Chapter 1
Review of C++ Programming

Basics of C++	
Character set	Fundamental unit of C++ language. Classified into letters, digits, special characters and white spaces.
Tokens	Building blocks of C++ programs. Classified into keywords, identifiers, literals, punctuators and operators.
Keywords	Reserved words that convey specific meaning to the language compiler.
Identifiers	User-defined words to identify memory locations, statements, etc. Identifiers include variables, labels, function names, etc.
Literals (Constants)	Tokens that do not change their value during the program run. Classified into integer constants, floating point constants, character constants and string constants.
Operators	Symbols that represents some operations. They consist of arithmetic, relational and logical operators. There are some special operators named *get from* (>>), *put to* (<<) and assignment (=). Another category of operators include increment (++), decrement (--) and arithmetic assignment (+=, -=, *=, /=, %=) operators.
Punctuators	Special characters like comma (,), semi colon (;), etc. used for the perfection of syntax of various constructs of the language.
Data types	These are means to identify the type of data and associated operations. Data types are classified into fundamental and user-defined data types. Fundamental data types include `int`, `char`, `float`, `double` and `void`.
Type modifiers	The keyword `signed`, `unsigned`, `short` and `long` are the type modifiers. They are used with data types to modify the size and range of data.
Expressions	Operators and operands are combined to form expressions. Classified into arithmetic expressions, relational expressions and logical expressions. Arithmetic expression is divided into integer expression and real expression. Integer expression - Only integer data and integer results. Real expressions – Only floating point data and float type result. Relational and logical expressions return True or False as outputs.
Type conversion	The process of converting the current data type of a value into another type. Two types - implicit and explicit conversions. In implicit type conversion, complier is responsible for the conversion. Also known as type promotion. In explicit conversion, user is responsible for the conversion. Also known as type casting.
Statements	Declaration statement (eg: int a; float b;), Variable initialization statement (eg: int a=5; const float pi=3.14;) Input statement (eg: cin>>a;), Output statement (cout<<a; cout<<"hello";), Assignment statement (eg: a=5; a=b+c;)

Increment and Decrement operators	a++; (postfix form) ++a; (prefix form) : Same as a = a + 1;
	a--; (postfix form) --a; (prefix form) : Same as a = a – 1;

Arithmetic assignment operators	num+=10; Same as num = num + 1; p –= q;
	Same as p = p – q;
	m*=n; (m = m * n;) n/=10; (n = n/10;) n%=2; (n = n %2;)

Structure of C++ program	Three parts – preprocessor directive (#include statement), using namespace statement, main() function.
Control statements	Two types – Selection statements (if, switch) Looping statements (while, for, do – while) while and for are entry controlled loops do – while is exit controlled loop
Looping statements	There will be four components – initialization expression, test expression, update expression, loop-body. In the case of entry-controlled loop, body will be executed only after evaluation the test expression (condition). But, in the case of the exit-controlled loop, condition will be checked only after executing the loop-body.
Conditional Operator (? :)	Ternary operator. Used instead of if – else statement. Eg: The statement if (n%2==) cout<< "Even" ; else cout<< "Odd"; can be replaced by (n%2==0) ? cout<< "Even" : cout<< "Odd";
Nested loop	If the body of a loop contains another loop, it is called nested loop.

Nested loop example:

```
for  (i=1;  i<=5;  i++)
{
        cout  <<  "\n";
        for  (j=1;  j<=i;  j++)
                Cout<<  j  <<  "\t";
}
```

```
The output
1
1       2
1       2       3
1       2       3       4
1       2       3       4       5
```

switch V/s if – else if statement

switch	if – else if
• Only equality conditions are checked.	• Any relational expression can be used for conditions.
• Program control goes outside the block only if break is used after each case.	• No need of break to take the control outside after executing a block.
• default case is for an action where all the conditions fail.	• else is used for an action where all the conditions fail.

Jump statements

Transfers program control from one place to another part of a program. There are four jump statements – `return`, `goto`, `break` and `continue`.

The **`goto`** statement can transfer the program control to a labeled statement.

When a **`break`** statement is encountered in a loop (`for`, `while`, `do-while`), it takes the program control outside the loop.

The **continue** statement is used for skipping the remaining statements within the loop-body and forcing the next iteration.

break V/s continue

break	continue
• Used with switch and loops. • Takes the control outside the loop by skipping the remaining part of the body. • Program control goes outside the loop even though the test expression returns true.	• Used only with loops. • Takes the control to the beginning of the loop by skipping the remaining part of the body. • Program control goes outside only when the test expression of the loop returns false.

Questions from Previous Years' Question Papers

1. Which among the following is an insertion operator?
   (a) << (b) >> (c) < (d) > (1) (March 2016)
2. What are the main components of a looping statement? (2) (March 2016)
3. How do break and continue differ in a loop. Explain with an example. (3) (March 2016)
4. _____is an exit-controlled loop. (1) (SAY 2016)
5. Explain switch statement with an example. (3) (SAY 2016)
6. How does a 'goto' statement work? (2) (SAY 2016)
7. Which among the following statement is equivalent to the statement series
   b = a; a = a+ 1; ?
   (a) b += a; (b) b = a++; (c) b = ++a; (d) b += a + b; (1) (March 2017)
8. Rewrite the following C++ code using 'switch' statement:

```
cin>>pcode;
if (pcode == 'C')
     cout<< "Computer";
else if (pcode == 'M')
     cout<< "Mobile Phone";
else if (pcode == 'L')
     cout<< "Laptop";
else
     cout<< "Invalid code";
```

   (3) (March 2017)

9. A_____statement in a loop forces the termination of that loop. (1) (March 2017)
10. Identify the following C++ tokens:
    (a) "welcome" (b) int (c) >= (d) ++ (2) (SAY2017)
11. ______operator is the arithmetic assignment operator.
    (a) >> (b) = = (c) += (d) = (1) (SAY 2017)
12. Explain break and continue statements with example. (3) (SAY 2017)