Object Oriented Programming

* Central around objects
* Objects encapsulate data and function together

E.g.:- C++, C#, Java, etc.

VS

Procedural programming

* Focus on procedures or actions

e.g.:- BASIC, C, FORTRAN, Pascal etc.

Class:-

* Description of an object
* Code that specifies

1. Attributes 2) Member Functions

e.g. class Rectangle

{

private:

double length; // both public and private could be in either order

double width;

string color;

public:

void setLength(double);

void setWidth(double); //Function will not modify object data

void setColor(string);

double getWidth() const;

double getLength() const;

string getColor() const;

double getArea() const;

};

Access Specifier

Private – Cannot be accessed directly

Public - 1) Member Functions can be accessed outside the class

1. Provides interface to Rectangle objects

:: -> Scope resolution operator

* Separates class name to function name

Types of Functions

* Assessors/getters –retrieve value use count
* Mutators/setters- change values
* Defines an instance of a class:

Rectangle box; //Instantiation

Access an object Member:

Box.setwidth(23.7);

Box.setLength(15.2);

Cout<<box.area()<<endl

**Constructor (initialize object)**

* Method automatically called when object declared
* Function name is same as class
* Void
* Can have parameters
* Can have multiple constructor(overloading)

**Destructor –** called when object goes out of scope

* Release memory, close file etc.

**INLINE FUNCTION**

|  |
| --- |
| class |
| Header |
| File |

DataType.h

Include

|  |
| --- |
| Class  Implementation  File |

DataTypeImp.cpp

|  |
| --- |
| Main  Driver  file |

Main.cpp

**Static variable: -**Exists between function calls

Class- can have static members , Functions or variables