|  |  |  |
| --- | --- | --- |
|  | Field (attribute) | Non-static field (instance variable): unique for each object  Static field (common feature): same value for every object |
| Class | Method (functionality) |  |
|  | Constructor (construct an object)  Constructor: same name with class name; construct an object; doesn’t have return type;  used to initialize each instance variable, you can change the value of any instance variable later by reassignment. Such as s1.ID=”0009”; then the student ID of student object s1 will get changed to “0009” and save it into the encapsulated object, when you output s1.ID again, it will show you the new value “0009”. | Constructor is used to construct/build an object.  Inside the constructor, it will initialize the instance variables.  If you don’t write any constructor, the system will create a default constructor for you, and the default constructor created by the system will initialize the String type variable to be empty string and int type to be 0, if it doesn’t mention one of the instance variable, it will get a default value.  Default constructor is the constructor without any parameters. (empty parenthesis)  Rule:  If you didn’t create any constructor, the system will create a default constructor for you; once you ever create any constructor, then the system won’t create any constructor for you at all. (the system will not create the default constructor for you anymore) |

Example:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Field: object-level instance variable: Name (string), ID (string), GPA (double)  static field: school (string): Queens College | |  |  | | --- | --- | |  | Name:Ada | | Student s1 | ID:0001 | |  | GPA:4.0 |  |  |  | | --- | --- | |  | Name:Lucy | | Student s2 | ID:0002 | |  | GPA:3.0 |   Both students share same college |
| Student | Method: register ( ) , drop ( ), evaluate ( ) | s1.register()… |
|  | Default constructor | You can create your own default constructor as following:  Student ( ){  ID=”000”;  Name=”null”;  GPA=0.0;  } |
|  | You can also create the other constructors, you can create a constructor with all the instance variables as the parameters, or one/two parameter constructor. | Student (String I, String n, double g) {  ID = I;  Name = n;  GPA = g;  }  How to call it?  The keyword new will invoke the constructor.  Student S1= new Student (“0001”, “Ada”, 4.0); |

Object Oriented Programming has an important feature: Encapsulation

Each object is encapsulated.

|  |  |
| --- | --- |
| S1 | |
| ID “0001” | fields |
| Name “Ada” |
| GPA 4.0 |
| Register() | methods |
| Drop() |
| Evaluate() |

Question 1: which constructor the following statement called? At which situation you can do this? Which situation you couldn’t do it?

Student s0; // It calls the default constructor

Student s0= new Student ( ); //it calls the default constructor as well

When you have a default constructor (either the system creates one for you or you create by yourself, you can use it, if you don’t have the default constructor, then you can’t use the default constructor, otherwise it will throw the exceptions(errors))

Student s0 = new Student ( ); NOT POSSIBLE if you **ONLY** create a single constructor with 3 parameters

Question 2: How to access static variable and instance variable?

For static variable, you can either access it use object to access it or Class to access it directly.

eg. Student.school (because it is a static variable which is class level ) or s1.school

For instance variable, you can access it **ONLY** by the object.

eg.   
S1. ID

S1.GPA

S1. Name

But **Student.ID** is not allowed, because we don’t know which student it asks for.