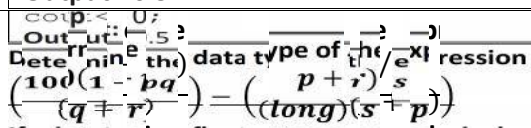


**CHAPTER-1**  
**C++ REVISION TOUR**  
**VERY SHORT/ SHORT ANSWER QUESTIONS**

<b>1.</b>	<b>Find out the errors, if any, in the following C++ statement:</b> (i) <code>cout&lt;&lt;"=a;</code> (ii) <code>m=5, n=12; o=15</code> (iii) <code>cout&lt;&lt;"x";&lt;&lt;x;</code> (iv) <code>cin&gt;&gt;y;&gt;&gt;j;</code> (v) <code>cin&gt;&gt;"\n"&gt;&gt;y;</code> (vi) <code>cout&gt;&gt;"\n "abc";</code> (vii) <code>a = b + c</code> (viii) <code>break = x*y;</code>	
<b>Ans.</b>	(i) <code>cout&lt;&lt;"=a;</code> (ii) <code>m=5, n=12; o=15</code> (iii) <code>cout&lt;&lt;"x";&lt;&lt;x;</code> (iv) <code>cin&gt;&gt;y;&gt;&gt;j;</code> (v) <code>cin&gt;&gt;"\n"&gt;&gt;y;</code> (vi) <code>cout&gt;&gt;"\n _abc";</code> (vii) <code>a = b + c</code> (viii) <code>break = x*y;</code>	
<b>2.</b>	<b>What is the difference between fundamental data types and derived data types? Explain with examples.</b>	
<b>Ans.</b>	<b>Fundamental data types</b>	<b>Derived data types</b>
	These are the data types that are not composed of any other data type.	These are the data types that are composed of fundamental data types.
	There are five fundamental data types: char, int, float, double and void.	These are: array, function, pointer, reference, constant, class, structure, union and enumeration.
	Example: <code>int a=10; float b;</code>	Example: <code>float marks[50]; Const int a=5;</code>
<b>3.</b>	<b>What is the purpose of a header file in a program?</b>	
<b>Ans.</b>	Header files provide function prototypes, definitions of library functions, declaration of data types and constants used with the library functions.	
<b>4.</b>	<b>What main integer types are offered by C++?</b>	
<b>Ans.</b>	C++ offered three types of integers : short, int and long. Each comes in both signed and unsigned versions.	
<b>5.</b>	<b>Explain the usage of following with the help of an example:</b> (i) constant      (ii) reference (iii) variable      (iv) union	
<b>Ans.</b>	<p><b>(i) constant:</b> The keyword <code>const</code> can be added to the declaration of an object to make that object a constant rather than a variable. Thus, the value of the named constant cannot be altered during the program run. The general form of constant declaration is as follows: <b>const type name = value.</b> Example: <code>const int a=10;</code></p> <p><b>(ii) reference:</b> A reference is an alternative name for an object. A reference variable provides an alias for a previously defined variable. The general form of declaring a reference variable is: <b>Type &amp;ref-var = var-name;</b> where <i>type</i> is any valid c++ data type, <i>ref-var</i> is the name of reference variable that will point to variable denoted by <i>var-name</i>.</p> <p>Example:</p> <pre>int total; int &amp;sum=total; total=100; cout&lt;&lt;"Sum="&lt;&lt;sum&lt;&lt;"\n" ; cout&lt;&lt;"Total="&lt;&lt;total&lt;&lt;"\n" ;</pre> <p>In above code both the variables refer to the same data object in the memory, thus, print the same value.</p>	

	<p><b>(iii) variable:</b> Variables represent named storage locations whose value can be manipulated during program run. Variables are generally declared as: <b>type name;</b> where <i>type</i> is any C++ data type and <i>name</i> is the variable name. A variable can be initialized either by a separate assignment statement following its declaration as shown below:</p> <pre>int age;    age = 10;</pre> <p><b>(iv) union:</b> A union is a memory location that is shared by two or more different variables, generally of different types at different times. Defining a union is similar to defining a structure. Following of declaration declares a <i>union share</i> having two variables and creates a union object <i>cnvt</i> of union type <i>share</i>:</p> <pre>union share{                 int i;                 char ch;             }; union share cnvt;</pre> <p>union can referred to the data stored in <i>cnvt</i> as either an integer or character. To assign the integer 20 to element <i>i</i> of <i>cnvt</i>, write - <code>cnvt.i = 20;</code> To print the value of element <i>ch</i> of <i>cnvt</i>, write - <code>cout&lt;&lt;cnvt.ch;</code></p>
<b>6.</b>	<b>How many ways can a variable be initialized into? Give examples for each type of initialization.</b>
<b>Ans.</b>	<p>A variable can be initialized into two ways as following:</p> <p><b>(i) By separate assignment statement:</b> Example: <code>int age;</code> <code>age = 10;</code></p> <p><b>(ii) At the time of declaration:</b> Example: <code>int age = 10;</code></p> <p><b>(iii) Dynamic initialization:</b> Example: <code>float avg = sum/count;</code></p>
<b>7.</b>	<b>How are the following two statements different?</b>
	<pre>char pcode = 75; char pcode = 'K';</pre>
<b>Ans.</b>	The first statement treats 75 as ascii value and converts it into its relative character 'K' whereas second statement stores character 'K' and does not make any conversion.
<b>8.</b>	<b>How are the following two statements different?</b>
	<pre>char pcode = 75;    short pcode = 75;</pre>
<b>Ans.</b>	The first statement treats 75 as ascii value and converts it into its relative character 'K' whereas second statement treats 75 as number.
<b>9.</b>	<b>If value is an identifier of int type and is holding value 200, is the following statement correct?</b>
	<pre>char code = value</pre>
<b>Ans.</b>	VALID Statement
<b>10.</b>	<b>The data type double is another floating-point type. Then why is it treated as a distinct data type?</b>
<b>Ans.</b>	The data type double is treated as a distinct data type because it occupies twice as much memory as type float, and stores floating-point numbers with much larger range and precision. It stands for double precision floating-point. It is used when type float is too small or insufficiently precise.
<b>11.</b>	<b>Explain the impact of access modifier const over variables. Support your answer with examples.</b>
<b>Ans.</b>	<p>The access modifier const can be added to the declaration of an object to make that object a constant rather than a variable. Thus, the value of the named constant cannot be altered during the program run whereas the value of the variable can be changed during the program run.</p> <p><b>Example:</b></p> <pre>void main(){     const int a=10;     int b=20;     a++;     cout&lt;&lt;"a="&lt;&lt;a;     b++;     cout&lt;&lt;"b="&lt;&lt;b; }</pre>
<b>12.</b>	<b>What are arithmetic operators in C++? Distinguish between unary and binary arithmetic operators. Give examples for each of them.</b>

