OBJECT ORIENTED PROGRAMMING

Class Member Accessibility

by
Dr. Nor Saradatul Akmar Zulkifli
Faculty of Computer Systems & Software Engineering
saradatulakmar@ump.edu.my

Content Overview

- Accessibility Modifiers
  - Private , public and protected modifier
  - Example
  - Effect of each modifier
- Accessor and Mutator
  - Accessor (Getter)
  - Mutator (Setter)
  - Example

Learning Objectives

- To understand the concept of Public, Private and Protected
- To describe the effect of private and public access to data and methods
- To understand accessor and mutator methods

ACCESSIBILITY MODIFIER

- Purpose : To determine the right access for class, object's data and methods.
- The class, variable and method can be accessed by any class in the same package by default.
- There are 2 common **accessibility modifier** used in a program:

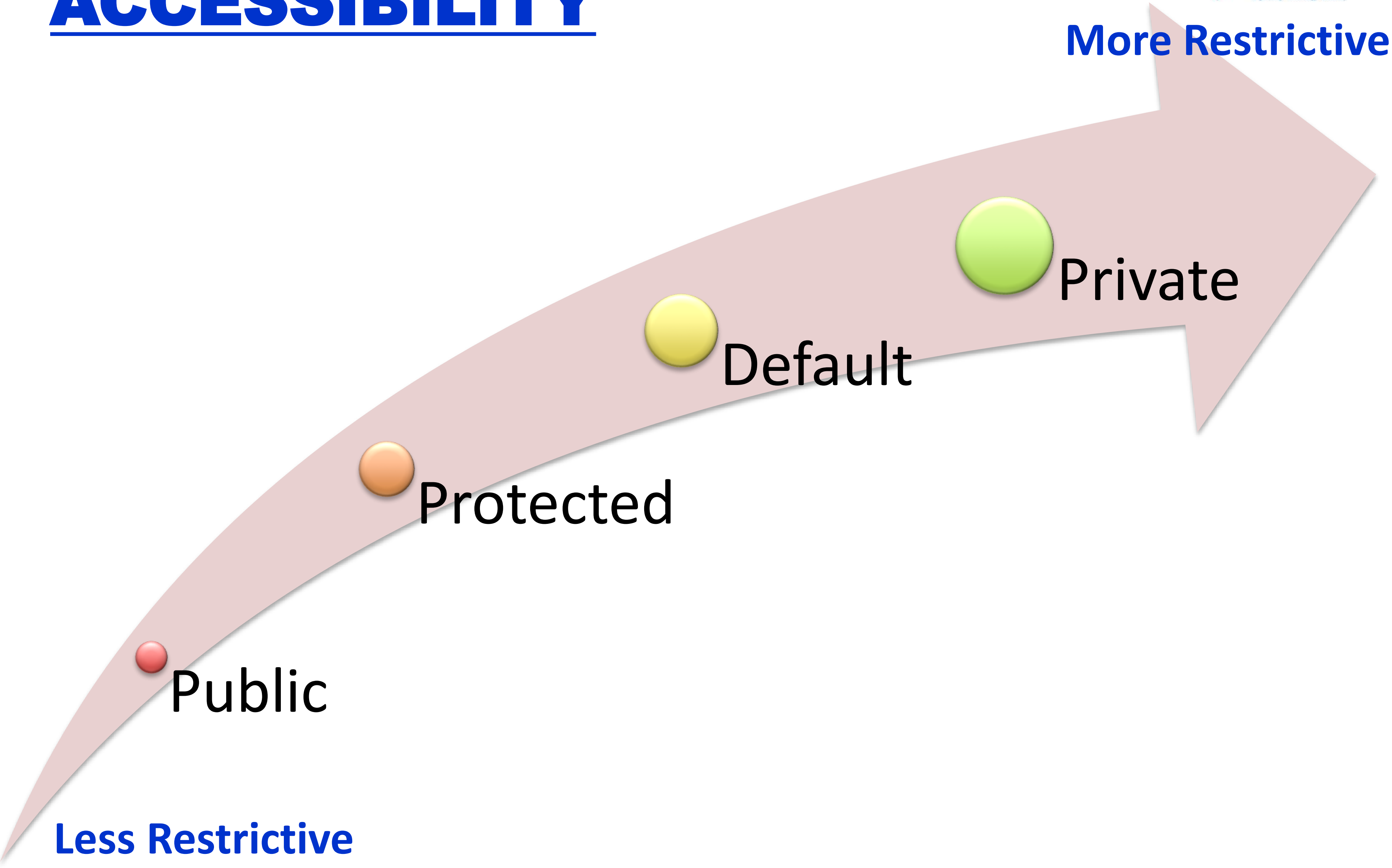
PUBLIC	**PRIVATE**
Both data and method are visible to any class in any package	Both data and method can only be accessed by the declaring class

