COMPUTER SCIENCE 10TH – DETAILED QUESTION ANSWERS

➔ BASICS OF PROGRAMMING IN C++

Chapter # 02

ADAMJEECOACHING.BLOGSPOT.COM

Q.1: Define Computer Program?

Ans. COMPUTER PROGRAM:

A computer program is a set of instructions that performs a specific task when executed by a computer. A computer requires programs to function and typically executes the program's instructions in a central processing unit. A computer program is usually written by a computer programmer in a programming language.

Q.2: Define syntax OR What is syntax in programming?

Ans. SYNTAX IN PROGRAMMING LANGUAGE:

Syntax refers to the rules that define the structure of a language. Syntax in computer programming means a set of keywords and characters that a computer can understand, interpret and perform task associated with them. The syntax of a language must be followed, and if it is not followed, the code will not be understood by a compiler or interpreter. Different programming languages have different types of syntax.

Q.3: Describe classification of programming language.

CLASSIFICATION OF PROGRAMMING LANGUAGE:

Based on the accessibility of hardware, programming languages can be classified into the following categories:

1. Low-level language
2. Middle-level language
3. High level language

Q.4: Define low level language.

LOW-LEVEL LANGUAGE:

Low-level programming languages are those languages that are directly communicated with computer hardware. The two languages come under this category are Machine language and Assembly language.

Q.5: Define machine language.

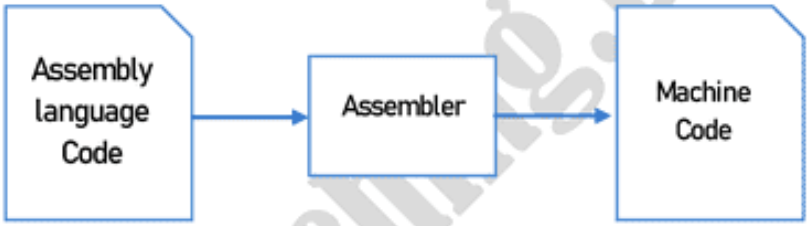
Ans. MACHINE-LEVEL LANGUAGE

Machine language is a collection of binary digits or bits that the computer reads and interprets. This language is the only language understood by computers while machine language is almost impossible for humans to use because they consist entirely of numbers (0s & 1s). Machine code doesn't require translator because machine code is directly executed by computer. It is also called first generation language.

Q.6: Define assembly language.

Ans. ASSEMBLY LANGUAGE

A program written in assembly language consists of a series of instructions called mnemonics that correspond to a stream of executable instructions. Assembly language code is translated by a translator called assembler. Assembly language uses keywords and symbols much like English and are easy to read, write and maintain as compared to machine language. It is also called second generation programming language.



Q.7: Define middle level language.

Ans. MIDDLE-LEVEL LANGUAGE:

The middle-level language lies in between the low level and high-level language. Middle-level language actually binds the gap between a machine level language and high-level languages, these languages are now become obsolete and are not in used.

Q.8: What do you know about high level language?

Ans. HIGH-LEVEL LANGUAGE:

High-level languages are relatively new and drastically revolutionized the programming world. It allows us to write computer code using instructions resembling everyday spoken language, usually English (for example: print, if, while). Programs written in a high-level language need to be translated by a translator (compiler or interpreter) into machine language before execution.

Q.9: write down the advantages or benefits of high-level language.

Ans. ADVANTAGES OF IDGH-LEVEL LANGUAGE

- High level language is much closer to human language so it is more suitable to write code in high level language.
- It is more or less independent or portable of the particular type of computer used.
- It is easier to read, write and maintain.

Q.10: Write down the differences between low-level and high-level languages.

Ans. DIFFERENCES BETWEEN LOW-LEVEL LANGUAGE AND IDGH-LEVEL LANGUAGE

Low Level Language	High Level Language
In low level language machine codes (0 and 1) are used as an instruction to the computer.	In high level language English like words are used as an instruction to the computer.
The execution of programs is quite fast.	The execution of programs is not very fast.
Instructions are directly understood by the CPU	Instruction are not directly understood by the CPU
No need to translate program. In case of assembly language assembler is required.	Translation of program is required.
The programs written in low level languages are machine dependent and are difficult to modify.	The programs written in high level languages are machine independent and are easy to modify.
The examples of low-level languages are: (1) Machine language (2) Assembly language	The examples of high-level languages are: BASIC, FORTRAN, COBOL, PASCAL, C languages etc.

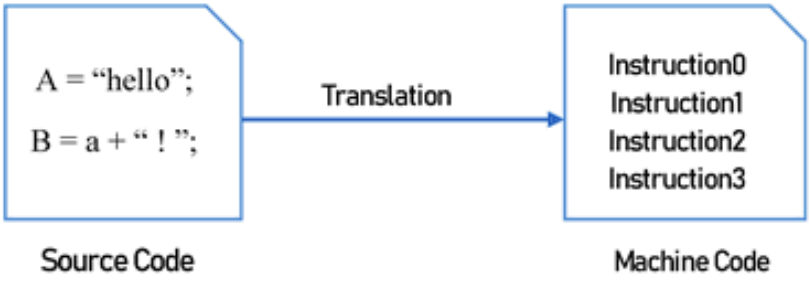
Q.11: Define translators or language translators.