Ans.	<b>Following are the arithmetic operators in C++: addition(+), subtraction(-), multiplication(*), division(/) and reminder(%).</b>									
	<table border="1"> <thead> <tr> <th data-bbox="138 220 706 262">Unary arithmetic operators</th> </tr> </thead> <tbody> <tr> <td data-bbox="138 262 706 304">Unary Operators has only one operand</td> </tr> <tr> <td data-bbox="138 304 706 346">The Unary Operators are ++,--,&amp;*,+, - etc.</td> </tr> <tr> <td data-bbox="138 346 706 441">Example:     short x = 987;               x = -x;</td> </tr> </tbody> </table>	Unary arithmetic operators	Unary Operators has only one operand	The Unary Operators are ++,--,&*,+, - etc.	Example:     short x = 987; x = -x;	<table border="1"> <thead> <tr> <th data-bbox="706 220 1477 262">Binary arithmetic operators</th> </tr> </thead> <tbody> <tr> <td data-bbox="706 262 1477 304">Binary operators ("bi" as in "two") have two operands</td> </tr> <tr> <td data-bbox="706 304 1477 346">The +, -, &amp;&amp;, &lt;&lt;, ==, &gt;&gt; etc. are binary operators.</td> </tr> <tr> <td data-bbox="706 346 1477 441">Example:     int x, y = 5, z;               z = 10;               x = y + z;</td> </tr> </tbody> </table>	Binary arithmetic operators	Binary operators ("bi" as in "two") have two operands	The +, -, &&, <<, ==, >> etc. are binary operators.	Example:     int x, y = 5, z; z = 10; x = y + z;
Unary arithmetic operators										
Unary Operators has only one operand										
The Unary Operators are ++,--,&*,+, - etc.										
Example:     short x = 987; x = -x;										
Binary arithmetic operators										
Binary operators ("bi" as in "two") have two operands										
The +, -, &&, <<, ==, >> etc. are binary operators.										
Example:     int x, y = 5, z; z = 10; x = y + z;										
13.	<b>What is the function of increment/decrement operators? How many varieties do they come in? How are these two varieties different from one another?</b>									
Ans.	The increment operator, ++ adds 1 to its operand and the decrement operator -- subtract 1 from its operands. The increment/decrement operator comes in two varieties as following: (i) Prefix version and   (ii) Postfix version:									
	<table border="1"> <thead> <tr> <th data-bbox="138 640 803 682">Prefix version</th> </tr> </thead> <tbody> <tr> <td data-bbox="138 682 803 787">Performs the increment or decrement operation before using the value of the operand.</td> </tr> <tr> <td data-bbox="138 787 803 903">Example:   sum = 10;           ctr = 5;           sum = sum + (++ctr);</td> </tr> </tbody> </table>	Prefix version	Performs the increment or decrement operation before using the value of the operand.	Example:   sum = 10; ctr = 5; sum = sum + (++ctr);	<table border="1"> <thead> <tr> <th data-bbox="803 640 1477 682">Postfix version</th> </tr> </thead> <tbody> <tr> <td data-bbox="803 682 1477 787">First uses the value of the operand in evaluating the expression before incrementing or decrementing the operand's value.</td> </tr> <tr> <td data-bbox="803 787 1477 903">Example:   sum = 10;           ctr = 5;           sum = sum + (ctr++);</td> </tr> </tbody> </table>	Postfix version	First uses the value of the operand in evaluating the expression before incrementing or decrementing the operand's value.	Example:   sum = 10; ctr = 5; sum = sum + (ctr++);		
Prefix version										
Performs the increment or decrement operation before using the value of the operand.										
Example:   sum = 10; ctr = 5; sum = sum + (++ctr);										
Postfix version										
First uses the value of the operand in evaluating the expression before incrementing or decrementing the operand's value.										
Example:   sum = 10; ctr = 5; sum = sum + (ctr++);										
14.	<b>State why are following expression invalid?</b> (i) asm = 5100    val < 35 (ii) age > 70 && < 90 (iii) income >= 500    && val < 500 (iv) res !> 20    ! X > 20									
Ans.	(i) In this expression single '=' operator is used for comparison which is not valid. (ii) In this expression variable's name is not mentioned after the '&&' operator which is not valid. (iii) In this expression both '  ' and '&&' operators are written together which is not valid. (iv) In this expression use of '!>' is invalid and !x>20 should be written in parenthesis like (!x>20).									
15.	<b>What is the difference between Type Casting and Automatic Type conversion? Also, give a suitable C++ code to illustrate both.</b>									
Ans.	<table border="1"> <thead> <tr> <th data-bbox="138 1312 803 1354">Type Casting</th> </tr> </thead> <tbody> <tr> <td data-bbox="138 1354 803 1417">It is an explicit process of conversion of a data from one type to another.</td> </tr> <tr> <td data-bbox="138 1417 803 1459">It is performed with the help of casting operator().</td> </tr> <tr> <td data-bbox="138 1459 803 1627"> <b>Example:</b>            int A=1, B=2;            float C = (float)A/B;//Type Casting            cout&lt;&lt;C;  <b>Output:</b> 0.5         </td> </tr> </tbody> </table>	Type Casting	It is an explicit process of conversion of a data from one type to another.	It is performed with the help of casting operator().	<b>Example:</b> int A=1, B=2; float C = (float)A/B;//Type Casting cout<<C; <b>Output:</b> 0.5	<table border="1"> <thead> <tr> <th data-bbox="803 1312 1477 1354">Automatic Type conversion</th> </tr> </thead> <tbody> <tr> <td data-bbox="803 1354 1477 1417">It is an implicit process of conversion of a data from one type to another.</td> </tr> <tr> <td data-bbox="803 1417 1477 1459">It is performed by compiler its own.</td> </tr> <tr> <td data-bbox="803 1459 1477 1627"> <b>Example:</b>            int N = 65;            char C = N; //Automatic type conversion            cout&lt;&lt;C;  <b>Output:</b> A         </td> </tr> </tbody> </table>	Automatic Type conversion	It is an implicit process of conversion of a data from one type to another.	It is performed by compiler its own.	<b>Example:</b> int N = 65; char C = N; //Automatic type conversion cout<<C; <b>Output:</b> A
Type Casting										
It is an explicit process of conversion of a data from one type to another.										
It is performed with the help of casting operator().										
<b>Example:</b> int A=1, B=2; float C = (float)A/B;//Type Casting cout<<C; <b>Output:</b> 0.5										
Automatic Type conversion										
It is an implicit process of conversion of a data from one type to another.										
It is performed by compiler its own.										
<b>Example:</b> int N = 65; char C = N; //Automatic type conversion cout<<C; <b>Output:</b> A										
16.	 <p><b>If p is int, r is a float, q is a long and s is double.</b></p>									
Ans.	The data type of the expression is double as double is the larger data type.									
17.	<b>What are the outputs of following two codes fragments? Justify your answer.</b> <pre> //version 1 int f = 1, i =2; while(++i &lt; 5)     f * = i;  //version 2 int f = 1, i = 2; do{     f * = i; </pre>									

