**chap 2 The first program : Little Crab**

**Objectives**

* Understand method call, passing parameters.
* Know how to use if statement and sequential structure

**Schedule:** This lesson covers Moodle folder “chap 2”. Please follow the steps below.

1, work on sec 2.1 ~ 2.4, and learn how to write method and pass parameter to method. Scenarios covered in this chapter: little-crab, fatcat.

2, be aware of the inheritance principle of OOP by studying the example of class Crab extended from class Actor.

3, unzip file: ***“OOPIntroPPT-JeepSourceCode.zip”***, and you will find one power point slide “***OOP-BasicIntro.ppt***” and a java file “***OperatingJeepGUI.java***”.

* Explain OOP principle “**data abstraction”** and “**encapsulation**” via the power point slide.
* Run the Java source code ***OperatingJeepGUI.java***. in Eclipse. If you don’t know how, you need to read step 8 in file “chap1-schedule.docx” to run java application in Eclipse. To install Eclipse, follow the instructions in step 7 in file “chap1-scheudle.docx”.
* In summary, **data abstraction** means to generalize the data members and the surrounding methods to form a class from the physical world.
* In summary, **encapsulatio**n means to protect (or, in other words, to encapsulate) the private data with public method, so that all accesses to the private data have to go through the public methods. Thus in the application class, a direct access to the private data without the public method will be considered a violation of the encapsulation rule and is deemed impossible or illegal (in other words, not supported).

4, follow the instructions in file “**homework2.docx**”, and work on homework 2. After you finish it, you need to submit the solution zip file to its Moodle drop box. When coding your homework, please follow all the rules in file “RulesForIndentAndAlignCode.docx”.

5, In Java, variable includes primitive type **variable** and class type **object** (another name is class type **reference)**.

Thus we can say that an object is a variable, but a primitive type variable is NOT an object.

For example, in this statement below, age is a primitive type variable, but not an object or reference, because its type int is an primitive type, not a class type.

int age = 18;

In comparison, in this statement below, kitty is an object, a reference, also kitty can be called as a variable, because its type MyCat is a class type, not a primitive data type in Java.

MyCat kitty = new MyCat();

Study textbook Appendix D.1 , which lists 8 different primitive data types, and they are:

**Four** integer types: **byte, short, int, long**

**Two** floating point type or real numbers: **float, double**

**One** bool type: **boolean**

**One** character type : **char**

for integer types, understand the **size** and **range** of each type

for floating point type, understand their **range** and **precision**, and type conversions between float and double;

for bool type, understand its **value** **range** are ***true*** and ***false***

for character type, understand that java uses Unicode for ***char*** type. Unicode has two bytes, and it uses **\u** representation, for example, **\u0041** is the Unicode coding for character 'A'.

In comparison, c++ uses ASCII code for its char type. ASCII code has one byte.

For further discussion about Unicode and ASCII code, please refer to “Test1StudyGuide.docx”, page 3 and 4.

6. keep working on the questions in file “Test1StudyGuide.docx”, and prepare for test 1. This file is available in Moodle folder “test 1 review lesson”.

7, when we code in Java, we need to follow its naming conventions

* Java naming convention follows CamelCase (or UpperCamelCase ), a naming convention in which a name is formed of multiple words that are joined together as a single word with the first letter of each of the multiple words capitalized, so that each word that makes up the name can easily be read.
* Java also follows lowerCamelCase, in which the first letter of the first word is lowercase of the CamelCase.
* Names that start with lower case letter ( lowerCamelCase ): variable/object/instance name, package name, method (except constructor) name.
* Names that start with upper case letter ( UpperCamelCase ): class/interface name, constructor name.
* Names that have to be all upper-case letters, with underscore connecting adjacent words: static final variables, in other words, constant variables.

We haven’t covered some of the concepts mentioned above yet, such as , static, final, interface. But don’t worry, and we will cover them later in this semester.

8, next lesson, we will be working on Moodle folder “chap 3”, please preview it.