Ans. TRANSLATORS OR LANGUAGE TRANSLATORS

Language translator is a computer program that converts high level language program into low level program (1s & 0s), or machine language. It transforms the source code into the object code (machine code) which understands directly by the computer processor. Translators also detect and report errors in the process of translation.

There are three types of language translators.

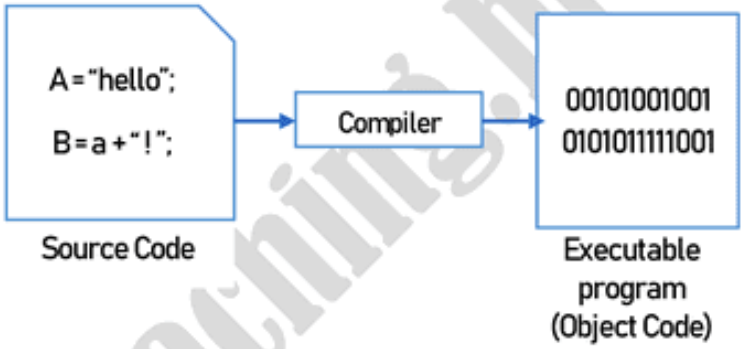
* Interpreter * Compiler * Assembler



Q.12: What is compiler?

Ans. COMPILER

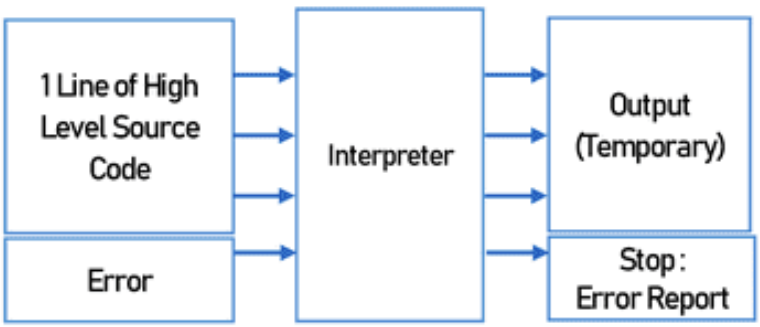
Compiler is a program that translates the high-level language program into machine language. Compiler translates the whole program at a time at once before it executed and makes a separate object file for the translated program. Translated program can be used multiple times without the need of retranslation of source codes. Each high-level language has its own compiler.



Q,13: Define Interpreter?

Ans. INTERPRETER

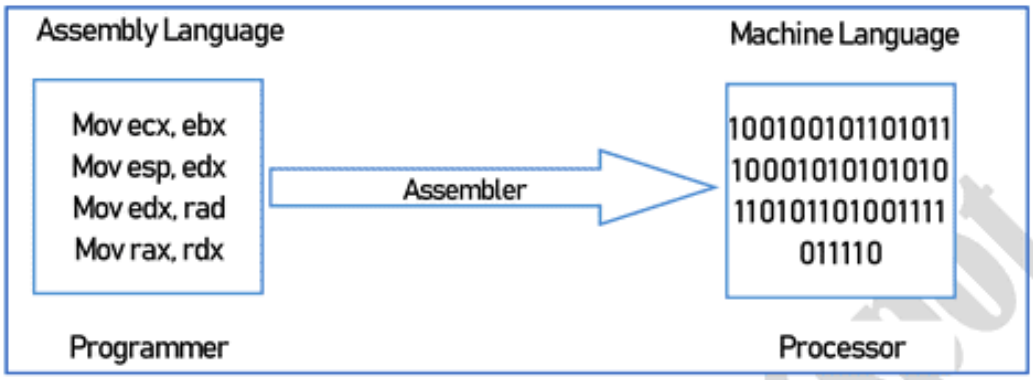
Interpreter is a language translator program, which converts high level program into machine language. It translates one instruction at a time. Interpreter does not make any object file and it translates the program every time when executed. An interpreter is faster than a compiler as it immediately executes the codes.



Q.14: Define Assembler.

Ans. ASSEMBLER

An assembler is a translator that converts assembly language program into machine language. An assembler translates assembly language code directly into machine code that can be understood by the CPU.



Q.15: Define source program or source code?

Ans. SOURCE PROGRAM / SOURCE CODE

A program or A program written by a programmer in any language other than machine language is called a source program or source code.

Q.16: Define object program or object code.

Ans. OBJECT PROGRAM / OBJECT CODE

Object program is a program or code that is converted into machine language. (OR) The output from a language translator, which consists of machine language instructions, is called the object program.

Q.17: Write down the differences between interpreter and compiler?

Ans. DIFFERENCES BETWEEN INTERPRETER AND COMPILER

Interpreter	Compiler
Interpreter translates high level language program into machine language line by line.	Compiler translates high level language program into machine language as a whole.
The interpreter translates the program every time when executed.	The compiler translates the program once at a time.

The interpreter does not make any object file.	The compiler makes a separate object file for the translated program
Errors are displayed for every instruction interpreted if any.	Errors are displayed after entire program is checked.
Examples: BASIC, LISP etc.	Examples: C Compiler, C++ compiler etc.

Q.18: Define errors or programming errors.

Ans. ERROR

An error describes any issue that arises unexpectedly that causes a program to not function properly. In general, there are three types of errors that occur in a computer program. Syntax Error, Logical Error and Runtime Error.

Q.19: What is syntax error?

