

# ANSWER BANK

Friday, 22 June 2012

Answers for 2 mark questions in C++

## ANSWER KEY

**1.NAME ANY FOUR KEY WORDS IN IN C++.**

**ANS:** The key words in c++ are

- a)return
- b)bool
- c)switch
- d)case

(for more keywords log on to this link

<http://en.cppreference.com/w/cpp/keyword>)

**2.WHAT ARE MANUPULATORS?**

**ANS:** Manipulators are functions specifically designed to be used in conjunction with the insertion (<<) and extraction (>>) operators on stream objects, for example:

```
cout << boolalpha;
```

They are still regular functions and can also be called as any other function using a stream object as argument, for example:

```
boolalpha (cout);
```

Manipulators are used to change formatting parameters on streams and to insert or extract certain special characters.

(Ref-<http://www.cplusplus.com/reference/iostream/manipulators/>)

**3.WHAT IS MEANT BY DATA ENCAPSULATION?**

**ANS:** Encapsulation is the process of combining data and functions into a single unit called class. Using the method of encapsulation, the programmer cannot directly access the data. Data is only accessible through the functions present inside the class. Data encapsulation led to the important concept of data hiding.

(REF-<http://in.answers.yahoo.com/question/index?qid=20100906203637AApAxQi>)

**4. DEFINE DYNAMIC BINDING.**

**DYNAMIC BINDING:** In C++ you can have an array of base classes but you can call different functions of the derived class by assigning the derived class objects addresses by using virtual functions--this is dynamic binding. If the methods are virtual then this is dynamic binding. The name is known at compile time but the method called cannot be determined without knowing the runtime object type.

(REF-<http://in.answers.yahoo.com/question/index?qid=20110621072544AAgbMui>) or (<http://www.learncpp.com/cpp-tutorial/124-early-binding-and-late-binding/>)

**5. Define Pointers.**

**ANS: Pointers:** A pointer is a variable that is used to store a memory address. The address is the location of the variable in the memory. Pointers help in allocating memory dynamically. Pointers improve execution time and saves space. Pointer points to a particular data type. The general form of declaring pointer is:-

```
type *variable_name;
```

type is the base type of the pointer and variable\_name is the name of the variable of the pointer. For example,

```
int *x;
```

x is the variable name and it is the pointer of type integer.

(Ref-[http://cpp-tutorial.cpp4u.com/compound\\_pointers.html](http://cpp-tutorial.cpp4u.com/compound_pointers.html))

**6. WHAT ARE THE RULES FOR FORMING IDENTIFIERS IN C++? EXPLAIN WITH EXAMPLE.**

**ANS:** Rules for Forming Identifiers

Sign In

## Blog Archive

▼ 2012 (2)

▼ June (2)

QUESTION BANK:  
ANSWERS FOR  
5MARK  
QUESTIONS  
(C++ B....

Answers for 2  
mark questions  
in C++

Total Pageviews

~~1309~~

Identifiers are defined according to the following rules:

1. It consists of letters and digits.
2. First character must be an alphabet or underscore.
3. Both upper and lower cases are allowed. Same text of different case is not equivalent, for example: TEXT is not same as text.
4. Except the special character underscore ( \_ ), no other special symbols can be used.
5. For example, some valid identifiers are shown below :

Z  
Z789  
\_BY  
reset  
value\_rate

For example, some invalid identifiers are shown below :

456                                      First character to be alphabet.  
"Y."                                      Not allowed.  
sheet-no                                  Hyphen  
Same thing                                  Blankspace allowed.  
(ref-<http://clinuxpro.com/identifiers-in-c>)

## 7. LIST OUT FOUR FEATURES OF OOPS.

**ANS:** Features of Object oriented Programming

The Objects Oriented programming language supports all the features of normal programming languages. In addition it supports some important concepts and terminology which has made it popular among programming methodology.

The important features of Object Oriented programming are:

Inheritance : Ability to get other class features

Polymorphism :Acquiring more than one form

Data Hiding : Hide complexity and gives necessary details.

Encapsulation : Hide complex part of program

Overloading : Can use object with all arithmetic operator.

Reusability : One class can be use again and again

## 8. WHAT ARE TOKENS?

**ANS:** A token is the smallest element of a C++ program that is meaningful to the compiler. The C++ parser recognizes these kinds of tokens: identifiers, keywords, literals, operators, punctuators, and other separators. A stream of these tokens makes up a translation unit.

