2. INTRODUCTION TO C++

Over view of OOP using C++

C++ is a object-oriented programming language. It was developed by Bjarne stroustrup at AT & T Bell Laboratories in 1980. C++ is an extension of C with a major addition of the class construct feature of simula67. The most important facilities that c++ adds on to C are classes, inheritance, function overloading, and operator overloading. These features enable creating of abstract data types inherit properties from existing data types and support polymorphism, thereby making C++ a truly object-oriented language.

Structure of C++ program

```
Documentation Section
Link Section
void main()
{
 Declaration part
 Executable part
}
```

Documentation Section

Programs written in high level languages are usually easy to read and understand. But, as the program become longer and more complex, even these can become difficult to understand. Hence, it is a good practice to add comment statements to your programs. These statements are only for the programmer’s benefit and are ignored by the compiler.

There are two types of comment lines available in c++.
They are single line comment and multi line comment.

Single line comment

A single line text preceded by a //(double slash) is called single line comment.

Multi line comment

Multi line comment starts with /* and ends with */

Link Section

The runtime library is a collection of object files. Each of these files contains machine instruction for performing group of functions such as I/O (Input and Output), memory management, mathematical operations and strings manipulation. For each group of functions, there is a source file, or a header file. The header file contains information, a compiler needs to perform such functions.

void main Section

The group of statements within main() are executed sequentially. The closing brace of the main() function signals the end of the program. main section contains two parts. Declaration part and executable part. Declaration part declares all the variables present in the executable part. Executable part consists of executable statements. In main section all the statements must and with semicolon.

Program:

```
#include<iostream.h>
#include<conio.h>
void main()
{
 clrscr();
 cout<<"Welcome to c++";
 getch();
}
output:            Welcome to c++
```

3. ELEMENTS OF C++ LANGUAGES

Tokens and Identifiers

Keywords

The keywords implement specific C++ language features. They are explicitly reserved identifiers and cannot be used as names for the program variables or other user-defined program elements.

auto	break	case	catch	char	class
const	continue	default	delete	do	double
else	enum	float	for	friend	goto
if	inline	int	long	new	operator
private	protected	public	register	return	short
signed	sizeof	static	struct	switch	template
this	throw	try	typedef	union	unsigned
virtual	void	volatile	while		

C++ identifiers

Identifiers refer to the names of variables, functions, arrays, classes, etc. created by the programmer. C++ has its own rules for naming these identifiers.

1. Only alphabetic Characters, digits and underscores are permitted.
2. The name cannot start with a digit
3. Upper case and lowercase letters are distinct
4. A declared keyword cannot be used as a variable name.

$$\begin{tikzcd}
 & & \textbf{Constants} \arrow[dl] \arrow[dr] & \\
 & \text{Numeric Constant} \arrow[dl] \arrow[dr] & & \text{Character Constant} \\
\text{Integer Constant} & & \text{Real Constant} &
\end{tikzcd}$$

Character Constant
1. Single Character
2. String constant

Integer Constant
1. Decimal
2. Octal
3. Hexa Decimal

Real Constant
1. Normal form
2. Exponent form

Decimal

Decimal numbers are integer numbers of base 10 and their digits are 0 to 9.

Eg: 23, 670, 35, 89

Octal

Octal numbers are integer numbers of base 8 and their digits are 0 to 7.

Eg: 023, 045, 076

Hexa Decimal

Hexadecimal numbers are integer numbers of base 16 and their digits are 0 to 9 and A to F (a to f).

Eg: 0x56, 0x7f

Real Constant

Real constants are often called Floating Point constants. The real constants could be written in two forms, Fractional form or Exponential form.

Normal Form

5.689, 0.068, 34.78

Exponent form

5.6=5.689e0
0.0678=6.78e-2
34.78=3.478e1

Single character constant

A character represented within single quotes denotes a single character constant.

Eg: 'A' 'p' '6'

String Constant

A string constant is a sequence of alphanumeric characters enclosed in double quotation marks whose maximum length is 255 characters.

Eg: 'The result" "4576" "$%#%#"

Variables

In C++, a quantity, which may vary during program execution, is called a variable. Variable names are names given to the locations in memory of the computer where different constants are stored. These locations can contain character, integer, real, or any such constants.

Data types

Basic data types

Type	Bytes	Range
char	1	-128 to 127
unsigned char	1	0 to 255
signed char	1	-128 to 127
int	2	-32768 to 32767
unsigned int	2	0 to 65535
signed int	2	-32768 to 32767
unsigned short int	2	0 to 65535
short int	2	-32768 to 32767

signed short int	2	-32768 to 32767
long int	4	-214783648 to 214783647
signed long int	4	-214783648 to 214783647
unsigned long int	4	0 to 4294967295
float	4	3.4E-38 to 3.4E+38
double	8	1.7E-308 to 1.7E+308
long double	10	3.4E-4932 to 1.1E+4932

Derived data types

Arrays

Array is used to store different elements of the same data type under a single variable name.

Function

Function groups a set of statements into a single unit and this unit can be called anywhere in our program.

Pointer

Pointer is a variable in which contains the address of the another variable.

User-defined data types

Structure

A structure is a collection of data item or variables of different data types that is reference under the same name.

The declaration of structure is as follows:

```
Struct tag_name
{
 data type members;
};
```

The keyword 'struct' tells the compiler that a structure template is being defined that may be used to create structure variable. The 'tag_name' identifies the particular structure and its type specifier. The fields that comprise the structure are called structure elements'. All elements in a structure are logically related to each other.

Unions

A union is a memory location that is shared by two ore more different variables, generally of different types, at different times. Declaring a union is similar to declaring a structure. Its general form is

```
union union_name
{
 data type members;
};
```

Enumerated Data type

An enumerated type is a type with user-specified values. To declare an enumerated type, we write the keyword enum, followed by

1. An optional identifier of the enumerated type (called a tag), followed by
2. A list of names that are permissible values for this data type. The values are enclosed in braces and separated by commas.

Declaring an enumerated type does not allocate storage but only describes the user specified data type and associates integer constants with the values given in braces. By default, the first values is associated with 0, the second with 1, and third with 2, and so on. A value of an enumerated type may be used anywhere an integer value is expected.

Symbolic constants

In C++, the program in any way cannot modify any value declared as const. As with long and short, if we use that const modifier alone it defaults to int.

Eg:
```
const long=50000;
const size=10;
```

Declaration of variables

In C++, all variables must be declared before they are used in executable statements. C++ allows the declaration of a variable anywhere in the scope. This means that a variable can be declared right at the place of its first use.

Dynamic initialization of variables

C++ permits initialization of the variables at run time. This is referred to as dynamic initialization. In C++, a variable can be initialized at run time using expressions at the place of declaration.

Reference variables

A reference variable provides an alias(alternation name) for a previously defined variable. A reference variable is created as follows:

```
Data_type & reference_name=variable_name;
```

Program :

```
#include<iostream.h>
#include<conio.h>
void main()
{
 int a=10;
 int &b=a;
 clrscr();
 a=a+5;
 cout<<a<<"\t"<<b<<endl;
 getch();
}
```

Console I/O

cin

cin is a predefined object in C++ to correspond to the standard input stream. This stream represents data coming from the input device normally keyboard.

Syntax

```
cin>>variable1>>variable2>>…………>>variable n;
```

cout

The cout displays an object onto the standard device, normally to the screen.

Syntax:

```
cout<<variable1<<variable2<<……….<<variable n;
```