Review

Operator:

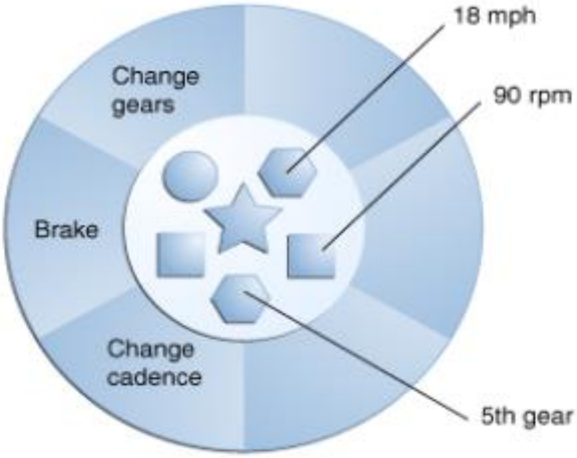
1. int a =3;
   int b=4;
   a==b a 多少？？
   a=b a 多少？？
2. Casting：4 round-off：8
3. 单目算数关系逻辑赋值 1+8>7-8 && 1-1<3+7 练习 5
4. Control structure
   int a =90;
   If (a>80){ System.out.println(“a>80”);}
   If(a>90){ System.out.println(“a>90”);}
   Else{ System.out.println(“a<80”);}
5. Logical operator 11 12
6. Hexadecimal---decimal 18
7. Program 21(eclipse: learn debug) 25
8. http://www.oracle.com/technetwork/java/javase/documentation/api-jsp-136079.html

Chapter2 Classes and Objects

Object and class

- Object

  An object is a software bundle of related state and behavior.



A bicycle modeled as a software object.

- Class

- Blueprint for implementing objects.an object is a single instance of the class
- In java, a variable that represents an object is called an object reference

- Encapsulation

- In class, state------data fields
- Behavior----methods
- Combine data and method into a single unit class ----encapsulation

- Bundling code into individual software objects provides a number of benefits, including:

1. Modularity: The source code for an object can be written and maintained independently of the source code for other objects. Once created, an object can be easily passed around inside the system.
2. Information-hiding: By interacting only with an object's methods, the details of its internal implementation remain hidden from the outside world.
3. Code re-use: If an object already exists (perhaps written by another software developer), you can use that object in your program. This allows specialists to implement/test/debug complex, task-specific objects, which you can then trust to run in your own code.
4. Pluggability and debugging ease: If a particular object turns out to be problematic, you can simply remove it from your application and plug in a different object as its replacement. This is analogous to fixing mechanical problems in the real world. If a bolt breaks, you replace *it*, not the entire machine.

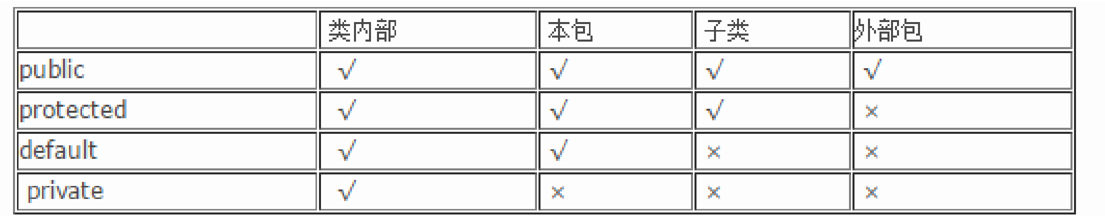
Keywords—access specifier

- Public

- Public class is usable by all client programs. Else, only be used by class in its own package.
- Public methods are accessible to all client program.

- Private

Private variables can be access only by methods of that class.



	类内部	本包	子类	外部包
public	√	√	√	√
protected	√	√	√	×
default	√	√	×	×
private	√	×	×	×

- Static

- A static variable contains a value that is shared by all instances of the class.
- Memory allocation happens once.

Information hiding---restriction of access

However, client are not privy to the class implementation and access the private variables and methods.

