Paradigms of Programming

The Object Oriented Paradigm

Examples: C++, Java, SmallTalk

# Key Features

* Objects are used to represent real-world items.
* State and behaviour are held together in Objects.
* Inheritance used to share common features.
* Uses polymorphism, inheritance, encapsulation and abstraction.

# Description

Object Oriented programming is based around software “objects” which are used to model real-world items.

In simple terms, an object is a structure that has both state and behaviour contained within it.

For example, a Person object would have a state (name, hair colour, eye colour) and behaviour (working, walking, talking) associated with it.

Person {

String name;

String hairColour;

String eyeColour;

Walk(){ //Walking implementation }

Talk(){ //Talking implementation }

Work(){ //Working implementation }

}

Objects interact with each other by the sending and receiving of messages, through method calls exposed in a public API. This public API hides the internal state of the object and allows users to interact with it without worrying about the underlying implementation. This is known as data encapsulation.

Good OO design also focuses on high cohesion (an object serves one purpose) and low coupling (an object is relatively self contained and relies on other objects as little as possible).

# A bit more about it:

One advantage of OO is that it allows programmers to create modular code that easily allows the adding and modification of code without interfering with the program’s existing functionality. A well-written object can also be reused in other pieces of software.

Inheritance also plays an important role, as objects can inherit more general behaviours from more generalised objects, allowing behaviour to be reused and consistent.