Object-Oriented Programming (OOP)

This module on object-oriented programming is intended for engineering degree students.

To go through this course, you're expected to have some basic computer programming knowledge (preferably C).

Object-oriented programming (OOP) is a programming technique that evolves from the more common procedural/modular programming. In procedural programming a solution usually revolves around breaking down a task into smaller ones and writing a function (or procedure - hence the word procedural) specifically to deal with each of that (usually single) small task. Thus, you see a lot of functions that takes data structure (usually in form of a pointer) as an argument. In contrast, object-oriented programming focus more on modelling a solution as an object - which can be an instance of a more generic category. As an object, the solution would have properties (merely a variable in procedural programming) that describes the object, and methods (functions in procedural programming) which are processes used to model how an object acts/reacts. Based on this, it is clear that object-oriented programming is a technique and not just language. So, writing a program in C(or using a C++ compiler) still doesn't qualify as implementing object oriented programming - it is still possible to write a solution purely based on modular programming using C++ (especially when the all classes have public members only). The main features of OOP are encapsulation, inheritance and polymorphism. The page on [[wp>object-oriented programming wikipedia]] also list others, but I feel those can actually be sub-categorized under (or stems from) the mentioned features. So, a language that supports these features should qualify as a language that supports OOP (i.e. enables implementing a solution using OOP technique). C and Java are the early popular implementations (Cbecause of its roots in C and Java because.... well, it's Java! Maybe also because it's among the first to promote cross platform implementation). //To be continued...// [[wp>object-oriented programming Read more @ wikipedia]] ===== Structured Programming ===== *coming soon...* ===== Development Environment ===== *coming soon...* ====== Part 00: Introduction to OOP ====== ===== Objective(s): ===== To familiarize with terms and stuffs in object-oriented programming. Also, basic transition from C to C will be demonstrated.

What is OOP?

It is a programming paradigm (or programming style, if you please) that focuses on modelling solution components as objects, rather than concentrating on the solution procedures. Obviously, it is not a new language - but it does need proper language support (i.e. constructs) in order to implement the concept. Some may use a language that supports OOP but still use modular or simply procedural programming.

to be continued...

Why OOP?

1. (lec) course briefing and refresh basic programming
2. (lec) implementation platform revised (c++ instead of java)
3. (lab) development environment (using mingw)
4. (lab) refresh basic programming

5. modular/procedural programming vs object-oriented programming
6. from c to c++
7. basic overview of c++
8. keyword(s): structs, classes
9. basic phonebook application
10. compare modular/procedural vs object-oriented
11. encapsulation
12. keyword(s): private, protected, publid
13. member functions
14. constructor, destructor (actually part of dynamic dispatch?)
15. dynamic dispatch
16. intro to inheritence
17. lab assessment 1
18. access for inherited properties/methods
19. more inheritence... multiple?
20. keyword(s): virtual