H.R.P.G. College Khalilabad, SantKabir Nagar

Online Classes

Faculty of Science

1.Department	: Mathematics
2.Class	: B.Sc.1st year
3.Name of teacher	: Dr.Dinesh Kumar Gupta
4.Date	: 08/05/2020 Time:10.15 A.M. - 11.00 A.M.
5.Paper	: 1st (Trigonometry & Algebra)
6.Topic	: QUESTIONS ON DIFFERENCE METHOD
7.Students	: 02

[illegible]

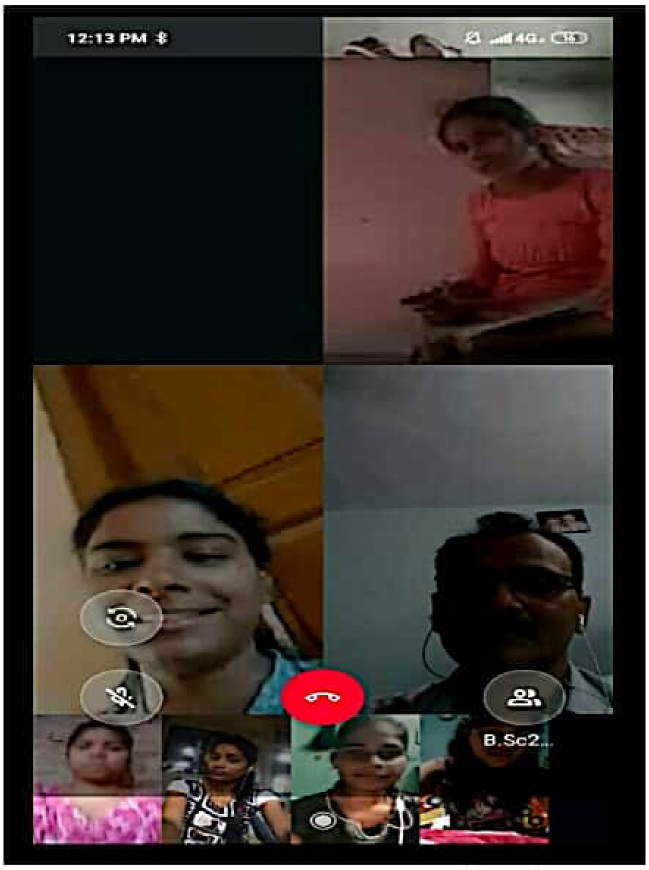
Dr. Dinesh Kumar Gupta
Assistant Professor
Department of Mathematics

H.R.P.G. College Khalilabad, SantKabir Nagar

Online Classes

Faculty of Science

1.Department	: Mathematics
2.Class	: B.Sc.2nd year
3.Name of teacher	: Dr.Dinesh Kumar Gupta
4.Date	: 08/05/2020 Time:11:00 A.M. - 11:45 A.M.
5.Paper	: 1st (ABSTRACT ALGEBRA)
6.Topic	: VECTOR SPACE
7.Students	: 07



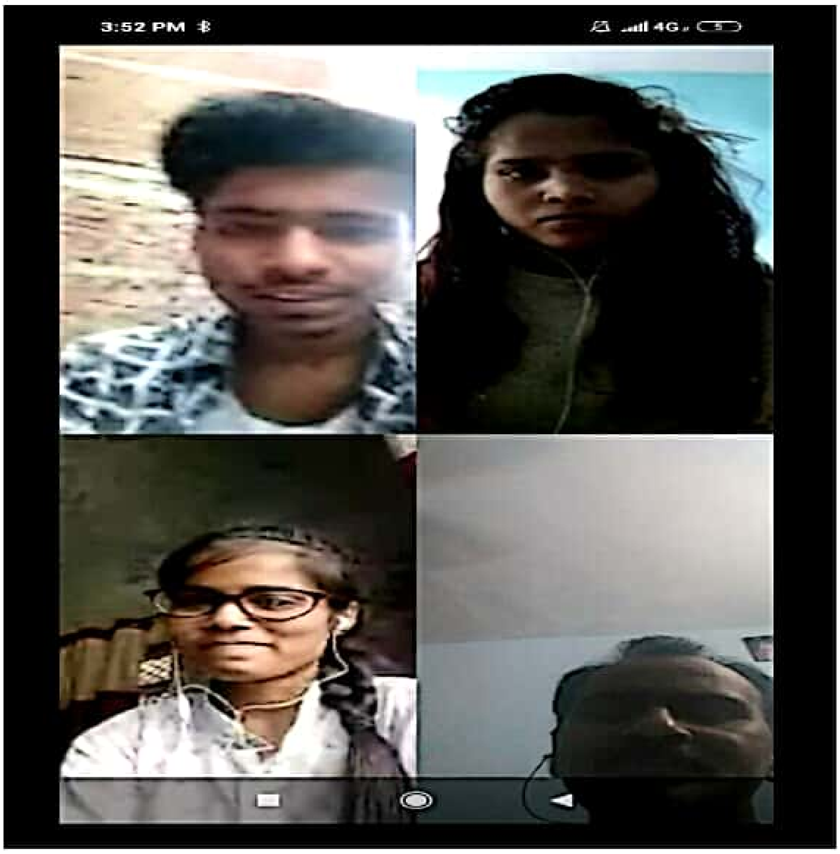
[illegible]

