ECAP202:- OOPS

CH-2:- BASICS OF C++:

C:- It is a powerful general-purpose programming language developed by Dennis M. Ritchie at Bell Laboratories
It can be used to develop S/w like O.S, DB, compilers & so on
C programming is an excellent language to learn to program for beginners

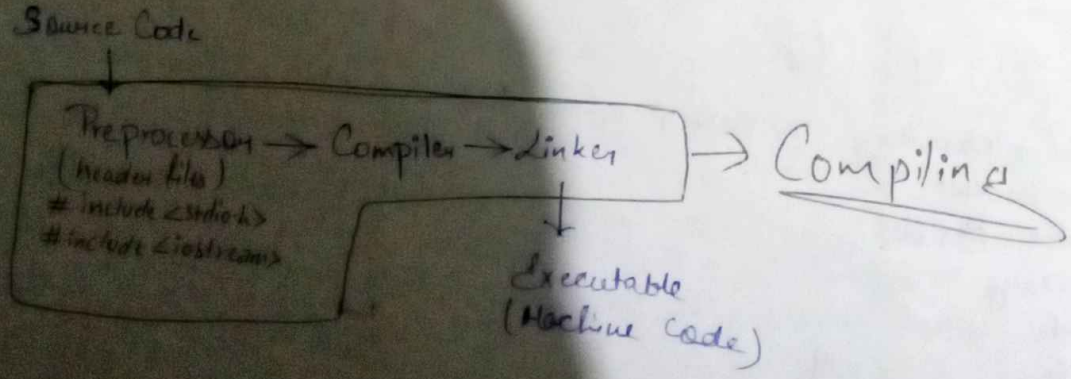
C++:- It is a middle level programming language developed by Bjarne Stroustrup starting in 1979 at Bell labs.
It runs on variety of platforms, such as Windows, Mac OS & the various versions of UNIX.
It is obj. oriented programming.

C Vs. C++:-

C	C++
C follows the procedural style Programming	C++ is multi-paradigm. It supports both procedural & obj. oriented
Data is less secured in C	In C++, you can use modifiers for class members to make it inaccessible for outside use.
C follows the Top-down approach	C++ follows the bottom-up approach
Scanf() & printf() are mainly used for input/output	C++ mainly use stream cin & cout to perform input & output opn.
C does not support the inheritance	C++ supports inheritance

Compiling:- The modified source code is compiled into binary obj. code. This code is not yet executable.

⇒ Compiling Process:-



Linking :- The obj. code is combined with required supporting code to make an executable program.

⇒ Linking Process :-

Preprocessing (Modifies the original program acc. to the directives that start with '#')

↓

Compilation (Translates the program into a obj. file containing machine language code)

↓

Linking (Handles merging & make executable file)

⇒ Compiling & Linking :-

C/C++ Source Text → Compiler → Obj. Code

Obj. Code → Linker → Machine Code

Compiling of C++ Program :-

The compilation of a C++ program consists of 3 steps :-

i) Preprocessing :- It is just a text substitution tool & it instructs the compiler to do required pre-processing before the actual compilation. It handles preprocessing directives like #include, #define etc.

ii) Compilation :- The compilation takes place on the preprocessed files. The compiler parses the pure C++ source code & converts it into assembly code. The compiler won't give an error unless the source code is not well-formed.

iii) Linking :- The linker produces the final compilation output from the obj. files the compiler produced. This output can be shared (or dynamic) lib. or an executable.

Tokens :- A token is the smallest unit of a program that the compiler understands. In C++, Tokens are divided into

1) Keywords :- ~~In a programming~~ lang reserved words that have fixed meaning

alignas ; case ; alignof ; asm ; auto , bool , break , catch ; char ; char16_t ; char32_t ; class ; const ; constexpr ; const_cast ; continue ; decltype ; default ; delete ; double ; do ; dynamic_cast ; else ; enum ; explicit ; export ; extern ; FALSE ; float ; for ; friend ; goto ; if ; inline ; int ; long ; mutable ; namespace ; public ; new ; ~~no exp~~ noexcept ; nullptr ; operator ; Private ; protected ; register ; reinterpret_cast ; return ; short ; signed ; sizeof ; static ; static_assert ; static_cast ; struct ; switch ; template ; this ; thread_local ; throw ; TRUE ; try ; typedef ; typeid ; typename ; union ; unsigned ; using ; virtual ; void ; volatile ; wchar_t ; while

ii) Identifiers:- assign names of the programmer's choice to variables, arrays, functions, structures, classes & various other data structures.
The programmer may use the mixture of different types of identifiers. The character sets available in C++ to name an identifier.

Rules:-
i) First character:- 1st character should begin with either an alphabet or an underscore. not with a number.
ii) No special characters
iii) No keywords
iv) No whitespaces
v) Word limit:- not exceed 31 characters
vi) Case sensitive

iii) Constant:- variables whose value cannot be changed. const keyword is used for declare constant.
e.g:- const float pi = 3.14;

Types:-
- Integer constants
- Floating ,,
- Character ,,
- String ,,
- Octal ,,
- Hexadecimal ,,

iv) Strings:- Stores a sequence of characters. It terminates with a null character '\0'. Unlike characters, strings in C++ are always enclosed double quotes (" ").
e.g:- char name[30] = "Hello";

v) Special Symbols:- Special symbols which have special meaning to the compiler. We cannot alter their meaning

e.g:- [] :- used for single dimensional & multidimensional subscripts of arrays
() :- ,, ,, function calls & parameters
{} :- ,, to indicate the beginning & end of a code block
, :- ,, ,, separate multiple statements like parameters in a function
: :- ,, ,, invoke an initialization list
; :- Also called statement terminator, it is used to mark the end of statements
* :- Used to create pointers
# :- ,, as a preprocessor directive to include header files & define constants
. :- Used to access a str. members.
~ :- ,, as a ~~destructor~~ destructor

vi) Operators:- Symbols that operate on operands. These operands can be variable or values. Operators help to perform mathematical & logical computations.

Common C++ Operators :- Arithmetic ; Assignment ; Relational ; Logical ; Bitwise ; Other operators

= D-type in C++ :- Defines the type of data a variable can hold.

Types :-

i) Primitive :- Int, Char (0 to 255), Boolean, Floating Point, Double Floating Point, (Valueless) Void, Wide character

ii) Derived :- Func^n, array, pointer, Reference

iii) User defined :- Class, Struct., Union, Enum, Typedef

Primitive :- ~~User~~ Built-in or predefined d-types & can be used directly by the user to declare a variable.

Derived :- derived from primitive or built-in d-types.

User-defined :- defined by user itself. Also called abstract d-type

i) Class :- Building block that leads to OOP. It holds both members & func^n

ii) Structures :- Store a group of items of non-similar d-types.

iii) Union :- As Struct. Only one member is initialized.

iv) Enum :- Construct multiple

v) Typedef :- define explicitly new d-type names. ~~by use~~

vi) Reference :- alternate name of already existing variable. It can not be changed to refer another variable & should be initialized at the time of declaration & cannot be NULL