

# QB365 Question Bank Software Study Materials

## Scoping Important 2, 3 & 5 Marks Questions With Answers (Book Back and Creative)

12th Standard

Computer Science

Total Marks : 75

### 2 Marks

10 x 2 = 20

1) What is a scope?

**Answer :** Scope refers to the visibility of variables, parameters and functions in one part of a program to another part of the same program.

2) Why scope should be used for variable. State the reason.

**Answer :** Scope should be used for a variable because; it limits a variables scope to a single definition. That is the variables are visible only to that part of the code. Essentially, variables are addresses to an object in memory. When you assign a variable with := to an instance (object), you're binding (or mapping) the variable to that instance. Multiple variables can be mapped to the same instance.

3) What is Mapping?

**Answer :** The process of binding a variable name with an object is called mapping = (equal to sign) is used in programming languages to map the variable and object.

4) What do you mean by Namespaces?

**Answer :** Namespaces are containers for mapping names of variables to objects.

**Example :** a := 5

Here the variable 'a' is mapped to the value '5'.

5) How Python represents the private and protected Access specifiers?

**Answer :** Python prescribes a convention of prefixing the name of the variable or method with single or double underscore to emulate the behavior of protected and private access specifiers.

**Example:** self\_n2 = n2

6) How the mappings are done?

**Answer :** The words are mapped with its meaning with objects (name:= object) in programming language This allows access to objects by names we choose to assign to them.

7) What is nested function?

**Answer :** All programming languages permit functions to be nested. A Function within another function is called nested function.

8) How the access control technique is performed in Oop's language?

**Answer :** In object oriented programming languages it is implemented through access modifiers. Classical object oriented language, such as c++ and Java, control the access to class members by public, private and protected keywords.

9) How the protected members of a class are accessed?

**Answer :** Protected members of a class are accessible from within the class and are also available to its sub-classes.

10) What is pair? Give an example.

**Answer :** The duration for which a variable is alive is called life time.

### 3 Marks

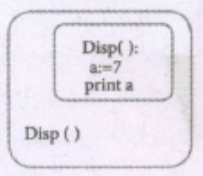
10 x 3 = 30

11) Define Local scope with an example.

**Answer : (i)** Local scope refers to variables defined in current function. Always, a function will first look up for a variable name in its local scope.

**(ii)** Only if it does not find it there, the outer scopes are checked.

**(iii)** Look at this example:

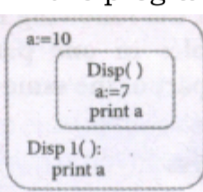
1. Disp(): 2. a:=7 3. print a 4. Disp()	Entire program 	Output of the Program 7
--	---	----------------------------

**(iv)** On execution of the above code the variable **a** displays the value 7, because it is defined and available in the local scope.

12) Define Global scope with an example.

**Answer : (i)** A variable which is declared outside of all the functions in a program is known as Global variable.

**(ii)** This means, global variable can be accessed inside or outside of all the functions in a program. Consider the following example

1. a:=10 2. Disp(): 3. a:=7 4. print a 5. Disp() 6. print a	Entire program 	Output of the program 7 10
--	--	-------------------------------

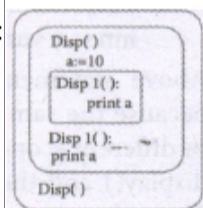
**(iii)** On execution of the above code the variable **a** which is defined inside the function displays the value 7 for the function call Disp() and then it displays 10, because **a** is defined in global scope.

13) Define Enclosed scope with an example.

**Answer : (i)** All programming languages permit functions to be nested. A function (method) within another function is called nested function.

**(ii)** A variable which is declared, inside a function which contains another function definition with in it, the inner function can also access the variable of the outer function. This scope is called enclosed scope.

**(iii)** When a compiler or interpreter search for a variable in a program, it first searches Local, and then searches Enclosing scopes. Consider the following example

1. Disp(); 2. a:=10 3. Disp1(): 4. print a 5. Disp1() 6. print a 7. Disp()	Entire program 	Output of the program 10 10
--	---	-----------------------------------

14) Why access control is required?

**Answer : (i)** Access control is a security technique that regulates who or what can view or use resources in a computing environment.

**(ii)** It is a fundamental concept in security that minimizes risk to the object.

**(iii)** In other words access control is a selective restriction of access to data.

**(iv)** In object oriented programming languages is implemented through access modifiers.

15) Identify the scope of the variables in the following pseudo code and write its output

```
color:= 'Red'
mycolor( ):
b:='Blue'
myfavcolor( ):
g:='Green'
printcolor, b, g
myfavcolor( )
printcolor, b
mycolor( )
print color
```

**Answer : Output:**

Red, Blue, Green

Red, Blue

Red.