ADDITIONAL

PROTECTED

Accessible only to the methods that belong to the same class or to the descendant classes AND inaccessible to the methods of an unrelated class

ACCESSIBILITY



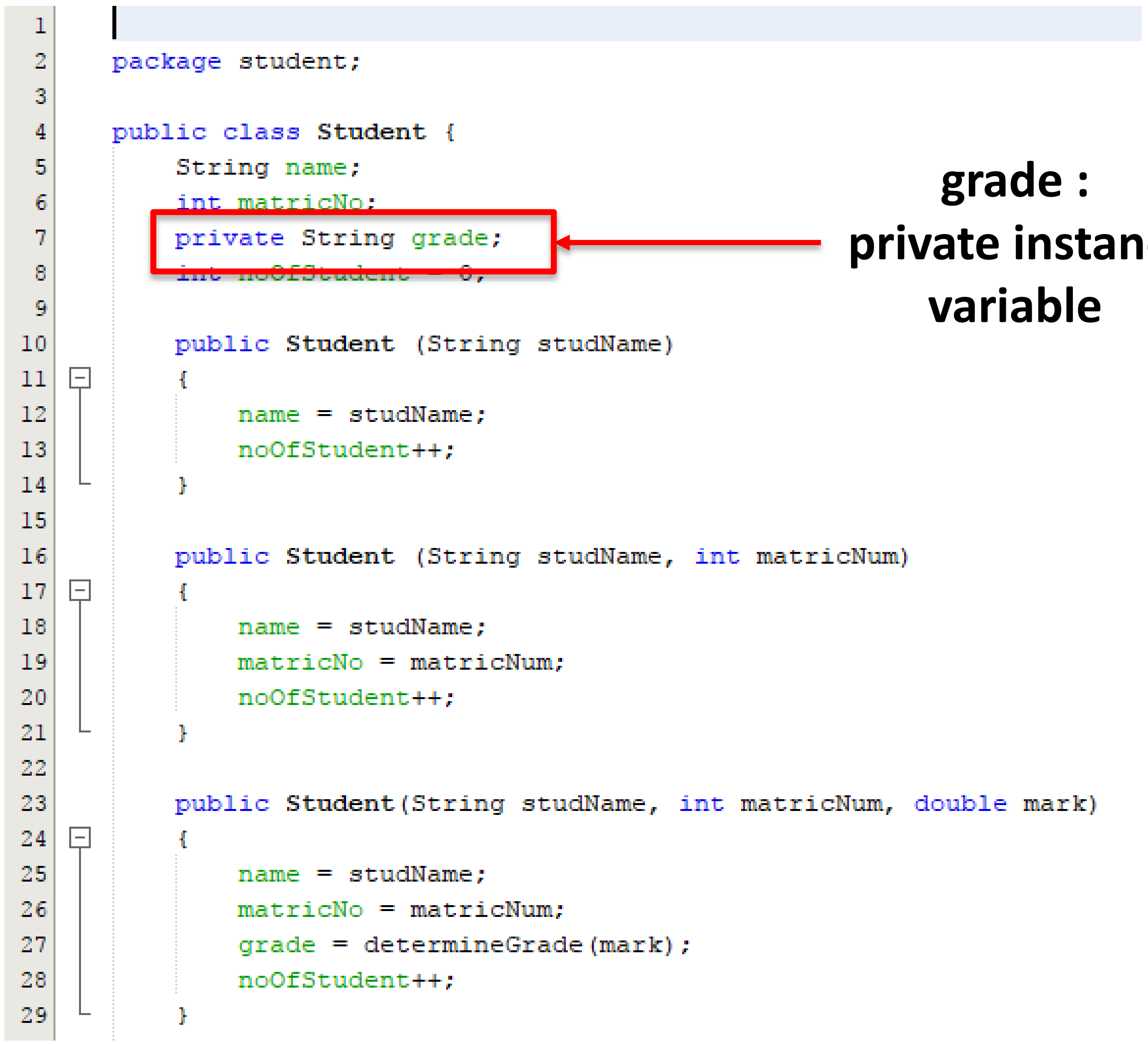
ACCESSIBILITY MODIFIER : PRIVATE

Universiti Malaysia PAHANG
Engineering • Technology • Creativity

- To **prevent** the client program (main) **directly access to the instance variable(s)**, it's being encapsulate by the private modifier.
- Used to **enforce information hiding** by making instance variable(s) private.
- Method(s) declared as private when it is **only to be accessible WITHIN the same class**