	<pre>cout&lt;&lt;f; : : }while (++i &lt; 5); cout&lt;&lt; f; :</pre>		
<b>Ans.</b>	<p>The output of the first code fragment is 12. The output of the first code fragment is 24.</p>		
<b>18</b>	<p><b>Will the following program execute successfully? If not, state the reason(s):</b></p> <table border="1" style="width: 100%;"> <tr> <td style="width: 50%; vertical-align: top;"> <pre>(i) #include&lt;stdio.h&gt; void main(){     int s1,s2,num;     s1=s2=0;     for(x=0;x&lt;11;x++)     {         cin&lt;&lt;num;         if(num&gt;0)s1+=num;         else s2=/num;     }     cout&lt;&lt;s1&lt;&lt;s2; }</pre> </td> <td style="width: 50%; vertical-align: top;"> <pre>(ii) #include&lt;stdio.h&gt; void main(){     int x,sum=0;     cin&lt;&lt;n;     for(x=1;x&lt;100;x+=2)         if x%2==0             sum+=x;     cout&lt;&lt;"SUM="&gt;&gt;sum; }</pre> </td> </tr> </table>	<pre>(i) #include&lt;stdio.h&gt; void main(){     int s1,s2,num;     s1=s2=0;     for(x=0;x&lt;11;x++)     {         cin&lt;&lt;num;         if(num&gt;0)s1+=num;         else s2=/num;     }     cout&lt;&lt;s1&lt;&lt;s2; }</pre>	<pre>(ii) #include&lt;stdio.h&gt; void main(){     int x,sum=0;     cin&lt;&lt;n;     for(x=1;x&lt;100;x+=2)         if x%2==0             sum+=x;     cout&lt;&lt;"SUM="&gt;&gt;sum; }</pre>
<pre>(i) #include&lt;stdio.h&gt; void main(){     int s1,s2,num;     s1=s2=0;     for(x=0;x&lt;11;x++)     {         cin&lt;&lt;num;         if(num&gt;0)s1+=num;         else s2=/num;     }     cout&lt;&lt;s1&lt;&lt;s2; }</pre>	<pre>(ii) #include&lt;stdio.h&gt; void main(){     int x,sum=0;     cin&lt;&lt;n;     for(x=1;x&lt;100;x+=2)         if x%2==0             sum+=x;     cout&lt;&lt;"SUM="&gt;&gt;sum; }</pre>		
<b>Ans.</b>	<table border="1" style="width: 100%;"> <tr> <td style="width: 50%; vertical-align: top;"> <p>(i) Will encounter a compile time error for following reasons:</p> <ul style="list-style-type: none"> <li>• The variable 'x' is not declared.</li> <li>• With cin statement '&lt;&lt;' symbol is used instead of '&gt;&gt;'.</li> <li>• Invalid semicolon at the end of if statement.</li> </ul> </td> <td style="width: 50%; vertical-align: top;"> <p>(ii) Will encounter a compile time error for following reasons:</p> <ul style="list-style-type: none"> <li>• The variable 'n' is not declared.</li> <li>• There should be a parenthesis in if statement.</li> <li>• '&gt;&gt;' is used with cout statement instead of '&lt;&lt;'.</li> </ul> </td> </tr> </table>	<p>(i) Will encounter a compile time error for following reasons:</p> <ul style="list-style-type: none"> <li>• The variable 'x' is not declared.</li> <li>• With cin statement '&lt;&lt;' symbol is used instead of '&gt;&gt;'.</li> <li>• Invalid semicolon at the end of if statement.</li> </ul>	<p>(ii) Will encounter a compile time error for following reasons:</p> <ul style="list-style-type: none"> <li>• The variable 'n' is not declared.</li> <li>• There should be a parenthesis in if statement.</li> <li>• '&gt;&gt;' is used with cout statement instead of '&lt;&lt;'.</li> </ul>
<p>(i) Will encounter a compile time error for following reasons:</p> <ul style="list-style-type: none"> <li>• The variable 'x' is not declared.</li> <li>• With cin statement '&lt;&lt;' symbol is used instead of '&gt;&gt;'.</li> <li>• Invalid semicolon at the end of if statement.</li> </ul>	<p>(ii) Will encounter a compile time error for following reasons:</p> <ul style="list-style-type: none"> <li>• The variable 'n' is not declared.</li> <li>• There should be a parenthesis in if statement.</li> <li>• '&gt;&gt;' is used with cout statement instead of '&lt;&lt;'.</li> </ul>		
<b>19</b>	<p><b>Find the syntax error(s), if any, in the following program:</b></p>		
<b>Ans.</b>	<pre>(i) #include&lt;iostream.h&gt; main(){     int x[5],*y,z[5]     for(i=0;i&lt;=5;i++)     {         x[i]=i;         z[i]=i+3;         y=z;         x=y;     } }  (ii) #include&lt;iostream.h&gt; void main(){     int x,y;     cin&gt;&gt;x;     for(x=0;x&lt;5;++x)         cout y else cout&lt;&lt;x&lt;&lt;y; }  (iii) #include&lt;iostream.h&gt; void main(){     int R;W=90;     while W&gt;60     { R=W-50;     switch(W)     { 20:cout&lt;&lt;"Lower Range"&lt;&lt;endl;       30:cout&lt;&lt;"Middel Range"&lt;&lt;endl;       20:cout&lt;&lt;"Higher Range"&lt;&lt;endl;</pre>		

```

    }
}
}

```

(iv) Rewrite the following program after removing all the syntax error(s), if any.

```

#include<iostream.h>
void main(){
    int X[]={60, 50, 30, 40},Y;Count=4;
    cin>>Y;
    for(I=Count-1;I>=0,I--)
        switch(I)
        { case 0:
          case 2:cout<<Y*X[I]<<endl;break;
          case1:
          case 3:cout>>Y+X[I];
        }
}

```

(v) Rewrite the following program after removing all the syntax error(s), if any.

```

#include<iostream.h>
void main(){
    int P[]={90, 10, 24, 15},Q;Number=4;
    Q=9;
    for(int I=Number-1;I>=0,I--)
        switch(I)
        { case 0:
          case 2:cout>>P[I]*Q<<endl;
            break;
          case1:
          case 3:cout<<P[I]+Q;
        }
}

```

Ans. (i) Will encounter a following syntax error(s):

- The variable 'i' is not declared.
- There should be semicolon after the declaration statement if int x[5].