(REF-[http://msdn.microsoft.com/en-us/library/3yx2xe3h\(v=vs.80\).aspx](http://msdn.microsoft.com/en-us/library/3yx2xe3h(v=vs.80).aspx))

## 9. WRITE A NOTE ON SCOPE RESOLUTION OPERATORS.

**ANS:** The **scope resolution operator** (denoted ::) in C++ is used to define the already declared **member functions** (in the header file with the .hpp or the .h extension) of a particular class. In the .cpp file one can define the usual global functions or the member functions of the class. To differentiate between the normal functions and the member functions of the class, one needs to use the scope resolution operator (::) in between the class name and the member function name i.e. ship::foo() where ship is a class and foo() is a member function of the class ship.

(REF-[http://en.wikipedia.org/wiki/Scope\\_resolution\\_operator](http://en.wikipedia.org/wiki/Scope_resolution_operator))

## 10. HOW TO DECLARE A CLASS IN C++?

Classes are generally declared using the keyword `class`, with the following format:

```
class class_name {
    access_specifier_1:
        member1;
    access_specifier_2:
        member2;
    ...
}
```

```
} object_names;
```

Where `class_name` is a valid identifier for the class, `object_names` is an optional list of names for objects of this class. The body of the declaration can contain members, that can be either data or function declarations, and optionally access specifiers.

(REF-<http://www.cplusplus.com/doc/tutorial/classes/>)

### 11. WHAT ARE OBJECTS? GIVE EXAMPLE.

**ANS:** An object is a component of a program that knows how to perform certain actions and to interact with other pieces of the program. Functions have previously been described as "black boxes" that take an input and spit out an output. Objects can be thought of as "smart" black boxes. That is, objects can know how to do more than one specific task, and they can store their own set of data. Designing a program with objects allows a programmer to model the program after the real world. A program can be broken down into specific parts, and each of these parts can perform fairly simple tasks. When all of these simple pieces are meshed together into a program, it can produce a very complicated and useful application

(REF-[http://www.intap.net/~drw/cpp/cpp06\\_01.htm](http://www.intap.net/~drw/cpp/cpp06_01.htm))

### 12. WHAT ARE IDENTIFIERS IN C++? GIVE EXAMPLE.

**ANS:** An identifier is a sequence of characters used to denote one of the following:

- Object or variable name
- Class, structure, or union name
- Enumerated type name
- Member of a class, structure, union, or enumeration
- Function or class-member function
- typedef name
- Label name
- Macro name
- Macro parameter

(OR)

Identifiers provide names for the following language elements:

- Functions
- Objects
- Labels
- Function parameters
- Macros and macro parameters
- Type definitions
- Enumerated types and enumerators
- Structure and union names
- Classes and class members
- Templates
- Template parameters
- Namespaces

An identifier consists of an arbitrary number of letters, digits, or the underscore character in the form:

```

[ ] [ ] .-----
      v      |
>>--+-letter+-----+letter+--+-----
-----><
      '-_-----'    +-digit--+
                  '-_-----'
```

(REF-<http://msdn.microsoft.com/en-us/library/565w213d.aspx> AND

<http://publib.boulder.ibm.com/infocenter/lnxpcmp/v8v101/index.jsp?topic=/%2Fcom.ibm.xlcpp8l.doc%2Flanguage%2Fref%2Fident.htm> )

### 13. DEFINE CLASS.

**ANS:** A *class* is a mechanism for creating user-defined data types. It is similar to the C language structure data type. In C, a structure is composed of a set of data members. In C++, a class type is like a C structure, except that a class is composed of a set of data members and a set of operations that can be performed on the class.

(REF-<http://publib.boulder.ibm.com/infocenter/lnxpcmp/v8v101/index.jsp?topic=%2Fcom.ibm.xlcpp8l.doc%2Flanguage%2Fref%2Fident.htm>)

#### 14. DEFINE VARIABLE.

**ANS:** Variables are a way of reserving memory to hold some data and assign names to them so that we don't have to remember the numbers like 46735 and instead we can use the memory location by simply referring to the variable. Every variable is mapped to a unique memory address. For example, we have 3 variable v1, v2, v3. They may be assigned the memory addresses 32000, 12456, 67893 respectively.

(REF-[http://cpp-tutorial.cpp4u.com/basics\\_variables.html](http://cpp-tutorial.cpp4u.com/basics_variables.html))

#### 15. WHAT ARE FORMAL PARAMETERS?

**ANS:** Formal parameters are written in the function prototype and function header of the definition. Formal parameters are local variables which are assigned values from the arguments when the function is called.

