Packages

- Why do programming languages provide different accessibilities (i.e. `private` and `public` qualifiers):
  - A large computer program has many methods stored in different classes
  - Methods defined inside a class are used to solve the same problem
  - Methods defined inside different classes usually do not solve the same problem
  - A common cause for errors is **accidental update** of variable(s) by a method in a different class
  - Limiting access to variables (with the `private` qualifiers) will reduce accidental update of variable(s) by a method from another class
- Java packages:
  - Some problems are too complex that they cannot be solved by methods inside one single class
    - We may need to write multiple classes to solve such complex problems
      - Programs will run more efficiently if they can access variables directly
      - I.e., Getter (`getRadius()`) and setter (`setRadius()`) will slow down a program

> **Package** in Java is used to organize multiple classes that cooperate and perform similar tasks

  - It facilitates (=make it easier) the cooperation:
    - Java can allow methods defined in classes inside the same package to access each other members (variables and methods) directly
- Example on using a package

```
package p1;
public class C1{
    public int x1;
           int x2; // Default (= package) access modifier
           //
    private int x3;

    public void m1() { }
           void m2() { } // Default (= package) access modifier
    private void m3() { }
}
```

- The numbers `x2` and `m2()` has no access modifier. In that case, they will be assigned with the default access modifier.
  - A member with the default access can be accessed from methods in classes belong to the same package. - Class in a same package cooperate to perform a task
  - A member with the default access is not accessible from methods in classes belong to a different package. - Class in different packages do not cooperate to perform a task
-
  - See Demo.java