Ans. SYNTAX ERROR

A syntax error is an error in the syntax due to violation of rules of whatever language program is being written. These are sometimes caused by simple typing mistakes.

Q.20: What is Logic error?

Ans. LOGIC ERROR

In computer programming, a logic error is a bug or error in planning the program's logic. They do not cause the program to crash or simply not work at all, they cause it to "misbehave" in some way, rendering wrong output of some kind. The computer does not tell you what is wrong.

Q.21: What is runtime error?

Ans. RUNTIME ERROR

A runtime error is a program error that occurs when a program is run on the computer and the result are not achieved due to some misinterpretation of a particular instruction. The code doesn't have any syntax or logic error but when it executes it cannot perform specific task.

Q.22: What is Programming Environment of C++ ?

Ans. PROGRAMMING ENVIRONMENT OF C++

Programming environment is an environment which supports execution of programming language smoothly and efficiently on a local computer to compile and run programs.

Q.23: Define IDE or integrated development environment?

Ans. INTEGRATED DEVELOPMENT ENVIRONMENT (IDE)

An integrated development environment (IDE) is software for building applications or programs that combines common developer tools into a single graphical user interface (GUI). IDE facilitates the development of applications designed to encompass all programming tasks in one application, one of the main benefits of an IDE is that they offer a central interface with all the tools, a developer needs. Dev-C++ is used for writing programs in C++ language, however there are many multiple language IDEs.

Q.24: Write down the benefits or advantages of Integrated development environment.

Ans. BENEFITS / ADVANTAGES OF INTEGRATED DEVELOPMENT ENVIRONMENT (IDE)

- IDE combines all tools that need for development. Programmers don't need to switch between different tools to design a layout, write the code, debug, build, etc.
- Many IDEs incorporate basic spelling checkers, so automatically check for errors to improve code.
- Libraries provide for functions in IDEs that are not included in the core part of the programming language.

Q.25: What are the components of Integrated Development Environment?

Ans. COMPONENTS OF INTEGRATED DEVELOPMENT ENVIRONMENT (IDE)

IDEs combining common tools that are necessary for a programmer to develop program, these are:

EDITING SOURCE CODE

Writing and editing source codes is a major part of programming. A text editor is used for writing and editing source codes with feature providing language specific autocompletion, and checking for bugs as code is being written.

SYNTAX HIGHLIGHTING

Syntax highlighting is a feature of IDEs that provides visual cues. keywords, and other words that have special meaning in languages are highlighted. This feature makes code easier to read or understand.

CODE COMPLETION

It is a feature of IDE that completely knows the language syntax that speeds up the process of coding by reducing typos and other mistakes. Autocompletion popups while typing, querying parameters of functions, query hints related to syntax errors.

COMPILER

Compiler is a component of IDEs that translate source code into machine code. IDEs provide automated build process. This feature can help automate developer tasks that are more common to save time.

Q.26: Define Linker, Loader and Debugging?

Ans. LINKER

Linker connects or links referenced library files with compiled code. It saved the linked objects into an executable file.

LOADER

This is the operating system's program that loads executable files into memory and directs the CPU to start running the program as directed by the IDE.

DEBUGGING

The process of removing errors from a program is known as debugging. Debugging

Q.27: What do you know about Dev C++?

Ans. INTRODUCTION TO DEV-C++

Dev-C++ is a fully featured graphical IDE (Integrated Development Environment) for programming in CIC++. Dev-C++ is developed by Bloodshed software. It was originally developed by Colin Laplace and first released in 1998. It is written in Delphi. With Dev C++ programmer can write Windows or console-based C/C++ programs easily.

Q.28: Write down the installation procedure of Dev C++.

Ans. INSTALLING AND CONFIGURING DEV-C++ IDE

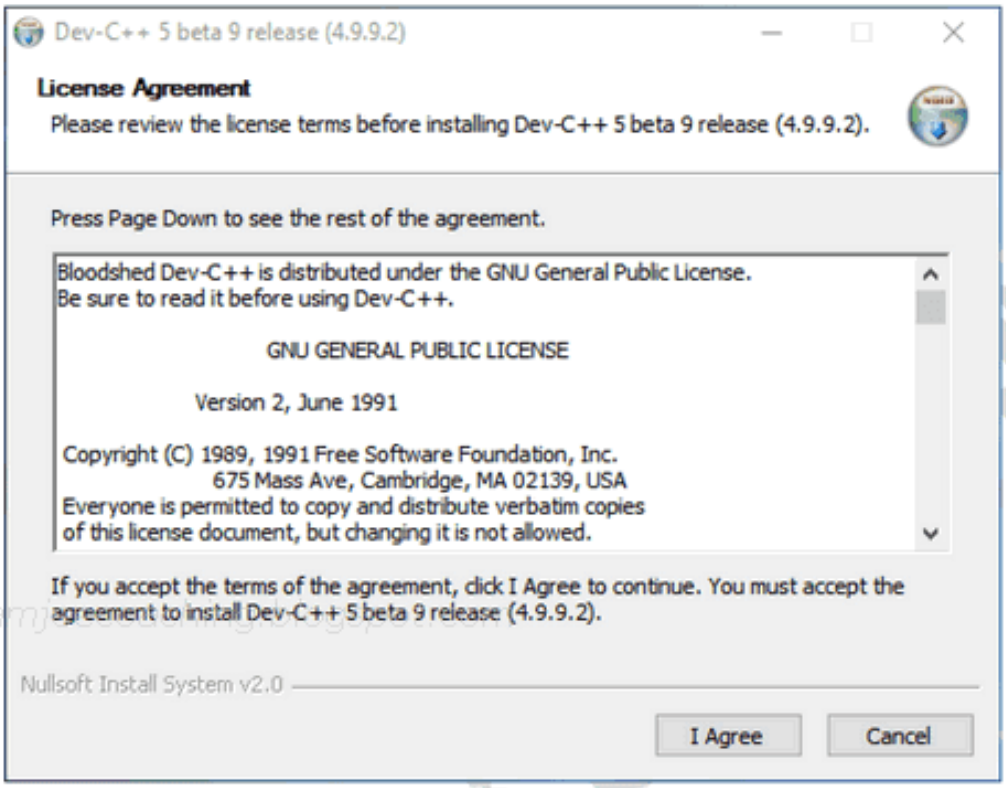
Dev-C++ is free available for download on Internet. After downloading the package begin the installation process. Following are the steps for installing Dec-C++ IDE.

