PROGRAMMING

Programming is a recent human activity related to planning and problem-solving, but oriented to computing. It involves communication with computers using a programming language. The language is at a high level for communication between humans also.

Junior Java (tm), or JJ (tm), is the programming language that we will use here. It has a simple syntax (grammar, structure), and it conveys the fundamentals simply, without distracting details. It can also be converted into Java in a very quick and easy way. This leads to an efficient way to learn programming and Java. JJ also goes beyond Java in some ways.

Programming is a creative human activity; it is partly a science, partly an art, and partly a craft. It borrows concepts and tools from many areas including mathematics, philosophy, engineering, business, psychology, and linguistics.

Writing programs is closer to writing poetry than to writing prose. The words are carefully chosen to reflect the proper meaning, and they are arranged in particular patterns (with indenting, but usually without rhyming).

Modeling is a significant use of programming; it involves dealing with the "real" world by creating an "abstract" computer world, which has only the relevant real parts. A challenge of programming is to bridge the gap between the real and the abstract, but also to realize that the abstract is not the real. A map is not reality, but it still can be very useful.

Many ways are possible to do programming, just as there are of building a house. The extreme view is that of the the child's fable of 3 pigs building houses using straw, sticks and bricks. The computer equivalent is building out of bits, instructions, and then methods (routines and functions), and beyond to classes, packages, etc. This analogy extends the n pigs construction to walls, rooms, and prefabricated houses.

Paradigms are different ways of viewing something; we will involve a few paradigms and show the shifts from one to another. The paradigms here begin with control-orientation, then shift to data orientation, and finally to a combination of the two known as object-oriented programming. The three paradigms are shown as diagrams on the corresponding page; they involve three ways of looking at a Pay program, but the details are not shown.

Programming involves both actions (control, procedures, functions, methods, operations) and also objects (data, attributes, states, entities). The emphasis often shifts from one to the other of these two.

Control-oriented programming emphasizes the control or actions. It is sometimes called imperative or procedural programming. It is represented by flaw charts or flow blocks, or pseudo-code which describe how the control moves between the various boxes.

Data-oriented programming emphasizes the objects or data. The actions are packaged as "black" boxes with their details hidden inside. It is represented by data flow diagrams (DFDs) which show the data values flowing through various action boxes.

Object-oriented programming tries to treat both (objects and actions) in a unified manner as one entity called a class; it should really be called class-oriented programming. A class is essentially a blueprint or pattern or plan for creating any number of objects.

State-oriented programming is yet another paradigm; it emphasizes states related to the dataand describes changes or transitions between states.

Other programming paradigms are possible, but at this point in beginning programming it is essential to know that there are at least two different ways; there is no one special way!