Difference Between Procedure Oriented Programming (POP) & Object Oriented Programming (OOP)

	Procedure Oriented Programming	Object Oriented Programming
Divided Into	In POP, program is divided into small parts called **functions**.	In OOP, program is divided into parts called **objects**.
Importance	In POP, Importance is not given to **data** but to functions as well as **sequence** of actions to be done.	In OOP, Importance is given to the data rather than procedures or functions because it works as a **real world**.
Approach	POP follows **Top Down approach**.	OOP follows **Bottom Up approach**.
Access Specifiers	POP does not have any access specifier.	OOP has access specifiers named Public, Private, Protected, etc.
Data Moving	In POP, Data can move freely from function to function in the system.	In OOP, objects can move and communicate with each other through member functions.
Expansion	To add new data and function in POP is not so easy.	OOP provides an easy way to add new data and function.
Data Access	In POP, Most function uses Global data for sharing that can be accessed freely from function to function in the system.	In OOP, data can not move easily from function to function,it can be kept public or private so we can control the access of data.
Data Hiding	POP does not have any proper way for hiding data so it is **less secure**.	OOP provides Data Hiding so provides **more security**.
Overloading	In POP, Overloading is not possible.	In OOP, overloading is possible in the form of Function Overloading and Operator Overloading.
Examples	Example of POP are : C, VB, FORTRAN, Pascal.	Example of OOP are : C++, JAVA, VB.NET, C#.NET.

Advantages of OOP over POP language -

- OOPs makes development and maintenance easier where as in Procedure-oriented programming language it is not easy to manage if code grows as project size grows.

- OOPs provide data hiding whereas in Procedure-oriented programming language a global data can be accessed from anywhere.

- OOPs provide ability to simulate real-world event much more effectively. We can provide the solution of real word problem if we are using the Object-Oriented Programming language.

Module – II
Basics of C++

First C++ Program -

```
#include <iostream.h>
using namespace std;
void main()
{
   cout << "Hello this is C++";
}
```

- ***Header files*** **-** Header files are included at the beginning just like in C program. Here iostream is a header file which provides us with input & output streams. Header files contained predeclared function libraries, which can be used by users for their ease.

- ***Namespace std*** **-** Using namespace std, tells the compiler to use standard namespace. Namespace collects identifiers used for class, object and variables. Namespace can be used by two ways in a program, either by the use of using statement at the beginning, like we did in above mentioned program or by using name of namespace as prefix before the identifier with scope resolution (::) operator.

C++ Basic Input / Output -

- C++ I/O operation is using the stream concept. Stream is the sequence of bytes or flow of data. It makes the performance fast.
- If bytes flow from main memory to device like printer, display screen, or a network connection etc, this is called as **output operation**.
- If bytes flow from device like printer, display screen, or a network connection etc to main memory, this is called as **input operation**.

- I/O Library Header Files –

Header File	Function and Description
<iostream>	It is used to define the **cout, cin and cerr** objects, which correspond to standard output stream, standard input stream and standard error stream, respectively.
<iomanip>	It is used to declare services useful for performing formatted I/O, such as **setprecision and setw.**
<fstream>	It is used to declare services for user-controlled file processing.

Standard output stream (cout) -

- The cout is a predefined object of ostream class. It is connected with the standard output device, which is usually a display screen. The cout is used in conjunction with stream *insertion operator* (<<) to display the output on a console.

```
#include <iostream.h>
void main()
{
   cout << "Hello this is C++";
}
```

Standard input stream (cin) -

- The cin is a predefined object of istream class. It is connected with the standard input device, which is usually a keyboard. The cin is used in conjunction with stream *extraction operator* (>>) to read the input from a console.

```
#include <iostream.h>
using namespace std;
void main( )
{
      int age;
      cout << "Enter your age: ";
      cin >> age;
      cout << "Your age is: " << age << endl;
}
```

Standard end line (endl) -

- The cin is a predefined object of istream class. It is connected with the standard input device, which is usually a keyboard. The cin is used in conjunction with stream *extraction operator* (>>) to read the input from a console.

```
#include <iostream.h>
using namespace std;
void main( )
{
      int age;
      cout << "Enter your age: ";
      cin >> age;
      cout << "Your age is: " << age << endl;
}
```

Escape Sequence -

- Certain sequences of symbols make special meaning to the computer. They are called escape sequences
- Escape Sequence starts with a backslash (\). It is actually just one special character.
- Useful escape sequences –

Topics	Symbols
New – Line	\n
Horizontal Tab	\t
Alert	\a
Vertical Tab	\v
Backslash	\\
Double Quote	\”

- Example –

```
cout<<“Hello\nWorld”
```

IOSTREAM
IOMANIP
FSTREAM

STREAM IS A SEQUENCE OF BYTES OR FLOW OF DATA

STDIO

CLOG
FORMAT SPECIFIER