STEP 1:

Select "English" as the language to be used for installation process.

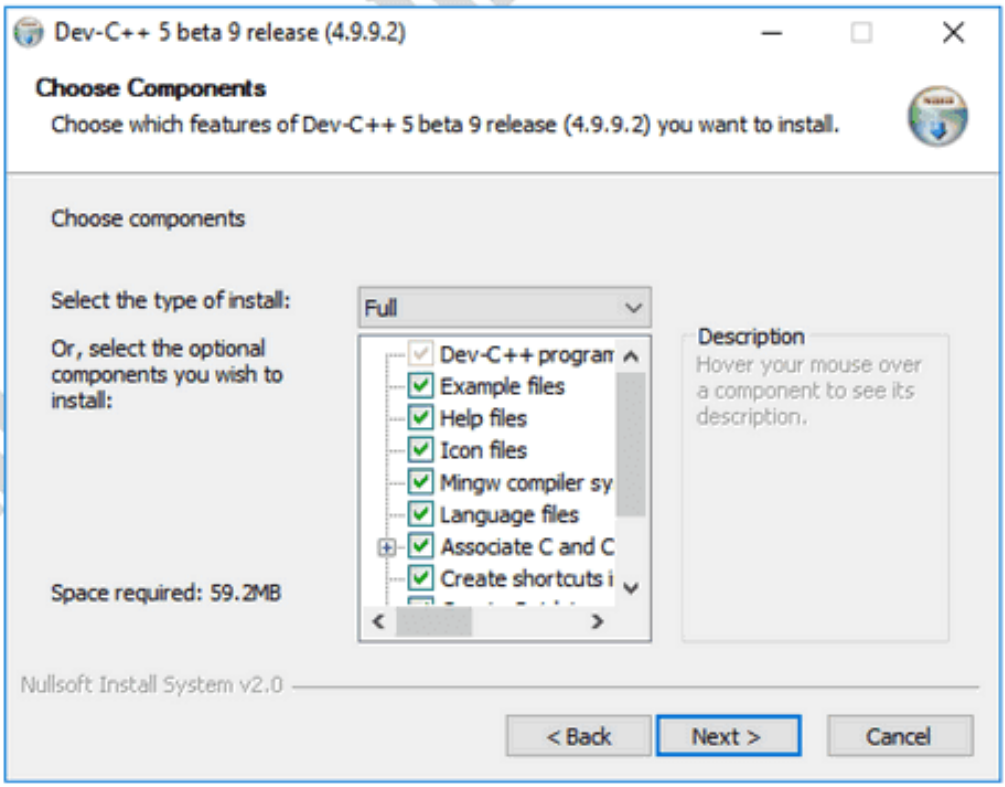
STEP 2

Press "I Agree" button to agree the license agreement.