(ii) There is a syntax error in cout statement.

(iii) Will encounter a following syntax error(s):

- There should be a ',' between R and W instead of ';' in declaration statement.
- There should be a parenthesis in 'while' statement.
- There is missing a use of 'case' keyword in switch statement.

(iv) #include<iostream.h>

```

void main(){
    int X[]={60, 50, 30, 40},Y,Count=4;
    cin>>Y;
    for(int I=Count-1;I>=0;I--)
        switch(I)
        { case 0:
          case 1:
          case 2:cout<<Y*X[I]<<endl;break;

          case 3:cout<<Y+X[I];break;
        }
}

```

(v) #include<iostream.h>

```

void main(){
    int P[]={90, 10, 24, 15},Q,Number=4;

```

	<pre> Q=9; for(int I=Number-1;I&gt;=0;I--)     switch(I)     { case 0:       case 1:       case 2:cout&lt;&lt;P[I]*Q&lt;&lt;endl;                 break;       case 3:cout&lt;&lt;P[I]+Q;                 break;     } } </pre>
20	<p>Rewrite the following program after removing all the syntactical error(s), if any. Underline each correction.</p> <pre> #include&lt;iostream.h&gt; void main(){     Present=25,Past=35;     Assign(Present;Past);     Assign(Past); } void Assign(int Default1,Default2=30) {     Default1=Default1+Default2;     cout&lt;&lt;Default1&gt;&gt;Default2; } </pre>
Ans.	<pre> #include&lt;iostream.h&gt; void Assign(int Default1,int Default2=30); void main(){     clrscr();     int Present=25,Past=35;     Assign(Present,Past);     Assign(Present);     getch(); } void Assign(int Default1,int Default2) {     Default1=Default1+Default2;     cout&lt;&lt;Default1&lt;&lt;Default2; } </pre>
21	<p>Given the following code fragment:</p> <pre> if(a==0)     cout&lt;&lt;"Zero"; if(a==1)     cout&lt;&lt;"One"; if(a==2)     cout&lt;&lt;"Two"; if(a==3)     cout&lt;&lt;"Three"; </pre> <p>Write an alternative code (using if) that saves on number on compressions.</p>
Ans.	<pre> #include&lt;iostream.h&gt; void main(){     int a;     cout&lt;&lt;"enter a:";     cin&gt;&gt;a;     if(a==0)         cout&lt;&lt;"Zero";     else if(a==1) </pre>

	<pre>         cout&lt;&lt;"One" ;     else if(a==2)         cout&lt;&lt;"Two" ;     else if(a==3)         cout&lt;&lt;"Three" ;     else         cout&lt;&lt;"other than 0,1,2,3" ; } </pre>
<b>22</b>	<p><b>Write the names of the header files, which is/are essentially required to run/execute the following C++ code:</b></p> <pre> #include&lt;iostream.h&gt; void main(){     char CH,Text[]="+ve Attitude";     for(int I=0;Text[I]!='\0';I++)         if(Text[I]==' ')             cout&lt;&lt;endl;         else         {             CH=toupper(Text[I]);             cout&lt;&lt;CH;         } } </pre>
<b>Ans.</b>	1. ctype.h
<b>23</b>	<p><b>Find the output of the following program:</b></p> <pre> #include&lt;iostream.h&gt; void main(){     int A=5,B=10;     for(int I=1;I&lt;=2;I++)     {         cout&lt;&lt;"Line1"&lt;&lt;A++             &lt;&lt;"&amp;"&lt;&lt;B-2&lt;&lt;endl;         cout&lt;&lt;"Line2"&lt;&lt;A++B             &lt;&lt;"&amp;"&lt;&lt;A+3&lt;&lt;endl;     } } </pre>
<b>Ans.</b>	<p><u>Output:</u>  Line15&amp;8  Line211&amp;9  Line1679  Line212&amp;10</p>
<b>24</b>	<p><b>Rewrite the following program after removing all the syntactical error(s), if any. Underline each correction.</b></p> <pre> #include&lt;iostream.h&gt; void main(){     One=10,Two=20;     Callme(One;Two);     Callme(Two); } void Callme(int Arg1,int Arg2=20) {     Arg1=Arg1+Arg2     cout&lt;&lt;Arg1&gt;&gt;Arg2; } </pre>
<b>Ans.</b>	<pre> #include&lt;iostream.h&gt; void Callme(int Arg1,int Arg2=20); void main(){ </pre>

	<pre> int One=10,Two=20; Callme(One;Two); Callme(Two); } void Callme(int Arg1,int Arg2=20) { Arg1=Arg1+Arg2 cout&lt;&lt;Arg1&lt;&lt;Arg2; } </pre>
25	<p><b>Rewrite the following program after removing all the syntactical error(s), if any. Underline each correction.</b></p> <pre> #include&lt;iostream.h&gt; typedef char[80]; void main(){ String S="Peace"; int L=strlen(S); cout&lt;&lt;S&lt;&lt;'has'&lt;&lt;L &lt;&lt;'characters'&lt;&lt;endl; } </pre>
Ans.	<pre> #include&lt;iostream.h&gt; #include&lt;string.h&gt; typedef char <u>String</u>[80]; void main(){ String S="Peace"; int L=strlen(S); cout&lt;&lt;S&lt;&lt;"<u>has</u>"&lt;&lt;L &lt;&lt;"<u>characters</u>"&lt;&lt;endl; } </pre>
26	<p><b>Find the output of the following program:</b></p> <pre> #include&lt;iostream.h&gt; void SwitchOver(int A[],int N,int split) { for(int K=0;K&lt;N;K++) if(K&lt;Split) A[K]+=K; else A[K]*=K; } void Display(it A[],int N) { for(int K=0;K&lt;N;K++) (K%2==0)?cout&lt;&lt;A[K] &lt;&lt;"%":cout&lt;&lt;A[K]&lt;&lt;endl; } void main(){ int H[]={30,40,50,20,10,5}; SwitchOver(H,6,3); Display(H,6); } </pre>
Ans.	<p>Output:</p> <pre> 30%41 52%60 40%25 </pre>
27(a)	<p><b>The following code is from a game, which generates a set of 4 random numbers. Praful is playing this game, help him to identify the correct option(s) out of the four choices given below as the possible set of such numbers generated from the program code so that he wins the game. Justify your answer.</b></p>



```
#include<iostream.h>
#include<stdlib.h>
const int LOW=25;
void main(){
    randomize();
    int POINT=5,Number;
    for(int I=1;I<=4;I--)
    {
        Number=LOW+random(POINT);
        cout<<Number<<" ";
        POINT--;
    }
}
```