PRIVATE MODIFIER : EXAMPLE

Class : Student



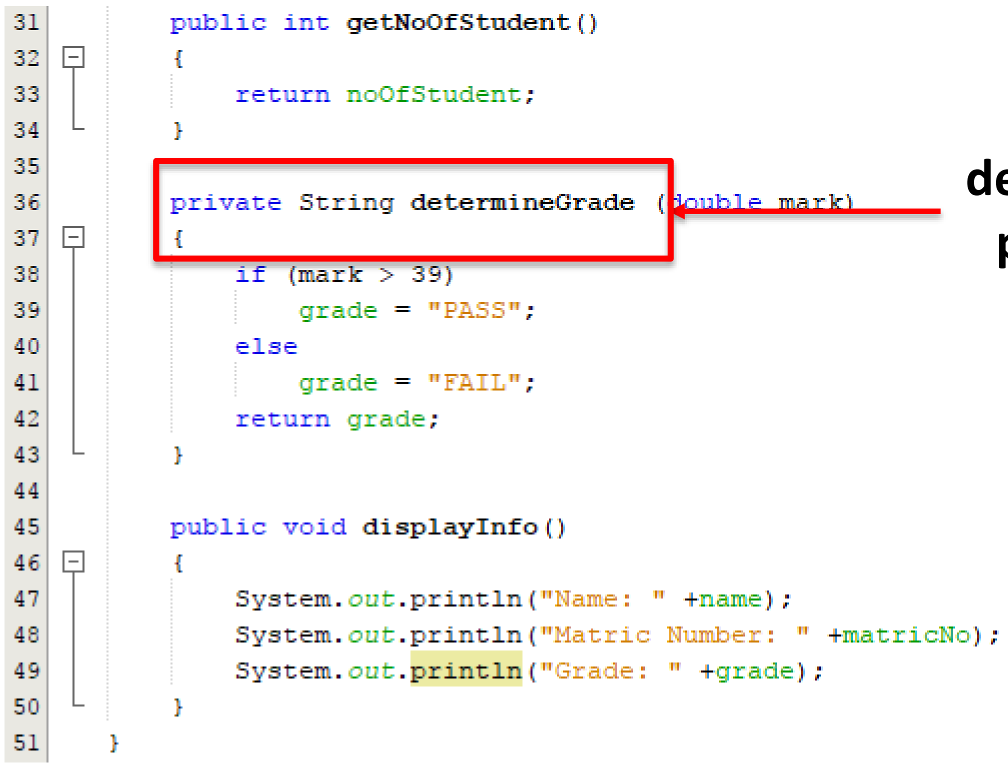
```

package student;

public class Student {
    String name;
    int matricNo;
    private String grade;
    int noOfStudent = 0;

    public Student (String studName)
    {
        name = studName;
        noOfStudent++;
    }

    public Student (String studName, int matricNum)
    {
        name = studName;
        matricNo = matricNum;
        noOfStudent++;
    }

    public Student(String studName, int matricNum, double mark)
    {
        name = studName;
        matricNo = matricNum;
        grade = determineGrade(mark);
        noOfStudent++;
    }
```