Dr. Dinesh Kumar Gupta
Assistant Professor
Department of Mathematics

H.R.P.G. College Khalilabad SantKabir Nagar

Online Classes
Faculty of B.C.A.

1.Department	: Mathematics
2.Class	: B.C.A. Ist year (IInd semester)
3.Name of teacher	: Dr.Dinesh Kumar Gupta
4.Date	: 08/05/2020 Time: 03:00 P.M. - 04:00 P.M.
5.Paper	: 3rd -Discrete Mathematics
6.Topic	: Partial Ordering Relations
7.Students	: 03



[illegible]

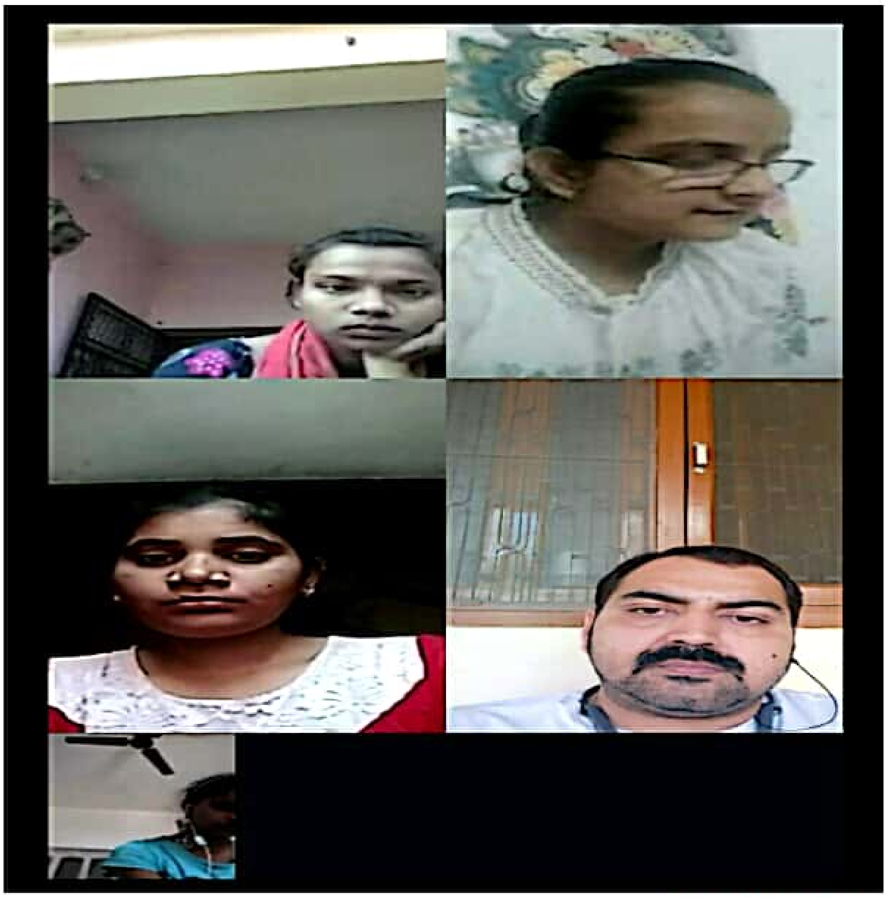
Dr. Dinesh Kumar Gupta
Assistant Professor
Department of Mathematics

