<https://docs.oracle.com/javase/tutorial/java/javaOO/returnvalue.html>

<https://docs.oracle.com/javase/tutorial/java/javaOO/accesscontrol.html>

<https://docs.oracle.com/javase/tutorial/java/javaOO/classvars.html>

<https://docs.oracle.com/javase/tutorial/java/javaOO/thiskey.html>

<https://docs.oracle.com/javase/tutorial/java/javaOO/initial.html>

<https://docs.oracle.com/javase/tutorial/java/javaOO/summaryclasses.html>

<https://docs.oracle.com/javase/tutorial/java/javaOO/QandE/creating-questions.html>

<https://docs.oracle.com/javase/tutorial/java/javaOO/QandE/objects-questions.html>

Object-oriented Programming

(OOP )

### 1.  Why OOP?

Suppose that you want to assemble your own PC, you go to a hardware store and pick up a motherboard, a processor, some RAMs, a hard disk, a casing, a power supply, and put them together.  You turn on the power, and the PC runs.  You need not worry whether the CPU is 1-core or 6-core; the motherboard is a 4-layer or 6-layer; the hard disk has 4 plates or 6 plates, 3 inches or 5 inches in diameter; the RAM is made in Japan or Korea, and so on. You simply put the hardware components together and expect the machine to run.  Of course, you have to make sure that you have the correct interfaces, i.e., you pick an IDE hard disk rather than a SCSI hard disk, if your motherboard supports only IDE; you have to select RAMs with the correct speed rating, and so on.  Nevertheless, it is not difficult to set up a machine from hardware components.

Similarly, a car is assembled from parts and components, such as chassis, doors, engine, wheels, brake and transmission. The components are reusable, e.g., a wheel can be used in many cars (of the same specifications).

Hardware, such as computers and cars, are assembled from parts, which are reusable hardware components.

How about software?  Can you "assemble" a software application by picking a routine here, a routine there, and expect the program to run?  The answer is obviously NO!  Unlike hardware, it is very difficult to "assemble" an application from software components.  Since the advent of computer 70 years ago, we have written tons and tons of programs and routines.  However, for each new application, we have to re-invent the wheels and write the program from scratch!

Why re-invent the wheels? Why re-writing codes? Can you write better codes than those codes written by the experts?