- (i) 29:26:25:28:      (ii) 24:28:25:26:      (iii) 29:26:24:28:      (iv) 29:26:25:26:

**Ans.** (iv) 29: 26: 25: 26: is correct  
**Justification:**  
 The only option that satisfied the values as per the code is option (iv) because when:

I	POINT	Number	
		Minimum	Maximum
1	5	25	29
2	4	25	28
3	3	25	27
4	2	25	26

**27(b)** Study the following program and select the possible output from it:

```

)
#include<iostream.h>
#include<stdlib.h>
const int LIMIT=4;
void main(){
    randomize();
    int Points;
    points=100+random(LIMIT);
    for(int P=Pints;P>=100;P--)
        cout<<P<<"#";
    cout<<endl;
}

```

- (i) 103#102#101#100#                      (ii) 100#101#102#103#  
 (iii) 100#101#102#103#104#          (iv) 104#103#102#101#100#

**Ans.** (i) 103#102#101#100# is correct answer.

**28** Go through C++ code show below, and find out the possible output or outputs from the suggested Output Options (i) to (iv)0. Also, write the least value and highest value, which can be assigned to the variable MyNum.

```
#include<iostream.h>
#include<stdlib.h>
void main(){
    randomize();
    int MyNum,Max=5;
    MyNum=20+random(Max);
    for(int N=MyNum;N<=25;N++)
        cout<<N<<"*";
}

```

- (i) 20\*21\*22\*23\*24\*25                      (ii) 22\*23\*24\*25  
 (iii) 23\*24\*                                      (iv) 21\*22\*23\*24\*25

**Ans.** (ii) 22\*23\*24\*25 is correct answer. Minimum possible value = 20, Maximum possible value = 24

**29(a)** Find the output of the following program:

	<pre> ) #include&lt;iostream.h&gt; #include&lt;ctype.h&gt; void main(){     char Line[]="Good@LOGIC";     for(int I=0;Line(I]!='\0';I++)     {         if(!isalpha(Line[I]))             Line[I]='\$';         else if(islower(Line[I]))             Line[I]=Line[I]+1;         else             Line(I)=Line[I+1];     }     cout&lt;&lt;Line; } </pre>
Ans.	Output: Oppe\$OGIC
29(b)	<p>Find the output of the following program:</p> <pre> ) #include&lt;iostream.h&gt; void main(){     int First=25,Sec=30;     for(int I=1;I&lt;=2;I++0     {         cout&lt;&lt;"Output1="&lt;&lt;First++             &lt;&lt;"&amp;"&lt;&lt;Sec+5&lt;&lt;endl;         cout&lt;&lt;"Output2="&lt;&lt;-Sec             &lt;&lt;"&amp;"&lt;&lt;First-5&lt;&lt;endl;     } } </pre>
Ans.	<p>Output:</p> <p>Output1=25&amp;35  Output2=-30&amp;21  Output1=26&amp;35  Output2=-30&amp;22</p>
29(c)	<p>Find the output of the following program:</p> <pre> ) #include&lt;iostream.h&gt; #include&lt;ctype.h&gt; void Encode(char Info[],int N); void main(){     char Memo[]="Justnow";     Encode(Memo,2);     cout&lt;&lt;Memo&lt;&lt;endl; } void Encode(char Info[],int N) {     for(int I=0;Info[I]!='\0';I++)         if(I%2==0)             Info[I]=Info[I]-N;         else if(islower(Info[I]))             Info[I]=toupper(Info[I]);         else             Info[I]=Info[I]+N; } </pre>
Ans.	Output: HUqT10u
29(d)	<p>Find the output of the following program:</p> <pre> ) #include&lt;iostream.h&gt; </pre>

```

struct THREE_D
{
    int X,Y,Z; };
void MoveIn(THREE_D &T,int Step=1)
{
    T.X+=Step;
    T.Y-=Step;
    T.Z+=Step;
}
void MoveOut(THREE_D &T,it Step=1)
{
    T.X-=Step;
    T.Y+=Step;
    T.Z-=Step;
}
void main(){
    THREE_D T1={10,20,5}, T2={30,10,40};
    MoveIn(T1);
    MoveOut(T2,5);
    cout<<T1.X<<","<<T1.Y<<","<<T1.Z<<endl;
    cout<<T2.X<<","<<T2.Y<<","<<T2.Z<<endl;
    MoveIn(T2,10);
    cout<<T2.X<<","<<T2.Y<<","<<T2.Z<<endl;
}

```

Ans. Output:  
11,19,6  
25,15,35  
35,5,45

30 Give the output of the following program:

```

(i) void main(){
    char *p="Difficult";
    char c;
    c=++*p++;
    printf("%c",c);
}

(ii) #include<iostream.h>
static int i=100;
void abc()
{
    static int i=8;
    cout<<"first="<<i;
}
main(){
    static int i=2;
    abc();
    cout<<"second="<<i<<endl;
}

(iii) #include<iostream.h>
void Print(char *p)
{
    p="Pass";
}

```

	<pre> cout&lt;&lt;"Value is:"&lt;&lt;p&lt;&lt;endl; } void main(){ char *q="Best of Luck"; Printf(q); cout&lt;&lt;"New value is:"&lt;&lt;q; } </pre>		
<b>Ans.</b>	(i) <u>Output:</u> E	(ii) <u>Output:</u> first=8second=2	(iii) <u>Output:</u> Value is: Pass New value is: Best of Luck
<b>31</b>	<b>Give the output of the following:</b>		
	<pre> (i) #include&lt;iostream.h&gt; void Execute(int &amp;X,int Y=200) { int TEMP=X+Y; X+=TEMP; if(Y!=200) cout&lt;&lt;TEMP&lt;&lt;X&lt;&lt;Y&lt;&lt;endl; } void main(){ int A=50,B=20; Exwcute(B); cout&lt;&lt;a&lt;&lt;B&lt;&lt;endl; Exwcute(A,B); cout&lt;&lt;A&lt;&lt;B&lt;&lt;endl; } </pre>	<pre> (ii) #include&lt;iostream.h&gt; void Execute(int &amp;B,int C=200) { int TEMP=B+C; B+=TEMP; if(C==100) cout&lt;&lt;TEMP&lt;&lt;B&lt;&lt;C&lt;&lt;endl; } void main(){ int M=90,N=10; Exwcute(M); cout&lt;&lt;M&lt;&lt;N&lt;&lt;endl; Exwcute(M,N); cout&lt;&lt;m&lt;&lt;N&lt;&lt;endl; } </pre>	
<b>Ans.</b>	(i) <u>Output:</u> 50240 290340240 340240	(ii) <u>Output:</u> 38010 77010	
<b>32</b>	<b>Find the output of the following program:</b>		
	<pre> #include&lt;iostream.h&gt; #include&lt;ctype.h&gt; void MyCode(char Msg[],char CH) { for(int Cnt=0;Msg[cnt]!='\0';Cnt++) { if(Msg[Cnt]&gt;='B' &amp;&amp; Msg[Cnt]&lt;='G') Msg[Cnt]=tolower(Msg[Cnt]); else if(Msg[Cnt]&gt;='A' &amp;&amp; Msg[Cnt]&lt;='a') Msg[Cnt]=CH; else if(Cnt%2==0) Msg[Cnt]=toupper(Msg[Cnt]); else Msg[Cnt]=Msg[Cnt-1]; } } void main(){ char MyText[]="ApEACeDrIVE"; MyCode(MyText,'@'); cout&lt;&lt;"NEW TEXT:"&lt;&lt;MyText&lt;&lt;endl; } </pre>		

