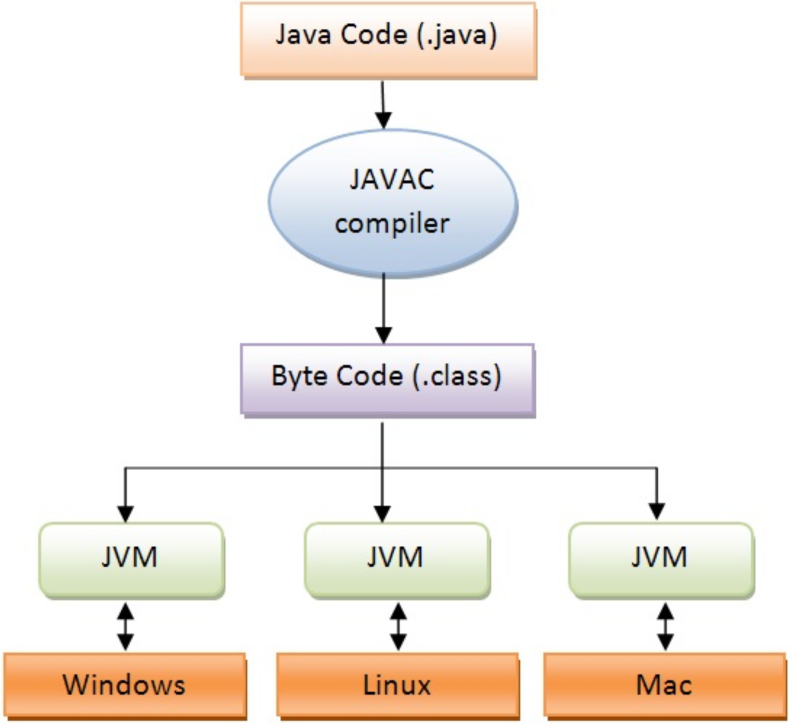
| # | COMPILER | INTERPRETER |
|---|---|---|
| 1 | Compiler works on the complete program at once. It takes the **entire program** as input. | Interpreter program works line-by-line. It takes **one statement at a time** as input. |
| 2 | Compiler generates intermediate code, called the o**bject code or machine code.** | Interpreter does not generate intermediate object code or machine code. |
| 3 | Compiler executes conditional control statements (like if-else and switch-case) and logical constructs **faster than interpreter.** | Interpreter execute conditional control statements at a much **slower speed**. |
| 4 | **Compiled programs take more memory** because the entire object code has to reside in memory. | Interpreter does not generate intermediate object code. As a result, **interpreted programs are more memory efficient.** |
| 5 | Compile once and run anytime. Compiled program does not need to be compiled every time. | Interpreted programs are interpreted line-by-line every time they are run. |
| 6 | Errors are reported after the **entire program is checked** for syntactical and other errors. | Error is reported as soon as the first error is encountered. Rest of the program will not be checked until the existing error is removed. |
| 7 | A compiled language is more difficult to debug. | Debugging is easy because interpreter stops and reports errors as it encounters them. |
| 8 | Compiler does not allow a program to run until it is completely error-free. | Interpreter runs the program from first line and stops execution only if it encounters an error. |
| 9 | Compiled languages are more efficient but difficult to debug. | Interpreted languages are less efficient but easier to debug. This makes such languages an ideal choice for new students. |
| 10 | **Examples** of programming languages that use compilers: C, C++, COBOL | **Examples** of programming languages that use interpreters: BASIC, Visual Basic, Python, Ruby, PHP, Perl, MATLAB, Lisp |

-------------------------------------------------------MORE-----------------------------------------------------

## Java Compile & Interpret:

**Compiler:** The Java compiler (javac) compiles Java code into bytecode. Bytecode is what the interpreter reads. The bytecode is platform-independent. This means it isn't targeted at any particular operating system; it targets the interpreter.

**Interpreter:** Java's interpreter is more correctly called the Java Virtual Machine (JVM). It reads the bytecode and translates it into something the hosting operating system can understand and execute. For example, "make a window" is done different ways on different operating systems (Windows, Linux, etc.).