H.R.P.G. College Khalilabad SantKabir Nagar

Online Classes

Faculty of Science

1.Department	: ZOOLOGY
2.Class	: .B.Sc. 1st year
3.Name of Teacher	: Dr. Anupam Pati Tripathi
4.Date	: 08/05/2020 Time: 09:00A.M. - 10:00A.M.
5.Paper	: 3rd Paper
6.Topic	: Linkage & Crossing Over
7.Students	: 04



Dr. Anupam Pati Tripathi

Assistant Professor

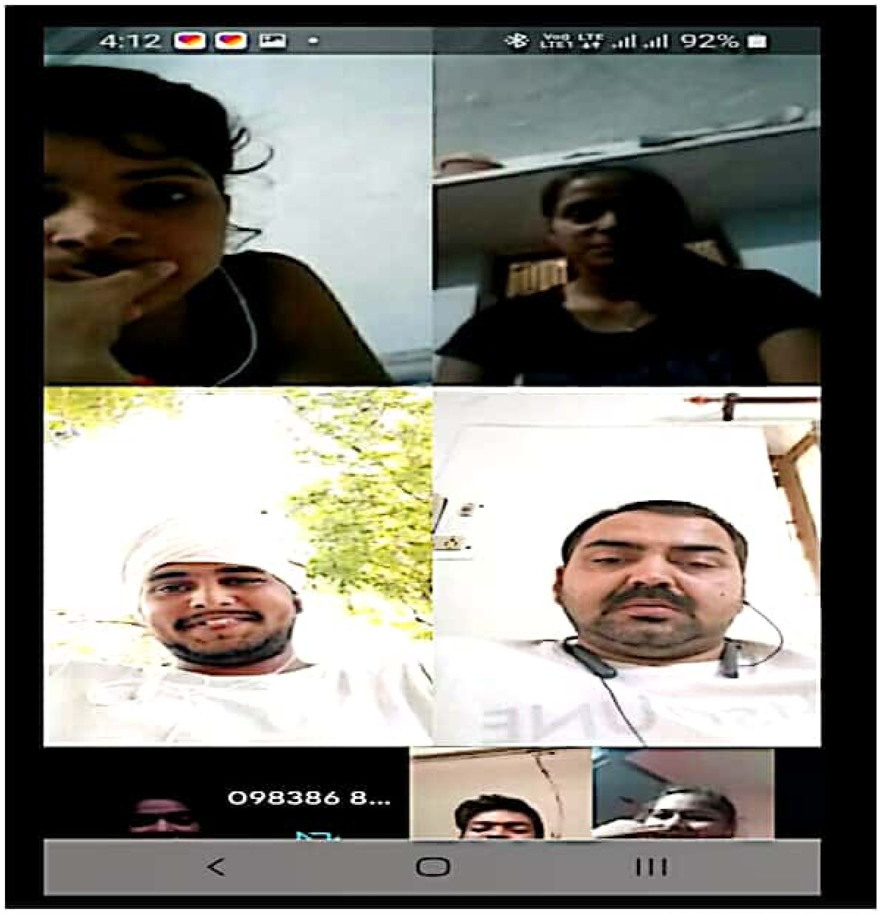
Department Of Zoology

H.R.P.G. College Khalilabad SantKabir Nagar

Online Classes

Faculty of Science

1.Department	: ZOOLOGY
2.Class	: .B.Sc. 2nd Year
3.Name of Teacher	: Dr. Anupam Pati Tripathi
4.Date	: 08/05/2020 Time: 03:00P.M. - 04:00P.M.
5.Paper	: 1st Paper
6.Topic	: **Protochordata, Histology and Embryology**
7.Students	: 06



Dr. Anupam Pati Tripathi

Assistant Professor

Department Of Zoology

H.R.P.G. College Khalilabad SantKabir Nagar

Online Classes
Faculty of Science

1.Department	: ZOOLOGY
2.Class	: .M.Sc. 2nd Year (4th Semester)
3.Name of Teacher	: Dr. Anupam Pati Tripathi
4.Date	: 08/05/2020 Time: 11:00A.M. - 12:00A.M.
5.Paper	: 1st Paper
6.Topic	: Comparative Anatomy
7.Students	: 05