	<pre> } </pre>
<b>Ans.</b>	(i) <u>Output:</u> NEW TEXT:@@e@ccddl@e
<b>33</b>	<p>Find the output of the following program:</p> <pre> #include&lt;iostream.h&gt; struct Package {     int Length,Breadth,Height; }; void Occupies(Package M) {     cout&lt;&lt;M.Length&lt;&lt;"x"         &lt;&lt;M.Breadth&lt;&lt;"x";     cout&lt;&lt;M.Height&lt;&lt;endl; } void main() {     Package P1={100,150,50},P2,P3;     ++P1.Length;     Occupies(P1);     P3=P1;     ++P3.Breadth;     P3.Breadth++;     Occupies(P3);     P2=P3;     P2.Breadth+=50;     P2.Height--;     Occupies(P2); } </pre>
<b>Ans.</b>	<p><u>Output:</u>  101x150x50  101x152x50  101x202x49</p>
<b>34</b>	<p>What values will be assigned to the variables ua, ub, uc and fail after the execution of the following program segment:</p> <pre> void main(){     int i=0,ua=0,ub=0,uc=0,fail=0;     while(i&lt;=5)     {         switch(i++)         {             case 1:             case 2: ++ua;             case 3:             case 4: ++ub;             case 5: ++uc;             default: ++fai;         } //switch     } //while     cout&lt;&lt;ua&lt;&lt;endl;     cout&lt;&lt;ub&lt;&lt;endl;     cout&lt;&lt;uc&lt;&lt;endl;     cout&lt;&lt;fail&lt;&lt;endl; } </pre>
<b>Ans.</b>	<p>Values assigned to the variables ua, ub, uc and fail after the execution are as following:  ua = 2</p>

	<pre>ub = 4 uc = 5 fail = 6</pre>
35	<p><b>What is wrong with this code:</b></p> <pre>cout&lt;&lt;" " Enter n:"; cin&gt;&gt;n; if(n&lt;0)     cout&lt;&lt;"That is negative."     &lt;&lt;"Please enter again"&lt;&lt;endl;     cin&gt;&gt;n; else     cout&lt;&lt;"O.K. n="&lt;&lt;n&lt;&lt;endl;</pre>
Ans.	<p>There are following errors in above code:</p> <p>(i) Imbalance Double quotes are in first statement.</p> <p>(ii) There are more than one statement in 'if' statement without braces which is not valid.</p>
36	<p><b>Identify the possible error(s) in the following code fragment. Discuss the reason(s) of error(s) and correct the code:</b></p> <pre>cin&gt;&gt;i&gt;&gt;j; while(i&lt;j)     cout&lt;,i*j; i++;</pre>
Ans.	<p><b>Correct code:</b></p> <pre>cin&gt;&gt;i&gt;&gt;j; while(i&lt;j) {     cout&lt;&lt;i*j;     i++; }</pre> <p><b>Reasons of errors:</b></p> <p>(i) There is a use of '&lt;' in cout statement instead of '&lt;&lt;' which is not valid.</p> <p>(ii) while statement should be in curly braces otherwise it will not work properly.</p>
37	<p><b>Given the following code fragment:</b></p> <pre>i=2; start: cout&lt;&lt;i; i+=2; if(i&lt;51) goto start; cout&lt;&lt;"\nThank You";</pre> <p><b>Rewrite the above code using a while loop.</b></p>
Ans.	<pre>int i=2; while(i&lt;51) {     cout&lt;&lt;i;     i+=2; }  cout&lt;&lt;"\nThank You";</pre>
38	<p><b>Is it necessary to include a header file in a program? If it is not done, what happens?</b></p>
Ans.	<p>No, it is not necessary to include header file in a C++ program at all.</p> <p>A header file is included if only the program intends to use the functions or macros etc. defined in that particular header file.</p>
39	<p><b>Name the header files that shall be needed for the following code:</b></p> <pre>void main(){     char Text[]="Computer";     cout&lt;&lt;setw(15)&lt;&lt;Text;</pre>