(REF-<http://answers.yahoo.com/question/index?qid=20110513042617AAZyWF>)

#### 16. WHAT IS AN ARRAY?

**ANS:** An array is a series of elements of the same type placed in contiguous memory locations that can be individually referenced by adding an index to a unique identifier.

That means that, for example, we can store 5 values of type `int` in an array without having to declare 5 different variables, each one with a different identifier. Instead of that, using an array we can store 5 different values of the same type, `int` for example, with a unique identifier.

(ref-<http://www.cplusplus.com/doc/tutorial/arrays/>)

#### 17. DEFINE FUNCTION PROTOTYPE.

**ANS:** One of the most important features of C++ is the function prototypes. A function prototype tells the compiler the name of the function, the type of data returned by the function, the number of parameters the function expects to receive, the types of the parameters, and the order in which these parameters are expected. The compiler use function prototypes to validate function calls. Early versions of C did not perform this kind of checking, so it was possible to call functions improperly without the compiler detecting the errors. Such calls could result in fatal execution-time errors or nonfatal fatal errors that caused, difficult to detect logic errors. Function prototypes correct this deficiency.

The function prototype for maximum in this program is

```
int maximum( int, int, int);
```

(REF-<http://www.codeproject.com/Articles/3725/Function-prototypes>)

#### 18. LIST ANY TWO MANUPULATORS IN C++.

<b>ANS: Independent flags (switch on):</b>	
<b>boolalpha</b>	Alphanumerical bool values (manipulator function)
<b>showbase</b>	Show numerical base prefixes (manipulator function)
<b>showpoint</b>	Show decimal point (manipulator function)
<b>showpos</b>	Show positive signs (manipulator function)
<b>skipws</b>	Skip whitespaces (manipulator function)
<b>unitbuf</b>	Flush buffer after insertions (manipulator function)
<b>uppercase</b>	Generate upper-case letters (manipulator function)
<b>Independent flags (switch off):</b>	
<b>nboolalpha</b>	No alphanumerical bool values (manipulator function)
<b>nshowbase</b>	Do not show numerical base prefixes (manipulator function)
<b>nshowpoint</b>	Do not show decimal point (manipulator function)
<b>nshowpos</b>	Do not show positive signs (manipulator function)
<b>noskipws</b>	Do not skip whitespaces (manipulator function)

<b>nunitbuf</b>	Do not force flushes after insertions (manipulator function)
<b>nouppercase</b>	Do not generate upper case letters (manipulator function)
<b>Numerical base format flags ("basefield" flags):</b>	
<b>dec</b>	Use decimal base (manipulator function)
<b>hex</b>	Use hexadecimal base (manipulator function)
<b>oct</b>	Use octal base (manipulator function)
<b>Floating-point format flags ("floatfield" flags):</b>	
<b>fixed</b>	Use fixed-point notation (manipulator function )
<b>scientific</b>	Use scientific notation (manipulator function)
<b>Adjustment format flags ("adjustfield" flags):</b>	
<b>internal</b>	Adjust field by inserting characters at an internal position (manipulator function)
<b>left</b>	Adjust output to the left (manipulator function)
<b>right</b>	Adjust output to the right (manipulator function)

## Input manipulators

<b>ws</b>	Extract whitespaces (manipulator function)
-----------	--

## Output manipulators

<b>endl</b>	Insert newline and flush (manipulator function)
<b>ends</b>	Insert null character (manipulator function)
<b>flush</b>	Flush stream buffer (manipulator function)

## Parameterized manipulators

These functions take parameters when used as manipulators. They require the explicit inclusion of the header file `<iomanip>`.

<b>setiosflags</b>	Set format flags (manipulator function)
<b>resetiosflags</b>	Reset format flags (manipulator function)
<b>setbase</b>	Set basefield flag (manipulator function)
<b>setfill</b>	Set fill character (manipulator function)
<b>setprecision</b>	Set decimal precision (manipulator function)
<b>setw</b>	Set field width (manipulator function)

(REF-<http://www.cplusplus.com/reference/iostream/manipulators/>)

ALL THE BEST

BY VAISHALI.R,  
[www.tnqbank.blogspot.com](http://www.tnqbank.blogspot.com)

Posted by Vaishali R at 09:59



No comments:

Post a Comment

Enter your comment...

Comment as:

Google Account ▼

Publish

Preview

[Newer Post](#)

[Home](#)

[Subscribe to: Post Comments \(Atom\)](#)

Simple template. Template images by [lucato](#). Powered by [Blogger](#).