C++ Programming – Lecture 2

Global and Member Functions

- Functions that do not belong to any class are called global functions
- Functions that belong to a class are called member functions of the class
- It is necessary to declare the prototype of the function being called
- C++ compiler uses strict prototype checking to pass a function call as valid

Default Values for Function Arguments

- A global function or a member function can default values for its arguments.
- Default values would get used when specific values are not passed to the function during the function call
- Default values can be assigned only for trailing arguments of a function in its prototype declaration as in:

  void fun (int x, int y, int z = 90, float mul = 3.14) ;

Function Overloading

- Multiple functions carrying similar jobs but differing in arguments can be overloaded
- Overloaded functions have same name, but their arguments differ in number, order or type
- Difference in return types of functions is not a sufficient criterion for functions to be overloaded
- Inline Functions
- Inline functions combine the advantage of function as well as a macro
- A call to inline function is substituted by the code present in its definition
- By marking a function as inline doesn't guarantee that C++ compiler will actually carry out inlining

Constructor Functions

- Short form - Ctor
- Name of Ctor must be same as name of class
- Ctor is a function
- Ctor doesn't return any value

- Ctor gets called automatically when an object is created
- Ctor is called only once during entire lifetime of an object
- Ctor can be overloaded
- Ctor can take default values for its arguments

Static Functions

- Static data members are shared amongst multiple objects
- Static member functions can access only static data members
- Static data member has to be declared inside the class but defined outside it
- Static member function can be accessed using the syntax classname::functionname( )

Operator Overloading

- Operators can be overloaded to make operations on user-defined types more intuitive
- Precedence of operators cannot be changed using operator overloading
- Operators ., :: and ? : cannot be overloaded
- Operator overloading cannot be done for standard types like int or float
- If + operator has been overloaded for Complex number addition, then the expression c = a + b gets expanded to c = a.operator + (b)

Friend Functions

- A function can be declared as a friend of a class
- A friend function can access private data of a class of which it is a friend
- Even though a friend declaration is done within a class, its definition must be done outside the class
- A friend declaration can be made in private or public section of a class

References

- References are constant pointers that get automatically dereferenced
- A reference can be tied with only one variable
- A variable may have multiple references
- A reference to a reference is not possible

- An array of references is not allowed

Types of Function Calls

- Three types of function calls exist in C++ :

  Call by value - pass values of actual arguments
  Call by address - pass addresses of actual arguments
  Call by reference - pass addresses of actual arguments

- Examples of call types :

  swapv (a, b) ; - Call by value
  swapa (&a, &b) ; - call by address
  swapr (a, b) ; - call by reference

- In a call by reference changing formal arguments in the function does not change actual arguments
- In call by address and call by reference, using formal arguments, actual arguments can be changed
- In a function called by address, to reach the actual arguments one has to use pointers and the associated syntax
- In a function called by reference, to reach the actual arguments one has to use references and the associated syntax (simpler)
- Call by address and call by reference are also useful while sending big objects to a function as they do not make a copy of the object being passed