	}																													
<b>Ans.</b>	(i) iomanip.h for setw() (ii) iostream.h for cout																													
<b>40</b>	<b>Briefly describe the importance of iostream.h file.</b>																													
<b>Ans.</b>	iostream.h is most important header file for basic input and output operations. iostream.h contain built-in functions cout and cin for input-output. So, every C++ program should include iostream.h header file for performing input and output operations.																													
<b>41</b>	<b>How do the following two statements differ in operation?</b> <code>cin&gt;&gt;ch;</code> <code>cin.get(ch);</code>																													
<b>Ans.</b>	The difference between cin>>ch and cin.get(ch) is that when >> operator is used, the white spaces (e.g., tabs), spaces etc. and new-line characters are ignored whereas it is not so with cin.get(ch).																													
<b>42</b>	<b>What is an array? What is the need for array?</b>																													
<b>Ans.</b>	<u>Array</u> : An array is a collection of variables of the same type that are referenced by a common name. <u>Need for array</u> : Arrays are very much useful in a case where many variables of the same (data) types need to be stored and processed. For example, suppose we have to write a program in which can accept salary of 50 employees. If we solve this problem by making use of variables, we need 50 variables to store employee's salary. Remembering and managing these 50 variables is not an easy task and it will make the program a complex and lengthy program. This problem can be solved by declaring 1 array having 50 elements; one for employee's salary. Now we only have to remember the name of that 1 array.																													
<b>43</b>	<b>What do you understand by two-dimensional arrays? State some situation that can be easily represented by two-dimensional arrays.</b>																													
<b>Ans.</b>	<u>Two-dimensional array</u> : A two-dimensional array is an array in which each element is itself an array. For instance, an array A[m][n] is an M by N table with M rows and N columns containing M x N elements. Two-dimensional arrays are used to represent tables, matrices, etc. For example, below marks table can be represented easily by 2D array :																													
	<table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr> <th></th> <th>Maths</th> <th>Physics</th> <th>Chemistry</th> <th>CS</th> <th>Englisg</th> <th>Toal</th> </tr> </thead> <tbody> <tr> <th>Amit</th> <td>60</td> <td>64</td> <td>64</td> <td>82</td> <td>79</td> <td>349</td> </tr> <tr> <th>Chandar</th> <td>84</td> <td>70</td> <td>82</td> <td>79</td> <td>69</td> <td>349</td> </tr> <tr> <th>Lalit</th> <td>79</td> <td>68</td> <td>86</td> <td>87</td> <td>85</td> <td>698</td> </tr> </tbody> </table>			Maths	Physics	Chemistry	CS	Englisg	Toal	Amit	60	64	64	82	79	349	Chandar	84	70	82	79	69	349	Lalit	79	68	86	87	85	698
	Maths	Physics	Chemistry	CS	Englisg	Toal																								
Amit	60	64	64	82	79	349																								
Chandar	84	70	82	79	69	349																								
Lalit	79	68	86	87	85	698																								
<b>44</b>	<b>Can you think of a difference between an array of strings and other two-dimensional arrays? What is it? Support your answer with examples.</b>																													
<b>Ans.</b>	<b>Array of strings</b>	<b>Two-dimensional array</b>																												
	Array of string is used to store string value.	Two-dimensional array is used to store numeric value.																												
	The first index determines the number of strings and the second index determines maximum length of each string.	The first index determines the number of rows and the second index determines maximum columns of each string.																												
	<u>Example:</u> <pre>int a[2][3]; int i, j; for(i=0; i&lt;2; i++) {     for(j=0; j&lt;3; ++j)     {         cout&lt;&lt;"Enter element:";         cin&gt;&gt;a[i][j];     } }</pre>	<u>Example:</u> <pre>char string[3][31]; int i; cout&lt;&lt;"Enter 3 strings:"; for(i=0; i&lt;3; i++)     cin.getline(string[i], 31);</pre>																												
<b>45</b>	<b>If an array is initialized at the time of declaration, what thing one must bear in mind?</b>																													

<b>Ans.</b>	The general form of array initialization is as shown below: <code>type array-name[size N] = {value-list};</code> If an array is initialized at the time of declaration, following thing one must bear in mind: <ul style="list-style-type: none"> <li>✓ The element values in the <i>value-list</i> must have the same data type as that of <i>type</i>, the base type of the array.</li> <li>✓ In character array, you must make sure that the array you declare is long enough to include the null.</li> </ul>										
<b>46</b>	<b>What is meant by unsized array initialization in C++? What are its benefits?</b>										
<b>Ans.</b>	Unsized array initialization means skip the size of the array in an initialization statement. The C++, then automatically creates an array big enough to hold all the initializers present and calculates the dimensions of unsized arrays. <u>Example:</u> <code>char S1[] = "First String";</code> <u>Advantages:</u> <ul style="list-style-type: none"> <li>✓ Less tedious.</li> <li>✓ Allow to change any of the values without fear of using incorrect array dimensions.</li> <li>✓ We may lengthen or shorten the value-list without changing the array dimensions.</li> </ul>										
<b>47</b>	<b>What do you mean by function prototyping? Write down the advantages of function prototypes in C++.</b>										
<b>Ans.</b>	A function prototype is a declaration of the function that tells the program about the type of value return by the function the number and type of arguments. <u>Example:</u> <code>int sum(int a, int b);</code> <u>Advantages:</u> <ul style="list-style-type: none"> <li>✓ Enables a compiler to carefully compare each use of the function with the prototype to determine whether the function is invoked properly i.e., the number and type of arguments are compared and any wrong number or type of the arguments is reported.</li> <li>✓ Tells the program about the return type</li> <li>✓ Tells the program the number and type of arguments.</li> </ul>										
<b>48</b>	<b>What are actual and formal parameters of a function?</b>										
<b>Ans.</b>	The arguments passed to the functions while the function is called is known as the actual arguments, whereas the arguments declared in the function header is called as formal arguments. Ex. Suppose sum() is a function. <pre>int sum(int x, int y) /*Here x and y are called formal arguments*/ {...} void main() { int ans; .... ans = sum(3,5); /*Here the arguments 3 and 5 are called actual arguments*/ ... getch(); }</pre>										
<b>49</b>	<b>Construct function prototype for descriptions given below:</b>										
	<table border="1"> <tr> <td><b>test()</b></td> <td>takes no arguments and has no return value.</td> </tr> <tr> <td><b>convert()</b></td> <td>takes a <b>float</b> argument and returns an <b>int</b>.</td> </tr> <tr> <td><b>promote()</b></td> <td>takes two <b>double</b> arguments and returns a <b>double</b>.</td> </tr> <tr> <td><b>sum()</b></td> <td>takes an <b>int</b> array and an <b>int</b> value and returns a <b>long</b> result.</td> </tr> <tr> <td><b>check()</b></td> <td>takes a <b>string</b> argument and returns an <b>int</b>.</td> </tr> </table>	<b>test()</b>	takes no arguments and has no return value.	<b>convert()</b>	takes a <b>float</b> argument and returns an <b>int</b> .	<b>promote()</b>	takes two <b>double</b> arguments and returns a <b>double</b> .	<b>sum()</b>	takes an <b>int</b> array and an <b>int</b> value and returns a <b>long</b> result.	<b>check()</b>	takes a <b>string</b> argument and returns an <b>int</b> .
<b>test()</b>	takes no arguments and has no return value.										
<b>convert()</b>	takes a <b>float</b> argument and returns an <b>int</b> .										
<b>promote()</b>	takes two <b>double</b> arguments and returns a <b>double</b> .										
<b>sum()</b>	takes an <b>int</b> array and an <b>int</b> value and returns a <b>long</b> result.										
<b>check()</b>	takes a <b>string</b> argument and returns an <b>int</b> .										
<b>Ans.</b>	(i) void test(); (ii) int convert(float a); (iii) double promote(double a, double b); (iv) long sum(int a[], int b); (v) int check(string s);										
<b>50</b>	<b>What do you understand by default arguments and constant argument? Write a short note on their usefulness.</b>										
<b>Ans.</b>	<b>Default arguments:</b> C++ allows us to assign default values to a function's parameters which are useful in case a matching argument is not passed in the function call statement. The default values are specified at the time of										



function declaration. Example: `float interest(float principal, int time, float rate=0.10);`  
**Constant argument:** By constant argument, it is meant that the function cannot modify these arguments. In order to make an argument constant to a function, we can use the keyword *const* as shown : `int sum (const int a, const int b);`  
 The qualifier *const* in function prototype tells the compiler that the function should not modify the argument. The constant arguments are useful when functions are called by reference.