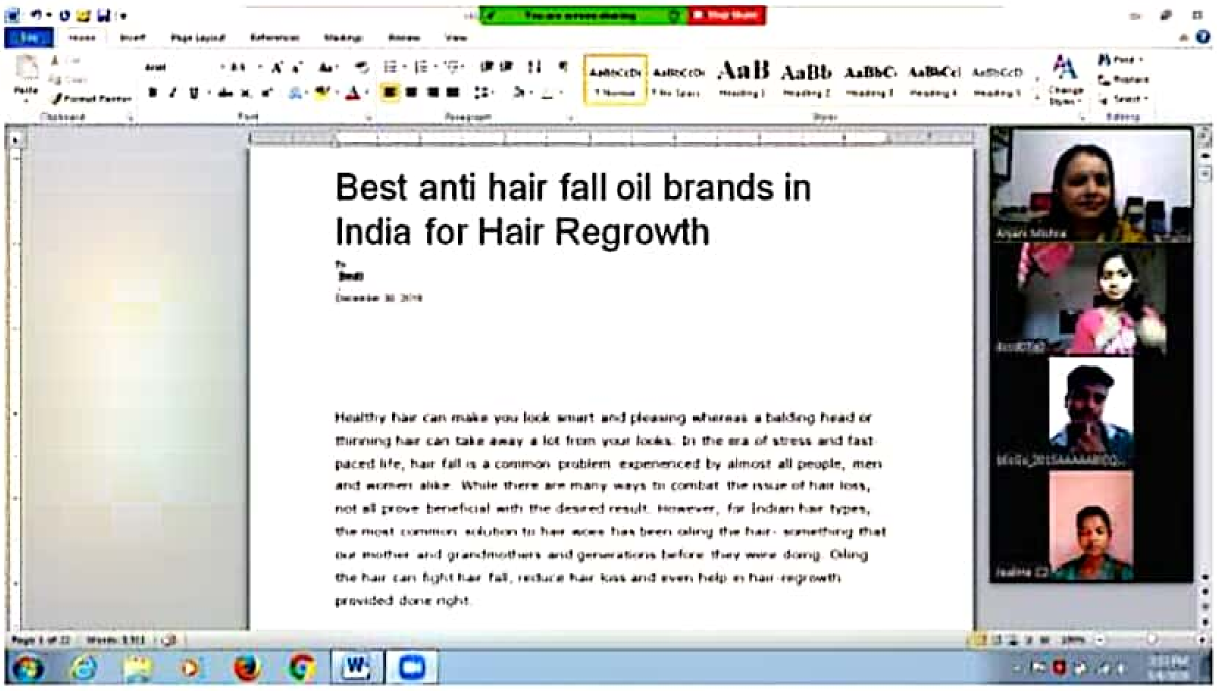
Dr. Anupam Pati Tripathi

Assistant Professor

Department Of Zoology

H.R.P.G. College, Khalilabad, SantKabir Nagar
Online Classes
Faculty of B.Sc.

1.	Department	: **Computer Application**
2.	Class	: B.Sc. 1st year
3.	Name of Teacher	: Smt. Anjani Mishra
4.	Date	: 8-MAY-2020 Time: 3:00 To 4:00 P.M.
5.	Paper	: 3rd Paper – pc software
6.	Topic	: Page set up And Change Case
7.	No. of Students	: 5


Best anti hair fall oil brands in India for Hair Regrowth

December 30, 2019

Healthy hair can make you look smart and pleasing whereas a balding head or thinning hair can take away a lot from your looks. In the era of stress and fast paced life, hair fall is a common problem experienced by almost all people, men and women alike. While there are many ways to combat the issue of hair loss, not all prove beneficial with the desired result. However, for Indian hair types, the most common solution to hair woes has been oiling the hair - something that our mother and grandmothers and generations before they were doing. Oiling the hair can fight hair fall, reduce hair loss and even help in hair regrowth provided done right.