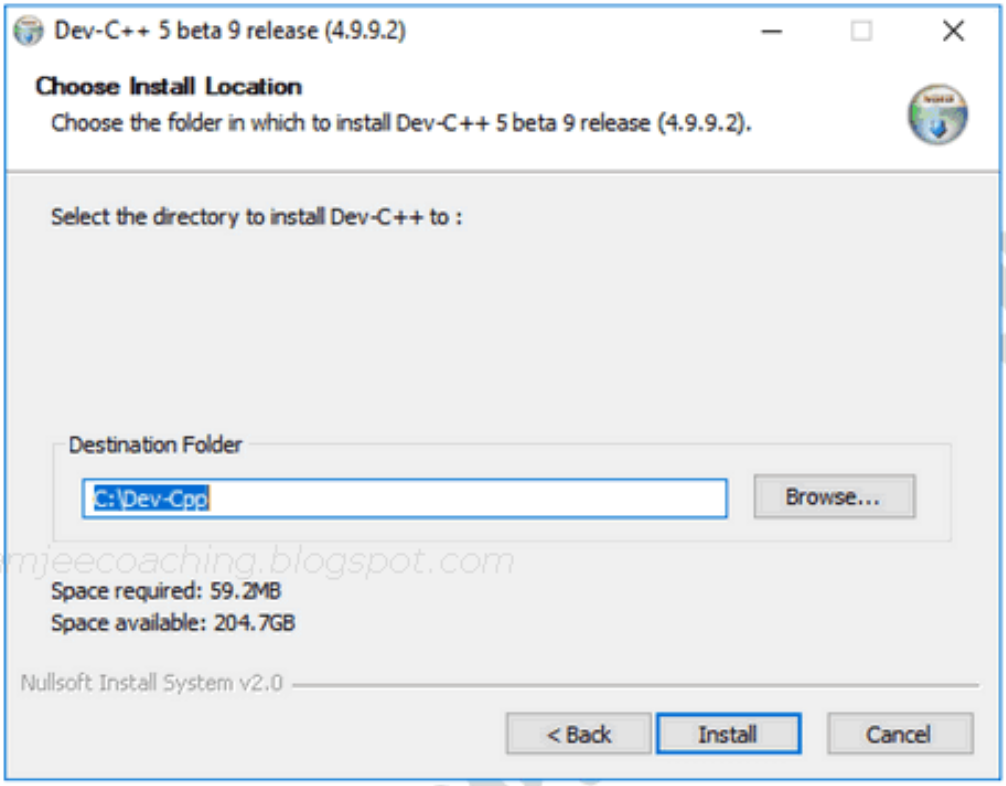
STEP 3

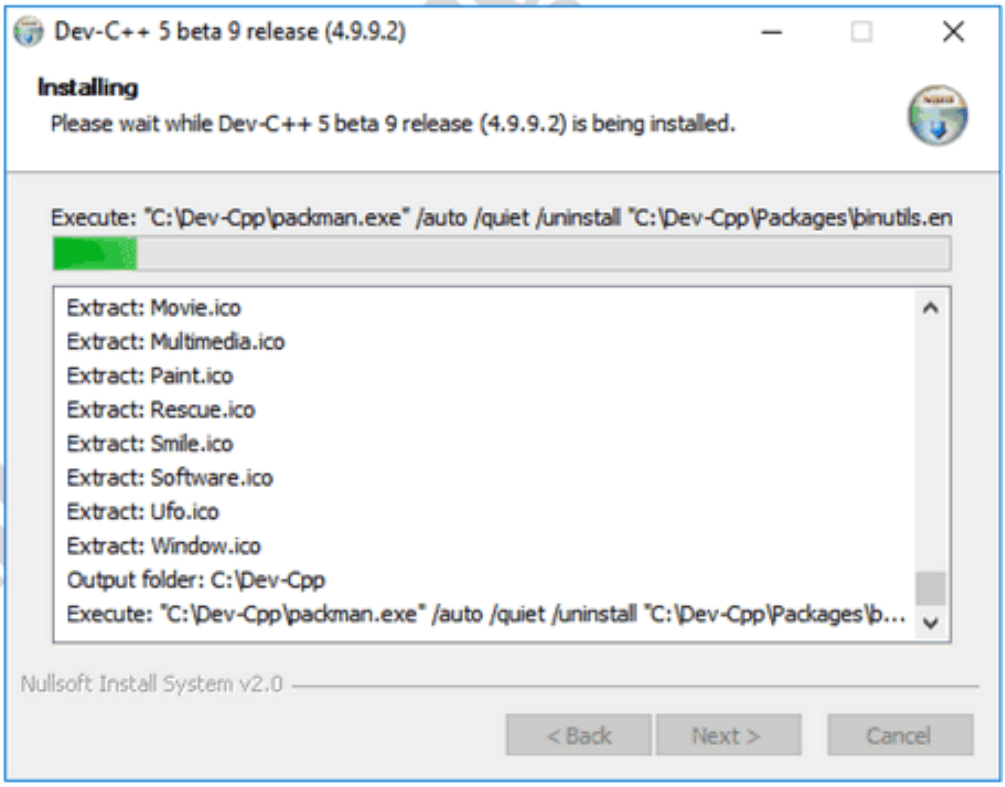
Select "Full" from the dropdown menu for "type of install". Click on "Next" to proceed.

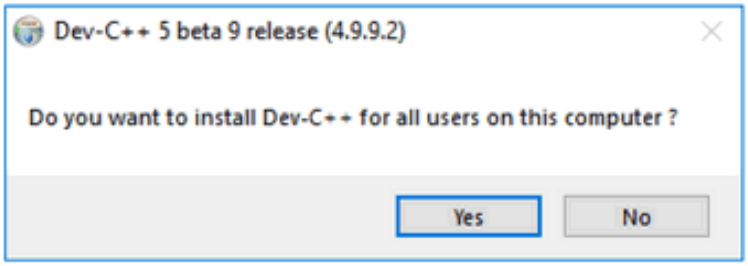


STEP 4

Select the installation folder where Dev-C++ files and libraries will be installed. Click on "Install" to start installation.