Comparison between static and non-static :

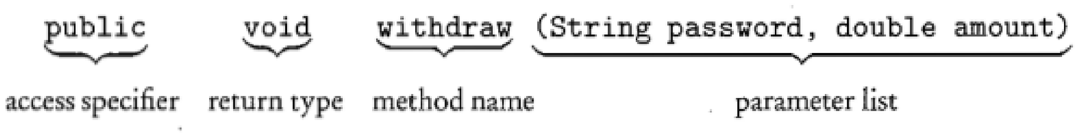
1. When the class is loaded member variable is initialized, associated with the class.
2. A static variable separately divided a storage space, the storage space is shared by all objects of

a class.

3. When created a new object, non-static variables are divided storage space.

4. className.staticVariable   objectName.nonstaticVariable

Methods



$$\underbrace{\texttt{public}}_{\text{access specifier}}\quad\underbrace{\texttt{void}}_{\text{return type}}\quad\underbrace{\texttt{withdraw}}_{\text{method name}}\quad\underbrace{\texttt{(String password, double amount)}}_{\text{parameter list}}$$

Type of methods

➢ Constructors

- Create an object of the class. The same name as the class and no return type.
- New operator returns the address of newly constructed object.

eg: BankAccount b=new BankAccount()

b stores the address of BankAccount object. Not the object itself.

➢ ACCESSORS

Access a class object without altering the object. Return some information about object

➢ MUTATORS

- Change the state of an object by modifying at least one of its instance variables.
- Dot operator b.withdraw();

Static methods vs instance methods

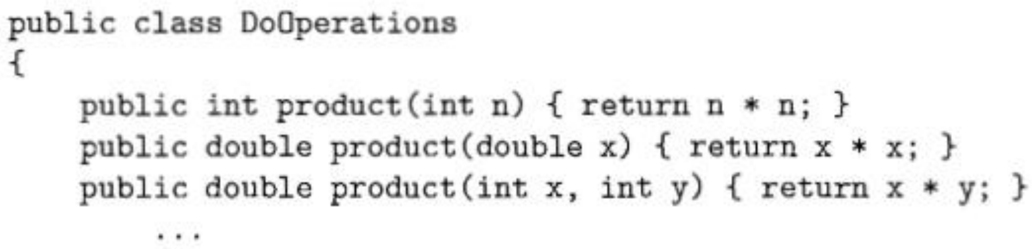
1. When the class is loaded member variable is initialized, associated with the class.
   A static variable separately divided a storage space, the storage space is shared by all objects of a class.
2. When created a new object, non-static variables are divided storage space.
3. Instance methods: all operate on individual object of a class.
4. Static methods: a method that performs an operation for the entire class. Not individual object.
5. Recall instance method, object.methodname()
   Recall static methods, classname.methodname();

Static methods in a driver class

- A class contains main() is used to test other class.
- Create no objects.

Method overloading

In the same class have the same name but different parameter lists.



```
public class DoOperations
{
    public int product(int n) { return n * n; }
    public double product(double x) { return x * x; }
    public double product(int x, int y) { return x * y; }
        ...
```

Method's signature

- Method's name and a list of parameter types.
- The return type of the method is irrelevant.

- Error: two methods with identical signature but different return

Scope:

- The region in which that variable or method is visible and can be accessed.
- Instance variables, static variable, and methods belong to class's scope.
- Local variable is defined inside a method. Automatically recycled.
- Local variables take precedence over instance variables with the same name.

The this keyword

Reference vs primitive data types

- Primitive data type: int, double…
- Reference data types: object
- The way they are stored is different

✓ Aliasing

Two references for the same object.

Use new to create a second object

✓ The null reference

An uninitialized object variable

Test:BankAccount b; If(b==null)

✓ NullpointerException

Method parameters

✓ Formal vs Actual Parameters

- Formal--The header of a method defines the parameters of that method. Placeholder
- Actual—supplied by a particular method call in client program.

✓ Passing primitive types as parameters

Passed by value/ a new memory slot

✓ Passing objects as parameters

Copy the address

Summary

課後作業：

1，2，