Anjani Mishra

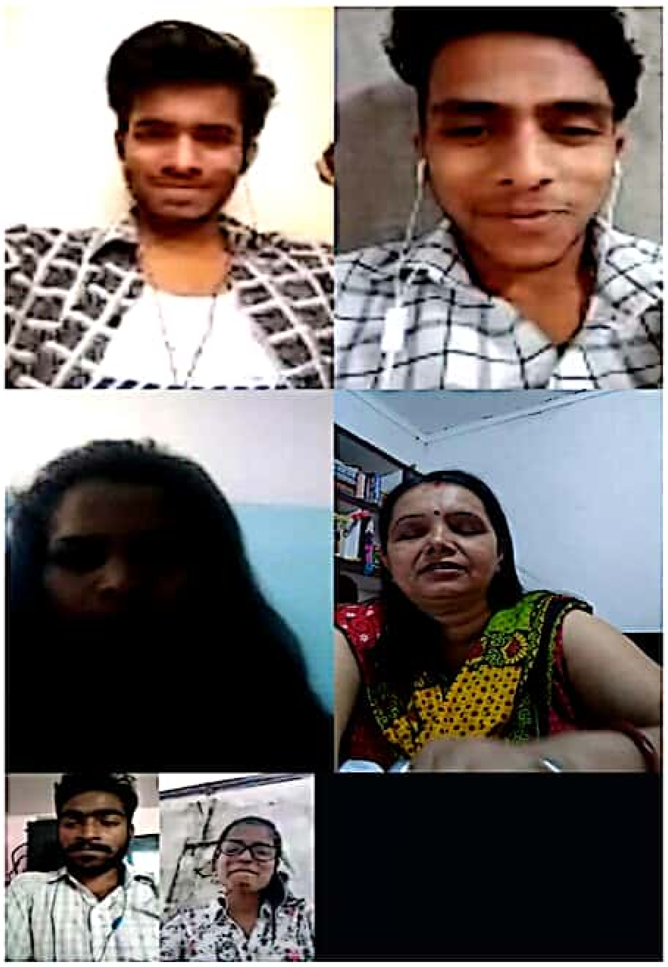
Smt. Anjani Mishra
Lecturer
B.Sc.

H.R.P.G. College, Khalilabad, SantKabir Nagar

Online Classes

Faculty of B.C.A.

1.	Department	: **Bachelor of Computer Application**
2.	Class	: B.C.A. 1st year (Second Semester)
3.	Name of Teacher	: Smt. Anjani Mishra
4.	Date	: 8-MAY-2020 Time : 4:00 To 5:00 P.M.
5.	Paper	: 1st Paper - Business System
6.	Topic	: Type Of Functions Using With Command Window
7.	No. of Students	: Five (5)



Anjani Mishra

Smt. Anjani Mishra
Lecturer
B.C.A.

H.R.P.G. COLLEGE KHALILABAD- SANT KABIR NAGAR

ONLINE CLASSES-FACULTY OF B.C.A.

Department	: Computer Application
Class	: B.C.A –II Semester(1st Year)
Name of Teacher	: Imran Ahmad
Date	: 08/05/2020
Time	: 02:00 P.M – 03:00 P.M.
Paper	: BCA-204 Introduction to Object Oriented Programming & C++
Topic	: Functions in C++(6)/notes
No. of Students	: 06

3.24 OPERATOR PRECEDENCE

Although C++ enables us to add multiple meanings to the operators, yet their association and precedence remain the same. For example, the multiplication operator will continue having higher precedence than the add operator. Table 3.7 gives the precedence and associativity of all the C++ operators. The groups are listed in the order of decreasing precedence. The labels *prefix* and *postfix* distinguish the uses of ++ and --. Also, the symbols +, −, *, and & are used as both unary and binary operators.

A complete list of ANSI C++ operators and their meanings, precedence, associativity and use are given in Appendix E.

Table 3.7 *Operator precedence and associativity*

Operator	*Associativity*
::	left to right
−> . ( ) [ ] postfix ++ postfix − −	left to right
prefix ++ prefix − − ~ ! unary + unary − unary * unary & (type) sizeof new delete	right to left
−> * *	left to right
* / %	left to right
+−	left to right
<< >>	left to right
<< = >> =	left to right
= = !=	left to right
&	left to right
^	left to right
|	left to right
&&	left to right
||	left to right
?:	left to right
= * = / = % = + = =	right to left
<< = >> = & = ^= |=	left to right
, (comma)	

***Note:** The unary operations assume higher precedence.*

3.25 CONTROL STRUCTURES

In C++, a large number of functions are used that pass messages, and process the data contained in objects. A function is set up to perform a task. When the task is complex, many different algorithms can be designed to achieve the same goal. Some are simple to comprehend, while others are not.