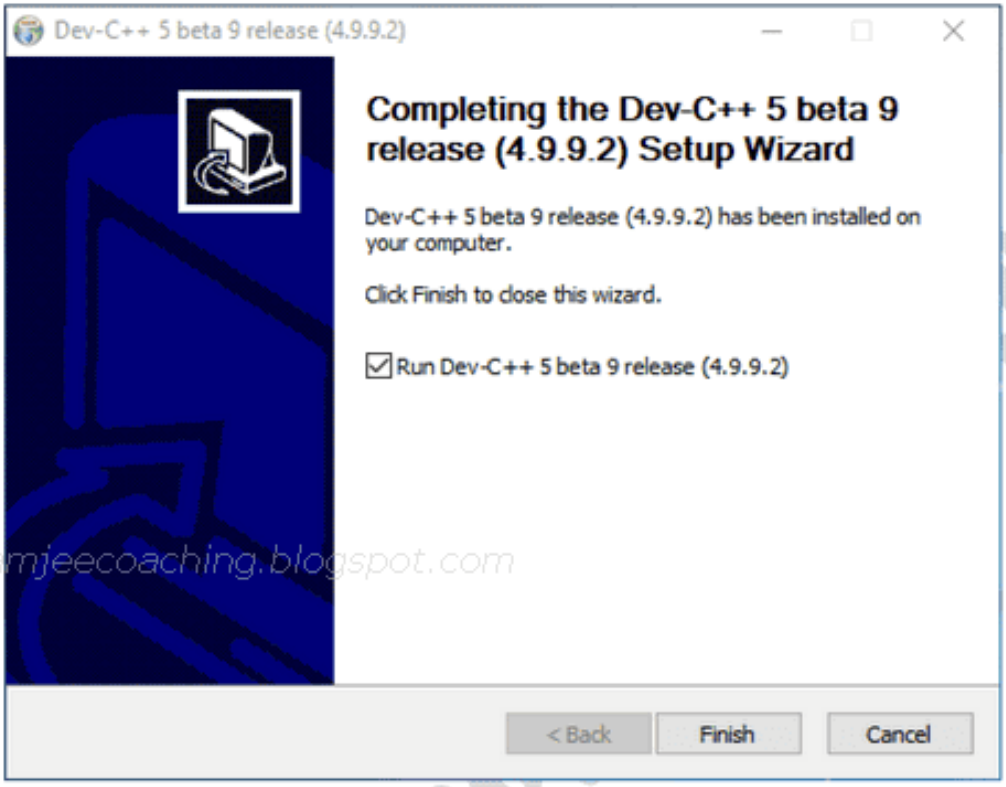






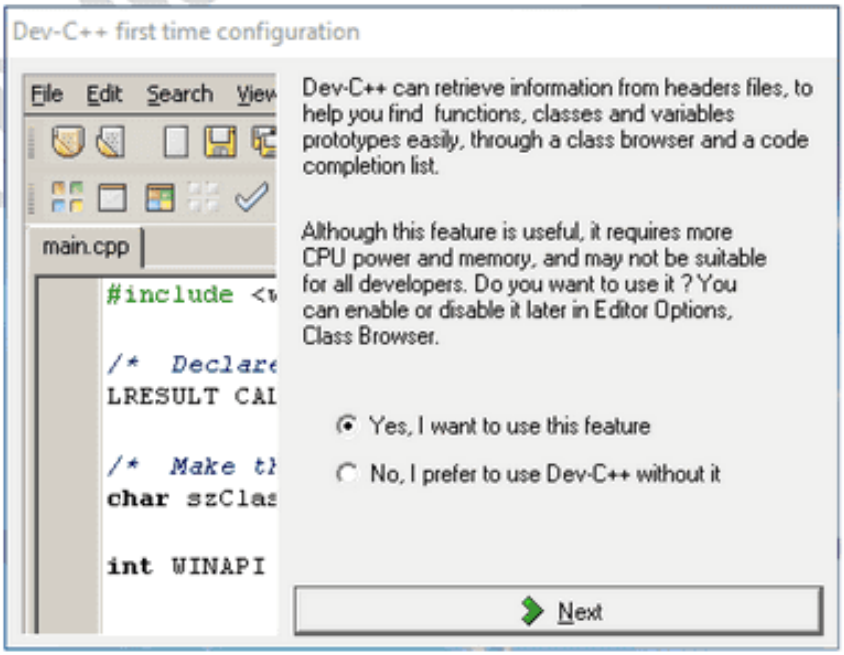
STEP 5

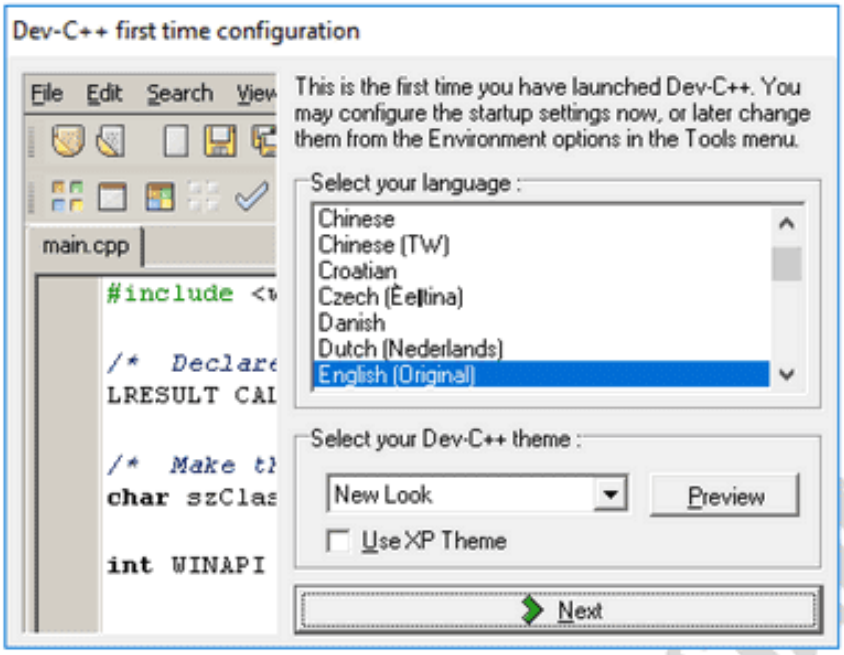
After completing process, it will show a "Finish" dialog box. Click "Finish" burton. This will automatically start Dev-C++ after installation completes.