grade : private instance variable

PRIVATE MODIFIER : EXAMPLE

…..Continue

Class : Student

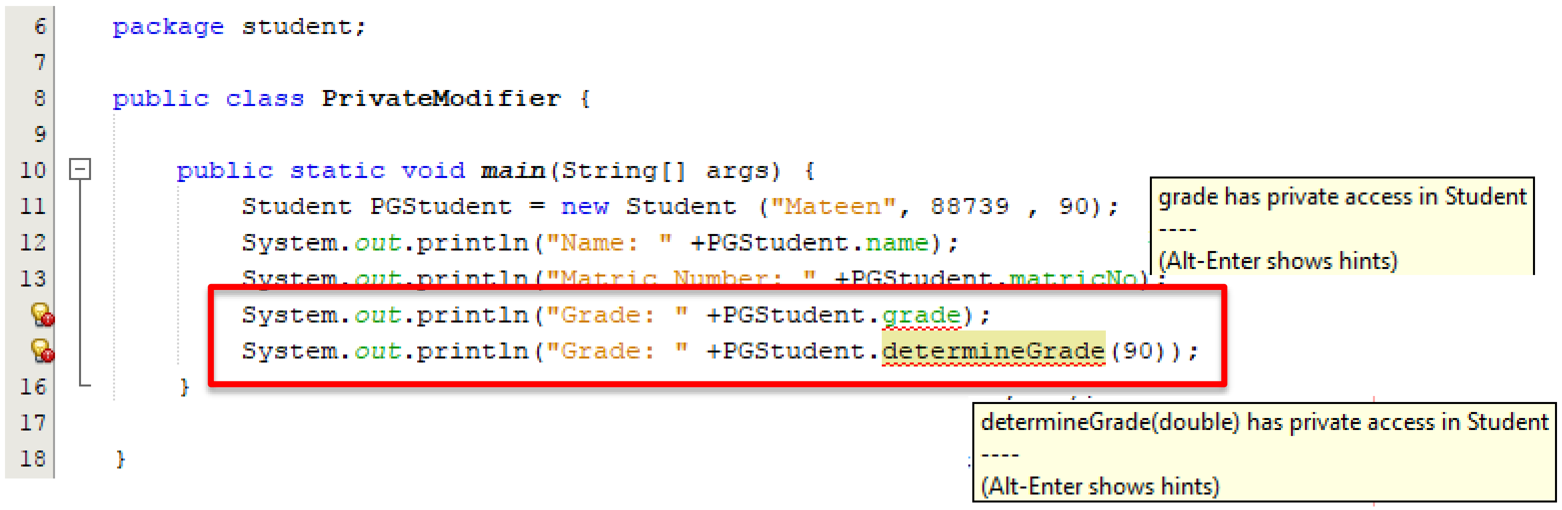


```
    public int getNoOfStudent()
    {
        return noOfStudent;
    }

    private String determineGrade (double mark)
    {
        if (mark > 39)
            grade = "PASS";
        else
            grade = "FAIL";
        return grade;
    }

    public void displayInfo()
    {
        System.out.println("Name: " +name);
        System.out.println("Matric Number: " +matricNo);
        System.out.println("Grade: " +grade);
    }
}
```

determineGrade : private method

PRIVATE MODIFIER : EXAMPLE

Main Class



```
package student;

public class PrivateModifier {

    public static void main(String[] args) {
        Student PGStudent = new Student ("Mateen", 88739 , 90);
        System.out.println("Name: " +PGStudent.name);
        System.out.println("Matric Number: " +PGStudent.matricNo);
        System.out.println("Grade: " +PGStudent.grade);
        System.out.println("Grade: " +PGStudent.determineGrade(90));
    }

}
```

```
grade has private access in Student
----
(Alt-Enter shows hints)
```

```
determineGrade(double) has private access in Student
----
(Alt-Enter shows hints)
```

