**2)Compiler-**it is a computer program that translates computer code written in one programming language into another language(low level to high level)

**Object code** is machine-readable code that provides instructions to a target computing platform, as defined by its operating system and hardware architecture

**source code** is any collection of text, with or without comments, written using a human-readable programming language.

**A linker** is an important utility program that takes the object files, produced by the assembler and compiler, and other code to join them into a single executable file.

3.

a)Preprocessing

Preprocessor directives include;including libraries,managing macros and removal of comments.

Conditional compilation also done:include and excludes certain directives

b)Compilation

The compiler checks for syntax or structure errors –in case of errors the compilation process stops and displays the corresponding errors–. After compiling it, it generates an intermediate code in assembly language

c)Assembling

Code generated in the compilation stage is converted into a machine code specific to the target processor architecture.The assembler takes the assembly code as input and produces a relocatable object file.

d)Linking

The linker here combines object files,resolves external references, handles library dependency and genenrates an executable file.

In our case it handles the input of the two integers from the loader and gives a final result.

4.

|  |  |
| --- | --- |
| compiler | interpreter |

|  |  |
| --- | --- |
| 1. Compilers convert the source code to object code. | Interpreters do not convert the source code into object code. |
| 1. The execution time of compiler is less, hence it is preferred. | It is not preferred due to its slow speed. Usually, interpreter is slow, and hence takes more time to execute the object code. |
| 1. Compiler doesn’t require the source code for execution later. | It requires the source code for execution later. |
| 1. Compiler can check syntactic and semantic errors in the program simultaneously. | Interpreter checks the syntactic errors only. |
| 1. Compiler are larger in size. | Interpreters are smaller in size. |

1. Compilers are not flexible Interpreters are relatively flexible

|  |
| --- |
|  |

5)

a)Bitwise operators

Bitwise AND: converts the two operands into binary and performs conjunctive operation bit by bit. Bitwise OR: converts the two operands into binary and performs disjunctive operation bit by bit.

Bitwise LEFT SHIFT:

Bitwise RIGHT SHIFT:

Bitwise XOR: converts both operands into binary and performs xor operation bit by bit

Bitwise ONE’S COMPLEMENT: returns the complementary form of the operand.

b) Arithmetic Operators

Include:Addition

Subtraction

Division

Multiplication

Modulus

c)Relational operators

!=not equal to

==equals to

<=lesser than equal to

>=greater than equal to

<lesser than

>greater than

d) Assignment operators

=assignment

+=addition assignment

/=division assignment

\*=multiplication assignment

-=subtraction assignment

%=modulus assignment

e)Special operators

f)Conditional operators

example syntax ;VariableName = (condition) ? TrueValue : FalseValue

example:a= (b>c) ? (b+c) : (b-c);

g) Increment(++) and decrement(-) operators

h)Logical operators