Fitur utama dan yang merupakan salah satu alasan bahasa pemrogrman c++ di ciptakan adalah penambahan konsep OOp pada bahasa pemrograman C. setelah perilisa bahasa pemrograman C++, bahasa c++ mendapatkan banyak pengguna yang sama seperti sekarang. bukan hanya itu tapi juga menginspirasi banyak pengembang karena fitur-fitur yang disediakan oleh c++.

Kita sebelumnya telah mengeal dan melihat tentang konsep OOp pada C++ di artikel sebelumnya. Yang memberitahu anda bagaimana cara membuat class dan mendirikan object. bukan hanya itu, kita juga bertemu nama-nama yang cukup asing bagi pemula, seperti constructor, inheritance, polymorphism dan lain-lain. semua itu akan penulis jelaskan secara detail satu persatu pada artikel berikutnya, tapi pada artikel ini kita akan mengenal sedikit mengenai mereka, agar kalian bisa membedakan pada contoh-contoh program pada artikel selanjutnya, mengerti apa yang penulis maksud saat menyebutkan nama yang meskipun anda belum mendapatkan penjelasan detail mengenai nama tersebut dan juga cepat untuk mengerti dalam mempelajari artikel selanjutnya,

**Object Oriented Programming (OOP)** adalah metodologi atau paradigm berdasarkan konsep dari object. konsep pemrograman dalam melakukan pemrograman yang beriontasi kepada object, semua data dan aktifitas dalam pemrograman akan dibungkus dalam object.

<Gambar 1>

**Class (Kelas)** adalah sebuah template / rancangan / blueprint untuk mendefinisikan sifat atau karakter dan perilaku dari object. yang merupakan kumpulan atas definisi dan fungsi-fungsi dalam suatu unit, untuk suatu tujuan tertentu.

**Member (Anggota)** adalah keseluruhan deklarasi yang berada di dalam class.

**Member Variable**, **Field**, **Attribute**, **properties** atau **Data Member**, adalah anggota deklarasi dari class untuk menyimpan data yang berbentuk seperti deklarasi variabel atau object.

**Method** atau **Member Fuction** adalah anggota class yang berbentuk function dengan tugas-tugas di dalamnya yang bisa melakukan suatu tindakan. Seperti halnya function biasa, kita bisa memasukan argument dan mendapatkan data dari method tersebut.

Untuk Member Variabel dan Method memang memiliki banyak nama, tergantung dalam bahasa pemrograman yang dipakai, tapi semua nama memiliki arti yang sama dalam berbagai bahasa pemrograman.

**Akses Specifier** adalah label untuk mengkelompokan dan memberi sifat akses pada member yang mengikutinya, Akses Specifier memiliki 3 sifat yaitu private, protected dan public.

**Object** adalah instance dari class atau representasi dari class, object akan memiliki sifat dan perilaku dari class yang digunakan.

<Gambar 2>

**Instantiation** adalah proses pembuatan object spesifik yang merupakan anggota atau instan dari class, dan instance adalah object tersebut dan instantiaton adalah proses pembuatan dari instance.

**Instance Variable** adalah nama dari dari Member Variable yang sudah melalui proses instantiation untuk membuat instance atau object (Member Variable yang sudah menjadi bagian dari object, dan masing-masing object memiliki variable mereka sendiri), dan hanya dapat diakses melalui object.

**Static Variable** atau **Class Variable** adalah member variabel yang berlaku untuk semua object yang menggunakan class tersebut. (semua object akan memiliki nilai yang sama pada anggota itu). Static Variable juga dapat diakses tanpa melalui object.

**Instance Method** adalahnama darimember function atau method yang telah melawati tahap instantiation untuk pembuatan instance atau object (Method yang sudah menjadi bagian dari object, dan masing masing object memliki method mereka sendiri). Dan instance method hanya bisa diakses melalui object.

**Static Function** adalah member function atau method yang dimiliki oleh class (bukan object) dan juga satu function akan berlaku untuk semua object yang menggunakan class tersebut (object tidak memiliki function itu sendiri). Static Function dapat diakses melalui object atau tanpa melalui object. Saat mendefinisikan static function kita hanya bisa memanggil/mengakses member yang berbentuk static.

**Constructor** adalah method yang berfungsi untuk membangun class (biasanya digunakan untuk inisialisasi pada member), yang akan dijalankan pertama kali saat pembuatan atau instantiaton object.

**Destructor** adalah method yang berfungsi untuk penutup class, yang akan dijalankan terakhir kali saat program berakhir atau disaat penghapusan memori object tersebut (jika menggunakan pointer).

**Encapsulation** adalah konsep pemanfaatan dari akses specifier, membuat beberapa member dibungkus dan tidak sembarangan perintah untuk melihat atau mengubah member yang seharusnya tidak bisa diakses secara langsung,

<!--Biasanya member dibuat menjadi sangat rahasia dan tidak ada yang bisa mengakses maupun melihat tapi ada juga pemanfaatan membuat member menjadi tidak bisa diakses atau diubah tetapi kita masih bisa melihatnya. -->

**Inheritance** adalah konsep pewarisan class. Class juga dapat menuruni dan memiliki apa yang dimiliki oleh class lainya. dan dapat dilakukan secara bertingkat, sehingga semakin ke bawah, kelas tersebut semakin spesifik.

**Abstraction** adalah kemampuan sebuah program juga untuk melenkwati aspek informasi yang diprosesnya, yaitu kemampuan untuk memfokuskan pada inti. Setiap objek dalam system dapat melayani beberapa perintah seperti memproses data, membuat lapotan, membuat perubahan dan biasanya berkomunikasi dengan object lain, tanpa mengungkapkan bagaimana semua itu terjadi.

**Polymorphism** berasal dari bahasa yunani yang berarti memiliki banyak bentuk, adalah kemampuan dalam menyampaikan pesan tertentu keluar dari hirarki objectnya, dimana object yang berbeda memberikan tanggapan atau respon terhadap pesan yang sama sesuai dengan sifat masing-masing object.