Variable **grade** and method **determineGrade** declared as private – thus, cannot be access by **PGStudent** object

ACCESSIBILITY MODIFIER : PUBLIC

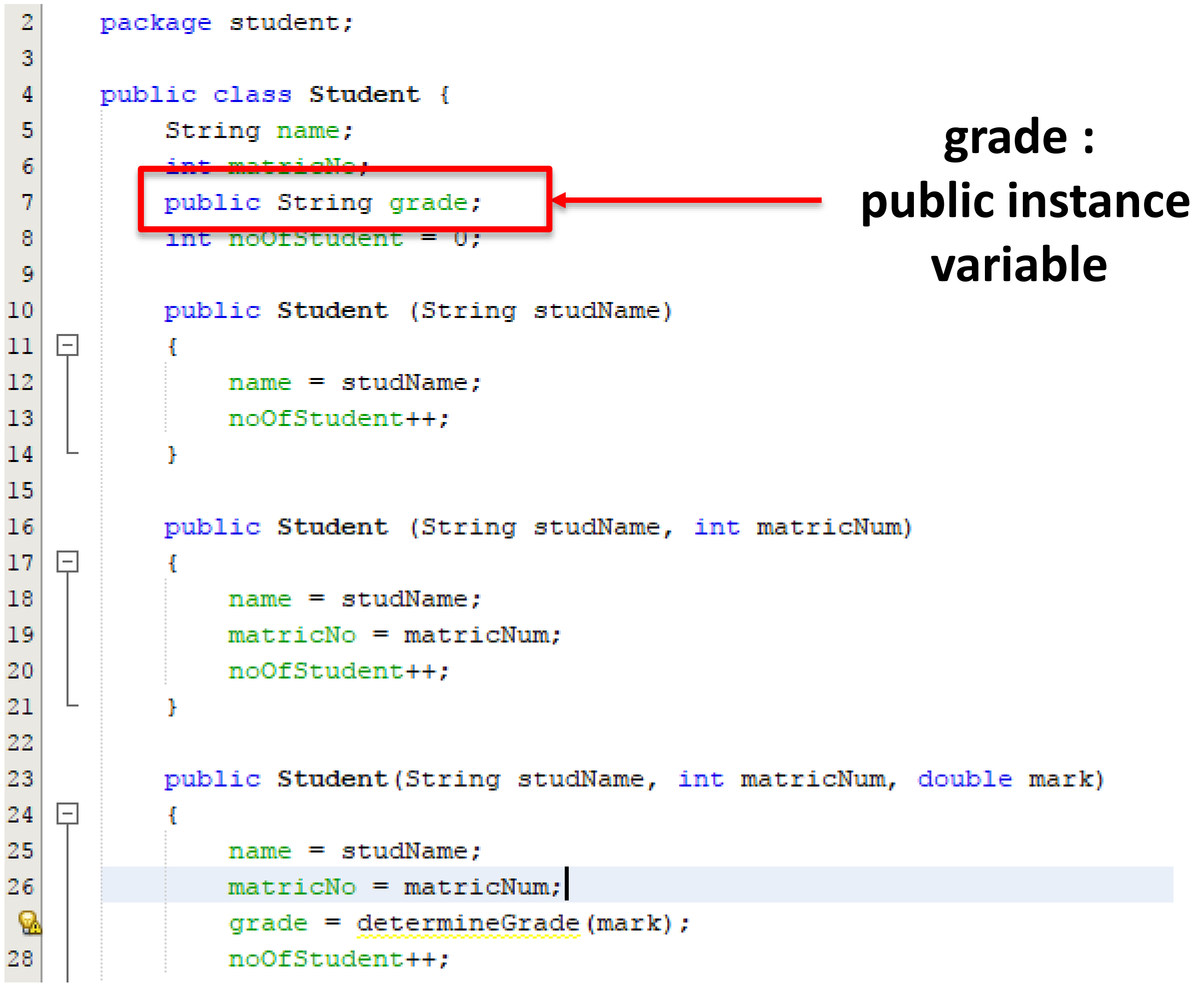
- Public instance variable and method can be accessed in its own class and other classes

- Variable(s) and method(s) that have been declared as `public` have unlimited access control but could violate the object encapsulation principle.