- ASSEMBLY and ASSEMBLER: While it is possible to write computer programs as long lists of numbers (machine language) and while this technique was used with many early computers, it is extremely tedious and potentially error-prone to do so in practice, especially for complicated programs. Instead, each basic instruction can be given a short name that is indicative of its function and easy to remember – a mnemonic such as ADD, SUB, MULT or JUMP. These mnemonics are collectively known as a computer's assembly language. Converting programs written in assembly language into something the computer can actually understand (machine language) is usually done by a computer program called an assembler.

## Decimal and Hex Codes for Instruction Set
## NUMERICAL

| DEC | HX | CHAR | opc | ALform |
|---|---|---|---|---|
| 0 | 00 | ♥ | BRK | |
| 1 | 01 | ├ | ORA | (aa,X) |
| 5 | 05 | ┐ | ORA | aa |
| 6 | 06 | ╱ | ASL | aa |
| 8 | 08 | ◢ | PHP | |
| 9 | 09 | ▗ | ORA | #nn |
| 10 | 0A | ◣ | ASL | A |
| 13 | 0D | ▔ | ORA | aaaa |
| 14 | 0E | ▁ | ASL | aaaa |
| 16 | 10 | ♣ | BPL | aa |
| 17 | 11 | ┌ | ORA | (aa),Y |
| 21 | 15 | ▬ | ORA | aa,X |
| 22 | 16 | ▎ | ASL | aa,Y |
| 24 | 18 | ┴ | CLC | |
| 25 | 19 | ▌ | ORA | aaaa,Y |
| 29 | 1D | ↓ | ORA | aaaa,X |
| 30 | 1E | ← | ASL | aaaa,X |
| 32 | 20 | | JSR | aaaa |
| 33 | 21 | ! | AND | (aa,X) |
| 36 | 24 | $ | BIT | aa |
| 37 | 25 | % | AND | aa |
| 38 | 26 | & | ROL | aa |
| 40 | 28 | ( | PLP | |
| 41 | 29 | ) | AND | #nn |
| 42 | 2A | * | ROL | A |
| 44 | 2C | , | BIT | aaaa |
| 45 | 2D | - | AND | aaaa |
| 46 | 2E | . | ROL | aaaa |
| 48 | 30 | 0 | BMI | aa |
| 49 | 31 | 1 | AND | (aa),Y |
| 53 | 35 | 5 | AND | aa,X |
| 54 | 36 | 6 | ROL | aa,X |
| 56 | 38 | 8 | SEC | |
| 57 | 39 | 9 | AND | aaaa,Y |
| 61 | 3D | = | AND | aaaa,X |
| 62 | 3E | > | ROL | aaaa,X |
| 64 | 40 | @ | RTI | |
| 65 | 41 | A | EOR | (aa,X) |
| 69 | 45 | E | EOR | aa |
| 70 | 46 | F | LSR | aa |
| 72 | 48 | H | PHA | |
| 73 | 49 | I | EOR | #nn |
| 74 | 4A | J | LSR | A |
| 76 | 4C | L | JMP | aaaa |
| 77 | 4D | M | EOR | aaaa |
| 78 | 4E | N | LSR | aaaa |
| 80 | 50 | P | BVC | aa |
| 81 | 51 | Q | EOR | (aa),Y |
| 85 | 55 | U | EOR | aa,X |
| 86 | 56 | V | LSR | aa,X |
| 88 | 58 | X | CLI | |
| 89 | 59 | Y | EOR | aaaa,Y |
| 93 | 5D | ] | EOR | aaaa,X |
| 94 | 5E | ^ | LSR | aaaa,X |
| 96 | 60 | ♦ | RTS | |
| 97 | 61 | a | ADC | (aa,X) |
| 101 | 65 | e | ADC | aa |
| 102 | 66 | f | ROR | aa |
| 104 | 68 | h | PLA | |
| 105 | 69 | i | ADC | #nn |
| 106 | 6A | j | ROR | A |
| 108 | 6C | l | JMP | (aaaa) |
| 109 | 6D | m | ADC | aaaa |