**Scope f Variables :**

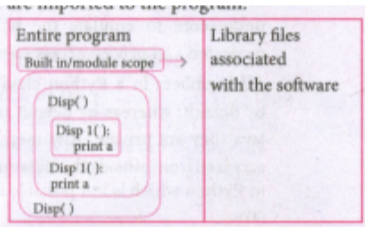
Variables	Scope
Color:=Red	Global
b:=Blue	Enclosed
G:=Green	Local

16) How the changes inside the function can't affect the variable on the outside of the function in unexpected ways?

- Answer :** (i) Every variable defined in a program has global scope.  
(ii) Once defined, every part of your program can access that variable. But it is a good practice to limit a variable's scope to a single definition.  
(iii) This way, changes inside the function can't affect the variable on the outside of the function in unexpected ways.

17) Write a note on built-in scope.

- Answer :** (i) Built-in scope is the widest scope. The built -in scope has all the names that are pre-loaded into the program scope when we start the compiler or interpreter.  
(ii) Any variable or module which is defined in the library functions of a programming language has Built-in or module scope. They are loaded as soon as the library files are imported to the program.



(iii) Normally only Functions or modules come along with the software, as packages, therefore they will come under Built in scope.

18) Write a note on module.

- Answer :** (i) A module is a part of a program. Programs are composed of one or more independently developed modules. A single module can contain one or several statements closely related each other.  
(ii) Modules work perfectly on individual level and can be integrated with other modules. A software program can be divided into modules to ease the job of programming and debugging as well.  
(iii) A program can be divided into small functional modules that work together to get the output. The process of subdividing a computer program into separate subprograms is called Modular programming.  
(iv) Modular programming enables programmers to divide up the work and debug pieces of the program independently. The examples of modules are procedures, subroutines, and functions.

19) Write a short note on types of variable scope.

- Answer :** (i) Public members (generally methods declared in a class) are accessible from outside the class.  
(ii) A variable which is declared outside of all the functions in a program is known as global variable.  
(iii) A variable which is declared inside a function which contains another function definition with in it, the inner function can also access the variable of the outer function. This scope is called enclosed scope.  
(iv) Built-in scope the widest scope has all the names that are pre-loaded into program scope when we. start the compiler or interpreter.

20) Write short notes on access specifiers in python language?

- Answer :** Python doesn't have any mechanism that effectively restricts access to any instance variable or method. Python prescribes a convention of prefixing the name of the variable or method with single or double underscore to emulate the behaviour of protected and private access Specifiers.  
All members in a python class are public by default, whereas by default in C++ and Java they are private. Any member can be accessed from outside the class environment in python which is not possible in C++ and java.

**5 Marks**

5 x 5 = 25

21) Explain the types of scopes for variable or LEGB rule with example.

**Answer : Types of Variable Scope:**

There are 4 types of Variable Scope, let's discuss them one by one:

**Local Scope:**

(i) Local scope refers to variables defined in current function. Always, a function will first look up for a variable name in its local scope. Only if it does not find it there, the outer scopes are checked.

**Look at this example**

1. Disp():	Entire program	
2. a:=7		Output of the Program
3. print a		7
4. Disp()		

(ii) On execution of the above code the variable a displays the value 7, because it is defined and available in the local scope.

**Global Scope:**

(i) A variable which is declared outside of all the functions in a program is known as global variable.

(ii) This means, global variable can be accessed inside or outside of all the functions in a program.

**Example:**

1. a:=10		Output of the Program
2. Disp():		7
3. a:=7		
4. print a		10
5. Disp()		
6. print a		

(iii) On execution of the above code the variable 'a' which is defined inside the function displays the value 7 for the function call Disp() and then it displays 10; because a is defined in global scope.

**Enclosed Scope:**

(i) All programming languages permit functions to be nested. A function (method) with in another function is called nested function.

(ii) A variable which is declared inside a function which contains another function definition with in it, the inner function can also access the variable of the outer function. This scope is called enclosed scope.

(iii) When a compiler or interpreter search for a variable in a program, it first search Local, and then search Enclosing scopes. **Consider**

**the following example:**

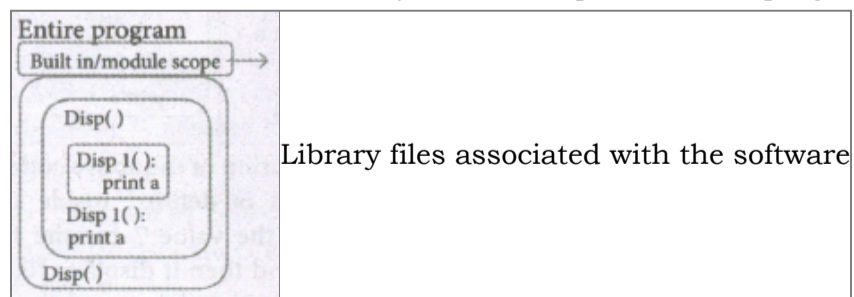
1. Disp():		Output of the Program
2. a:=10		10
3. Disp1()		10
4. print a		
5. Disp1()		
6. print a		
7. Disp()		

(iv) In the above example Disp1() is defined with in Disp(). The variable 'a' defined in Disp() can be even used by Disp1() because it is also a member of Disp().