Functions in C++ 4

Key Concepts

Return types in main() | Function prototyping | Call by reference | Call by value | Return by reference | Inline functions | Default arguments | Constant arguments | Function overloading

4.1 INTRODUCTION

We know that functions play an important role in C program development. Dividing a program into functions is one of the major principles of top-down, structured programming. Another advantage of using functions is that it is possible to reduce the size of a program by calling and using them at different places in the program.

Recall that we have used a syntax similar to the following in developing C programs.

```
void show();              /* Function declaration */
main()
{
      .....
      show();             /* Function call */
      .....
}
void show()               /* Function definition */
{
      .....
      .....               /* Function body */
      .....
}
```

When the function is called, control is transferred to the first statement in the function body. The other statements in the function body are then executed and control returns to the main program when the closing brace is encountered. C++ is no exception. Functions continue to be the building blocks of C++ programs. In fact, C++ has added many new features to functions to make them more reliable and flexible. Like C++ operators, a C++ function can be overloaded to make it perform different tasks depending on the arguments passed to it. Most of these modifications are aimed at meeting the requirements of object-oriented facilities.

In this chapter, we shall briefly discuss the various new features that are added to C++ functions and their implementation.

4.2 THE MAIN FUNCTION

C does not specify any return type for the **main()** function which is the starting point for the execution of a program. The definition of **main()** would look like this:

```
main()
{
        // main program statements
}
```

This is perfectly valid because the **main()** in C does not return any value.

In C++, the **main()** returns a value of type **int** to the operating system. C++, therefore, explicitly defines **main()** as matching one of the following prototypes:

```
int main();
int main(int argc, char * argv[]);
```

The functions that have a return value should use the **return** statement for termination. The **main()** function in C++ is, therefore, defined as follows:

```
int main()
{
    ......
    ......
    return 0;
}
```

Since the return type of functions is **int** by default, the keyword **int** in the **main()** header is optional. Most C++ compilers will generate an error or warning if there is no **return** statement. Turbo C++ issues the warning

```
Function should return a value
```

and then proceeds to compile the program. It is good programming practice to actually return a value from **main()**.

Many operating systems test the return value (called **exit value**) to determine if there is any problem. The normal convention is that an exit value of zero means the program ran successfully, while a non-zero value means there was a problem. The explicit use of a **return(0)** statement will indicate that the program was successfully executed.

4.3 FUNCTION PROTOTYPING

Function prototyping is one of the major improvements added to C++ functions. The prototype describes the function interface to the compiler by giving details such as the number and type of arguments and the type of return values. With function prototyping, a *template* is always used when declaring and defining a function. When a function is called, the compiler uses the template to ensure

that proper arguments are passed, and the return value is treated correctly. Any violation in matching the arguments or the return types will be caught by the compiler at the time of compilation itself. These checks and controls did not exist in the conventional C functions.

Remember, C also uses prototyping. But it was introduced first in C++ by Stroustrup and the success of this feature inspired the ANSI C committee to adopt it. However, there is a major difference in prototyping between C and C++. While C++ makes the prototyping essential, ANSI C makes it optional, perhaps, to preserve the compatibility with classic C.

Function prototype is a *declaration statement* in the calling program and is of the following form:

```
type function-name (argument-list);
```

The *argument-list* contains the types and names of arguments that must be passed to the function.

Example:

```
float volume(int x, float y, float z);
```

Note that each argument variable must be declared independently inside the parentheses. That is, a combined declaration like

```
float volume(int x, float y, z);
```

is illegal.

In a function declaration, the names of the arguments are *dummy* variables and therefore, they are optional. That is, the form

```
float volume(int, float, float);
```

is acceptable at the place of declaration. At this stage, the compiler only checks for the type of arguments when the function is called.

In general, we can either include or exclude the variable names in the argument list of prototypes. The variable names in the prototype just act as placeholders and, therefore, if names are used, they don't have to match the names used in the *function call* or *function definition*.

In the function definition, names are required because the arguments must be referenced inside the function. Example:

```
float volume(int a,float b,float c)
{
     float v = a*b*c;
     .....
     .....
}
```

The function **volume()** can be invoked in a program as follows:

```
float cube1 = volume(b1,w1,h1); // Function call
```

The variable ***b1***, ***w1***, and ***h1*** are known as the actual parameters which specify the dimensions of **cube1**. Their types (which have been declared earlier) should match with the types declared in the prototype. Remember, the calling statement should not include type names in the argument list.