CONFIGURING DEV-C++

Dev-C++ will require some configuration when it runs first time. Set" English (Original)" as default language and click "Next" to continue. On the "Theme" selection dialog box leave the default setting and click on "Next" and "OK" to continue.

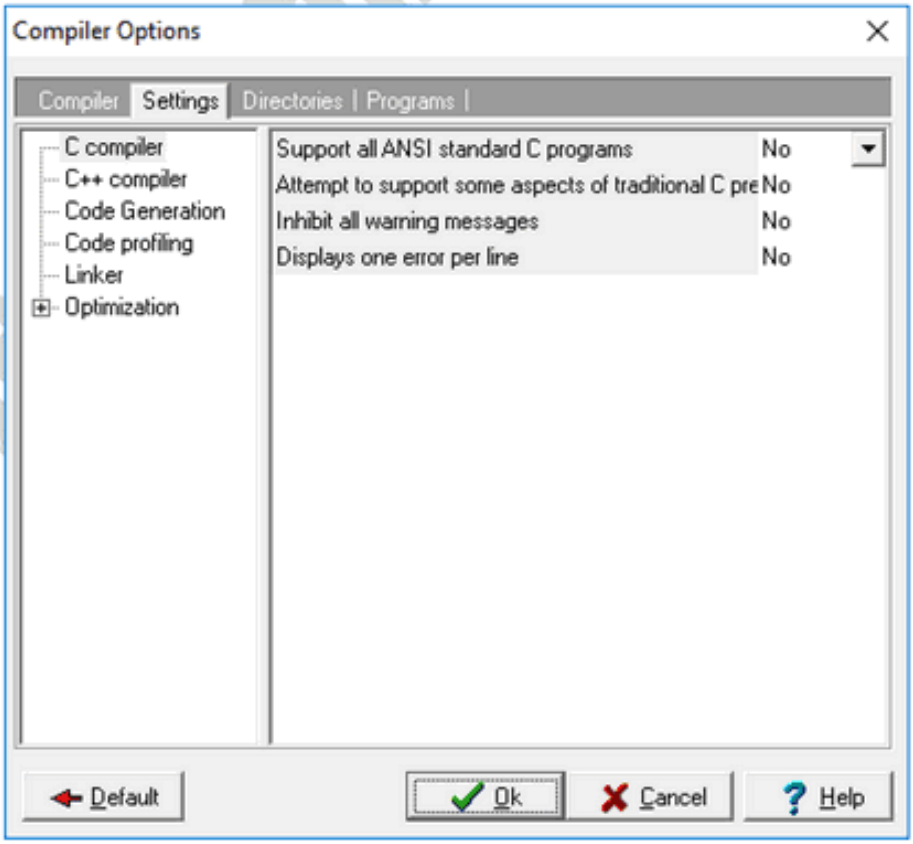




LINKER SETTING FORDEBUGGING

Linker setting for debugging is required first time to obtain information about problems in source code. The following steps are used to enable this configuration.

1. Click on Tools then Compiler Options and open the Settings tab.
2. Under Settings tab, open Linker tab. In Linker tab change the Generate Debugging Information (-g3) options to Yes.
3. Click on OK to save settings.

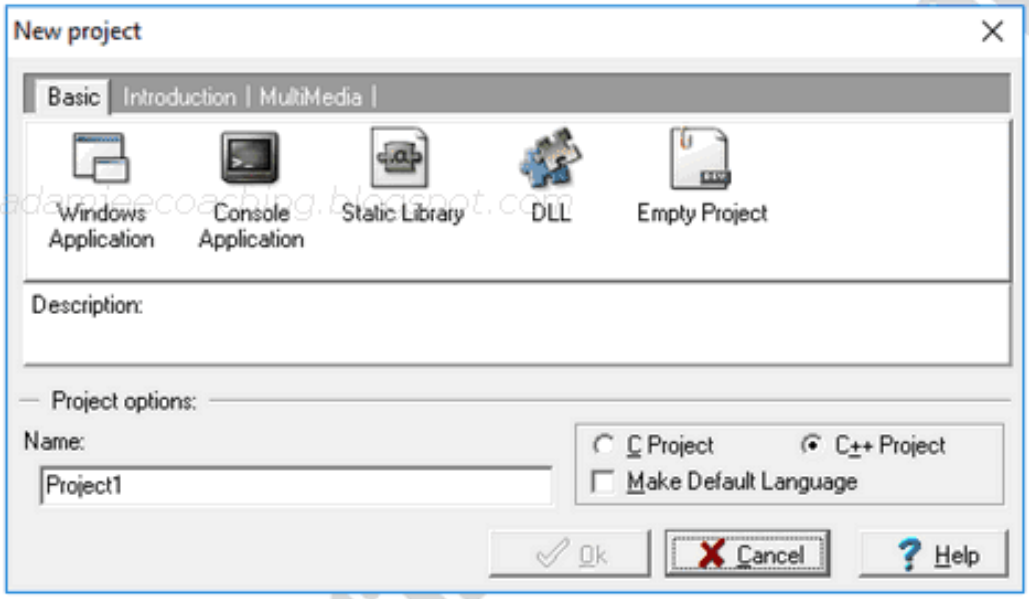


Q.29: Write down the procedure to develop a program in Dev C++.