**51 How is call-by-value method of function invoking different from call-by-reference method? Give appropriate examples supporting your answer.**

Ans.	Call By Value	Call by reference
	✓ In call by value method, the called function creates its own copies of the original values send to it.	✓ In call by reference method, the called function accesses ad works with the original values using their references.
	✓ The changes done in the function in formal parameter are not reflected back in the calling environment.	✓ The changes done in the function are reflected back in the calling environment.
	✓ It does not use the '&' sign	✓ It use '&' sign as the reference operator.
	<b>Example:</b> <pre>#include &lt;iostream.h&gt; void change(int x, int y){     x = 10; /*change the value of x */     y = 20; /*change the value of y */ } void change(int x, int y);  void main (){     // local variable declaration:     int a = 100;     int b = 200;      cout&lt;&lt;"Before value of a "&lt;&lt;a &lt;&lt;endl;     cout&lt;&lt;"Before value of b "&lt;&lt;b&lt;&lt; endl;     change(a, b);     cout&lt;&lt;"After value of a "&lt;&lt;a&lt;&lt;endl;     cout&lt;&lt;"After value of b "&lt;&lt;b&lt;&lt;endl; }</pre>	<b>Example:</b> <pre>#include &lt;iostream.h&gt; void change(int *x, int *y){     *x = 10; /*change the value of x */     *y = 20; /*change the value of y */ } void change(int *x, int *y);  void main (){     // local variable declaration:     int a = 100;     int b = 200;      cout&lt;&lt;"Before value of a "&lt;&lt;a &lt;&lt;endl;     cout&lt;&lt;"Before value of b "&lt;&lt;b&lt;&lt; endl;     change(&amp;a, &amp;b);     cout&lt;&lt;"After value of a "&lt;&lt;a&lt;&lt;endl;     cout&lt;&lt;"After value of b "&lt;&lt;b&lt;&lt;endl; }</pre>

**52 Discuss the similarities and difference between global and local variables in terms of their lifetime and scope.**

Ans.	Local variable	Global variable
	✓ It is a variable which is declared within a function or within a compound statement.	✓ It is a variable which is declared outside all the functions.
	✓ It is accessible only within a function/compound statement in which it is declared	✓ It is accessible throughout the program
	✓ A global variable comes into existence when the program execution starts and is destroyed when the program terminates.	✓ A local variable comes into existence when the function is entered and is destroyed upon exit.

**Example:**  

```
#include <iostream.h>
float NUM=900; //NUM is a global variable
void LOCAL(int T)
{
    int Total=0; //Total is a local variable
    for (int I=0;I<T;I++)
        Total+=I; cout<<NUM+Total;
```

	<pre> } void main(){     LOCAL(45); } </pre>
<b>53</b>	<b>What is structure? Declare a structure in C++ with name, roll number and total marks as components.</b>
<b>Ans.</b>	<p>A structure is a collection of variables referenced under one name. A structure is declared using the keyword struct as in following syntax:</p> <pre> struct &lt;structure tag&gt; {     [public:]   [private:]   [protected:]     /* data members' declarations */     /* member functios' declarations */ }; </pre> <p><u>Example:</u></p> <pre> struct Student {     char Name[30];     int Rollno;     float Total_Marks; }; </pre>
<b>54</b>	<b>What are Nested structures? Give an example.</b>
<b>Ans.</b>	<p>A structure within a structure is called nested structures.</p> <p><u>Example:</u></p> <pre> struct addr                //structure tag {     int houseno;     char area[26];     char city[26];     char state[26]; }; struct emp                //structure tag {     int empno;     char name[26];     char desig[16];     addr address;     float basic; }; emp worker;                // create structure variable </pre> <p><i>See, address is a structure variable itself and it is member of another structure, the emp structure.</i></p> <p>The structure emp has been defined having several elements including a structure address also. The address is itself a structure of type addr. While defining such structures are defined before outer structures.</p>
<b>55</b>	<b>Write a program that asks the user to enter two integers, obtains the two numbers from the user, and outputs the large number followed by the words "is larger by – units than smaller number" to the system console (e.g., if the larger number is 9 and smaller is 6, message should be "9 is larger by 3 units than smaller number"). If the numbers are equal print the message "These numbers are equal".</b>
<b>Ans.</b>	<pre> #include&lt;iostream.h&gt; #include&lt;conio.h&gt; void main() {     int a,b,dif;     clrscr();     cout&lt;&lt;"Enter a:"; </pre>

```

cin>>a;
cout<<endl<<"Enter b:";
cin>>b;
if(a>b)
{
    dif=a-b;
    cout<<a<<"is larger by"<<dif<<"units than smaller number"<<endl;
}
else if(b>a)
{
    dif=b-a;
    cout<<b<<"is larger by"<<dif<<"units than smaller number"<<endl;
}
else
    cout<<"These numbers are equal"<<endl;
getch();
}

```

- 56 Drivers are concerned with the mileage obtained by their automobiles. One driver has kept track of several tanks of CNG by recording the miles driven and the gallons used for each tank. Develop a C++ program that will input the kilometers driven and gallons used for each tank. The program should calculate and display the *kilometers per gallon* obtained for each tank of gasoline. After processing all input information, the program should calculate and print the average *kilometers per gallon* obtained for all tanks.**
- Formulate the algorithm as flowchart.
  - Write a C++ program as instructed.
  - Test, debug, and execute the C++ program.

**Ans.**

```

#include<iostream.h>
#include<conio.h>
void main(){
    clrscr();
    int tanks=0;
    float tot_km; float avg_k_p_g;
    cout<<"Enter how many tanks filled :";
    cin>>tanks;
    float *kms=new float[tanks];
    float *gallons_used=new float[tanks];
    float *k_p_g=new float[tanks];
    for(int i=0;i<tanks;i++)
    {
        cout<<"Enter how much kilometers covered for tank "<<i+1<<" ";
        cin>>kms[i];
        cout<<"Enter how much gallon used from tank "<<i+1<<" ";
        cin>>gallons_used[i];
        k_p_g[i]=kms[i]/gallons_used[i];
        cout<<"KMS per Gallon obtained for tank No. "<<i+1<<" "<<k_p_g[i]<<endl;
        tot_km+=k_p_g[i];
        cout<<endl;
    }
    avg_k_p_g=tot_km/tanks;
    cout<<"Average kilometers per gallon obtained for all tanks is "<<avg_k_p_g;
    getch();
}

```