| 110 | 6E | n | ROR | aaaa |
| 112 | 70 | p | BVS | aa |
| 113 | 71 | q | ADC | (aa),Y |
| 117 | 75 | u | ADC | aa,X |
| 118 | 76 | v | ROR | aa,X |
| 120 | 78 | x | SEI | |
| 121 | 79 | y | ADC | aaaa,Y |
| 125 | 7D | ↖ | ADC | aaaa,X |
| 126 | 7E | ◀ | ROR | aaaa,X |
| 129 | 81 | ├ | STA | (aa,X) |
| 132 | 84 | ┤ | STY | aa |
| 133 | 85 | ┐ | STA | aa |
| 134 | 86 | ╱ | STX | aa |
| 136 | 88 | ◤ | DEY | |
| 138 | 8A | ◥ | TXA | |
| 140 | 8C | ▄ | STY | aaaa |
| 141 | 8D | ▔ | STA | aaaa |
| 142 | 8E | ▁ | STX | aaaa |
| 144 | 90 | ♣ | BCC | aa |
| 145 | 91 | ┌ | STA | (aa),Y |
| 148 | 94 | ● | STY | aa,X |
| 149 | 95 | ▬ | STA | aa,X |
| 150 | 96 | ▎ | STX | aa,Y |
| 152 | 98 | ┴ | TYA | |
| 153 | 99 | ▌ | STA | aaaa,Y |
| 154 | 9A | └ | TXS | |
| 157 | 9D | ↓ | STA | aaaa,X |
| 160 | A0 | █ | LDY | #nn |
| 161 | A1 | ! | LDA | (aa,X) |
| 162 | A2 | " | LDX | #nn |
| 164 | A4 | $ | LDY | aa |
| 165 | A5 | % | LDA | aa |
| 166 | A6 | & | LDX | aa |
| 168 | A8 | ( | TAY | |
| 169 | A9 | ) | LDA | #nn |
| 170 | AA | * | TAX | |
| 172 | AC | , | LDY | aaaa |
| 173 | AD | - | LDA | aaaa |
| 174 | AE | . | LDX | aaaa |
| 176 | B0 | 0 | BCS | aa |
| 177 | B1 | 1 | LDA | (aa),Y |
| 180 | B4 | 4 | LDY | aa,X |
| 181 | B5 | 5 | LDA | aa,X |
| 182 | B6 | 6 | LDX | aa,Y |
| 184 | B8 | 8 | CLV | |
| 185 | B9 | 9 | LDA | aaaa,Y |
| 186 | BA | : | TSX | |
| 188 | BC | < | LDY | aaaa,Y |
| 189 | BD | = | LDA | aaaa,X |
| 190 | BE | > | LDX | aaaa,Y |
| 192 | C0 | @ | CPY | #nn |
| 193 | C1 | A | CMP | (aa,X) |
| 196 | C4 | D | CPY | aa |
| 197 | C5 | E | CMP | aa |
| 198 | C6 | F | DEC | aa |
| 200 | C8 | H | INY | |
| 201 | C9 | I | CMP | #nn |
| 202 | CA | J | DEX | |
| 204 | CC | L | CPY | aaaa |
| 205 | CD | M | CMP | aaaa |
| 206 | CE | N | DEC | aaaa |
| 208 | D0 | P | BNE | aa |
| 209 | D1 | Q | CMP | (aa),Y |
| 213 | D5 | U | CMP | aa,X |
| 214 | D6 | V | DEC | aa,X |
| 216 | D8 | X | CLD | |
| 217 | D9 | Y | CMP | aaaa,Y |
| 221 | DD | ] | CMP | aaaa,X |
| 222 | DE | ^ | DEC | aaaa,X |
| 224 | E0 | ♦ | CPX | #nn |
| 225 | E1 | a | SBC | (aa,X) |
| 228 | E4 | d | CPX | aa |
| 229 | E5 | e | SBC | aa |
| 230 | E6 | f | INC | aa |
| 232 | E8 | h | INX | |
| 233 | E9 | i | SBC | #nn |
| 234 | EA | j | NOP | |
| 236 | EC | l | CPX | aaaa |
| 237 | ED | m | SBC | aaaa |
| 238 | EE | n | INC | aaaa |
| 240 | F0 | p | BEQ | aa |
| 241 | F1 | q | SBC | (aa),Y |
| 245 | F5 | u | SBC | aa,X |
| 246 | F6 | v | INC | aa,X |
| 248 | F8 | x | SED | |
| 249 | F9 | y | SBC | aaaa,Y |
| 253 | FD | ↖ | SBC | aaaa,X |
| 254 | FE | ◀ | INC | aaaa,X |