**Built-in Scope:**

(i) The built-in scope has all the names that are pre-loaded into the program scope when we start the compiler or interpreter.

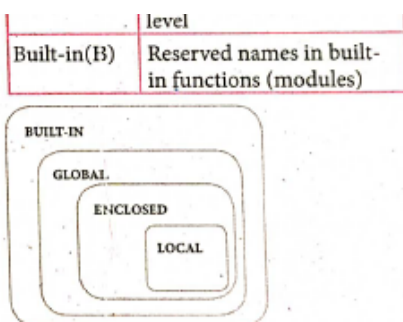
(ii) Any variable or function which is defined in the modules of a programming language has Built-in or module scope. They are loaded as soon as the library files are imported to the program. Consider the following example.



**LEGB rule :**

The **LEGB** rule is used to decide the order in which the scopes are to be searched for scope resolution. The scopes are listed below in terms of hierarchy (highest to lowest).

Local(L)	Defined inside function/class
Enclosed(E)	Defined inside enclosing functions (Nested function concept)
Global(G)	Defined at the uppermost



22) Write any Five Characteristics of Modules.

**Answer :** The following are the desirable characteristics of a module.

- (i) Modules contain instructions, processing logic, and data.
- (ii) Modules can be separately compiled and stored in a library.
- (iii) Modules can be included in a program.
- (iv) Module segments can be used by invoking a name and some parameters.
- (v) Module segments can be used by other modules.

23) Write any five benefits in using modular programming.

**Answer :** (i) Less code to be written.

- (ii) A single procedure can be developed for reuse, eliminating the need to retype the code many times.
- (iii) Programs can be designed more easily because a small team deals with only a small part of the entire code.
- (iv) Modular programming allows many programmers to collaborate on the same application.
- (v) The code is stored across multiple files.
- (vi) Code is short, simple and easy to understand.
- (vii) Errors can easily be identified, as they are localized to a subroutine or function.
- (viii) The same code can be used in many applications.
- (ix) The scoping of variables can easily be controlled.

24) Explain in detail about LEGB rule with an example?

**Answer :** Scope also defines the order in which variables have to be mapped to the object in order to obtain the value.

**Example:**

1. x:= 'outer x variable'
2. display ():
3. x:= 'inner x variable'
4. print x
5. display ()

When the above statements are executed the statements 4 and 5 display the result as

**Output:**

Outer x variable  
inner x variable

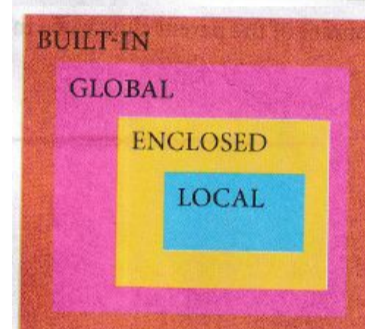
Above statements give different outputs because the same variable name x resides in different scopes, one inside the function display () and the other in the upper level.

The value 'outer x variable' is printed When x is referenced outside the function definition. Whereas when display() gets executed, 'inner x variable' is printed which is the x value inside the function definition.

A rule is followed in order to decide from which scope a variable has to be picked.

The LEGB rule is used to decide the order in which the scopes are to be searched for scope resolution. The scopes are listed below in terms of hierarchy.

Local (L)	Defined inside function/class.
Enclosed(E)	Defined inside enclosing functions.
Global(G)	Defined at the uppermost level.
Built-in (B)	Reserved names in built-in functions.



- 25) Explain in detail about modular programming and list out the benefits of modular programming.

**Answer :** A module is a part of a program, Programs are composed of one or more independently developed modules. A single module can contain one or several statements closely related to each other. Modules work perfectly on an individual level and can be integrated with other modules.

A software program can be divided into modules to ease the job of programming and debugging as well. A program can be divided into small functional modules that work together to get the output.

The process of subdividing a computer program into separate sub-programs is called modular programming.

Modular programming enables programmers to divide up the work and debug pieces of the program independently. The examples of modules are procedures, subroutines, and functions.

**Benefits:**

- (i) Less code to be written.
- (ii) The code is stored across multiple files.
- (iii) Code is short, simple and easy to understand.
- (iv) Errors can easily be identified, as they are localized to a subroutine or function.
- (v) The same code can be used in many applications.
- (vi) The scoping of variables can easily be controlled.