Ans. **DEVELOPING PROGRAM IN DEV-C++**

Development of C++ program requires writing source code and saving those files for compilation. The steps to create a new project in Dev-C++ are:

1. Click on File then click New » Project.
2. In New Project dialog, Select Empty Project then select C++ Project. Also enter Name for project.
3. Click on OK. Dev C++ will ask path to save the new project, enter path and save project.



ADD NEW FILES TO PROJECT IN DEV-C++

The steps to create a new file are:

1. Click on Project » New File.
2. Click on Yes on the confirm dialog box.
3. To save file, click on File» Save. Enter the path and provide its name. Click on Save to Store the file.

COMPILE & EXECUTE PROJECT IN DEV-C++

The steps to compile and execute project are:

1. To compile, click on Execute » Compile or press F9 key.
2. After successfully compiling the project, run it by clicking on Execute » Run or by pressing F10 key.
3. A console will open and show the output of the program.

Q.30: What do you know about C++ programming language?

Ans. C++ PROGRAMMING LANGUAGE

C++ is a powerful general-purpose programming language. It was created by Bjarne Stroustrup in 1979 at Bell Laboratories. It is used to develop operating systems, browsers, games, and other applications. C++ supports mainly support programming like object oriented. C++ is a flexible language aims to make writing programs easier and more pleasant for the individual programmer.

Q.31: Define reserved words in C++.

Ans. RESERVED WORDS IN C++

Reserved words are keywords that have standard predefined meanings in C++ language. These keywords can only be used for their intended purpose; they cannot be used as programmer defined identifiers.

The following list the keywords or reserved words of the C++ language:

asm	else	new	this
auto	enum	Operator	Throw
bool	explicit	private	true
break	export	protected	try
case	extern	public	typedef
catch	false	register	typeid
char	float	reinterpret_cast	typename
class	for	return	union
const	friend	short	unsigned
const_cast	goto	signed	Using
continue	if	sizeof	virtual
default	inline	static	void
delete	int	static_cast	volatile
do	long	struct	wchar_t
double	mutable	switch	while
dynamic_cast	namespace	template	and
or	not	requires	nullptr

Q.32: Define data types in C++.

Ans. C++ DATA TYPES

Data values passed in a program may be of different types. Each of these data types are represented differently within the computer's memory and have different memory requirements. These data types can be augmented by the use of data type qualifiers / modifiers.

The data types supported in C ++ are described below:

DATA TYPE	KEYWORD	SIZE	RANGE
Boolean	bool	1 Bytes	0 (false), 1 (true)
Character	char	1 Bytes	-127 to 127 OR 0 to 255
Integer	int	4 Bytes	-2.147483648 to 2147483647
Floating Point	float	4 Bytes	1.5×10^{-45} to 3.4×10^{38}
Double Floating Point	double	8 Bytes	5.0×10^{-345} to 1.7×10^{308}

Q.33: What is constant? Also define its types.

Ans. CONSTANT

A constant is an identifier whose value remains unchanged throughout the program. Constants are used in two ways, they are:

1. Literal Constants
2. Defined Constants

Q.34: Define literal constant.

Ans. LITERAL CONSTANT

Literal constants are data used for representing fixed values. They can be used directly in the code.

Example: 1,2.5, 'c', "good" etc.

Q.35: Define symbolic constant.

Ans. DEFINED OR SYMBOLIC CONSTANT

In C++, we can create symbolic constant whose value remains unchanged but used as a variable. A symbolic constant can be created using the #define preprocessor directive or const keyword.

Example:

const int LIGHT_SPEED = 299792458;

#define LIGHT SPEED 299792458

Q.36: . Define variable and rules for naming variables.

Ans. VARIABLE

A variable is nothing but a name given to a storage area that our programs can manipulate. Its value can change during program execution. Each variable in C++ has a specific data type, which determines the size and layout of the variable's memory.

RULES FOR NAMING VARIABLE

- A variable name contains alphabets, numbers, and the underscore.
- A variable name must start with a letter or an underscore.
- Variable names are case sensitive. (Sum and sum are different)
- A variable name cannot be a keyword.
- A variable name cannot be longer than 32 characters.

Q.37: Define Declaring (Creating) and Initializing Variables.

Ans. DECLARATION (CREATING) VARIABLES

Variable declaration is a process in which we create storage space for variable in memory. A variable declaration consists of data type and name of the variable written as follow:

data_type variable_name;

int sum;

INITIALIZATION

Assign initial value to a variable is known as variable initialization. It can be initialized during declaration or separately. The equal sign is used to assign value written as follows:

data_type variable_name = value;

int sum = 3;

Q.38: Define strings in C++.

Ans. STRINGS IN C++

Variables that can store alphanumeric value that consist of multiple characters are called strings. In C++, strings are used by one-dimensional array of characters, which is terminated by a null character \0.