Note :
Using the previous Example. Change variable **`grade`** and method **`determineGrade`** into **`public`**

PUBLIC MODIFIER : EXAMPLE

Class : Student



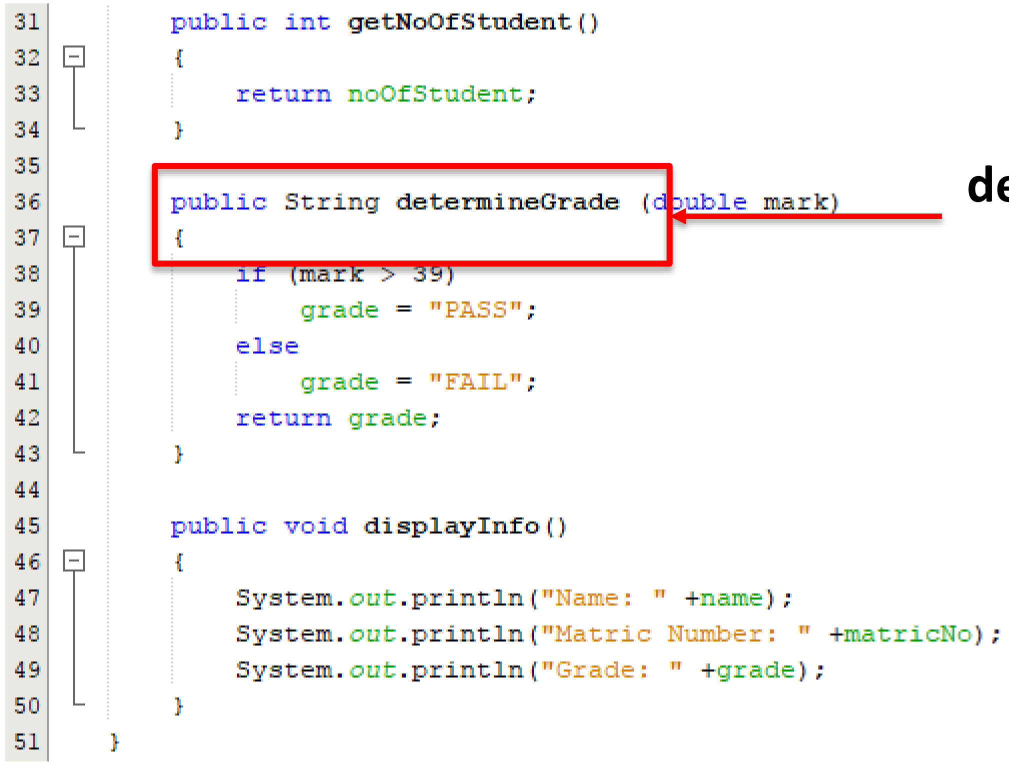
```
package student;

public class Student {
    String name;
    int matricNo;
    public String grade;
    int noOfStudent = 0;

    public Student (String studName)
    {
        name = studName;
        noOfStudent++;
    }

    public Student (String studName, int matricNum)
    {
        name = studName;
        matricNo = matricNum;
        noOfStudent++;
    }

    public Student(String studName, int matricNum, double mark)
    {
        name = studName;
        matricNo = matricNum;
        grade = determineGrade(mark);
        noOfStudent++;
```

grade :
public instance
variable

PUBLIC MODIFIER : EXAMPLE

…..Continue

Class : Student

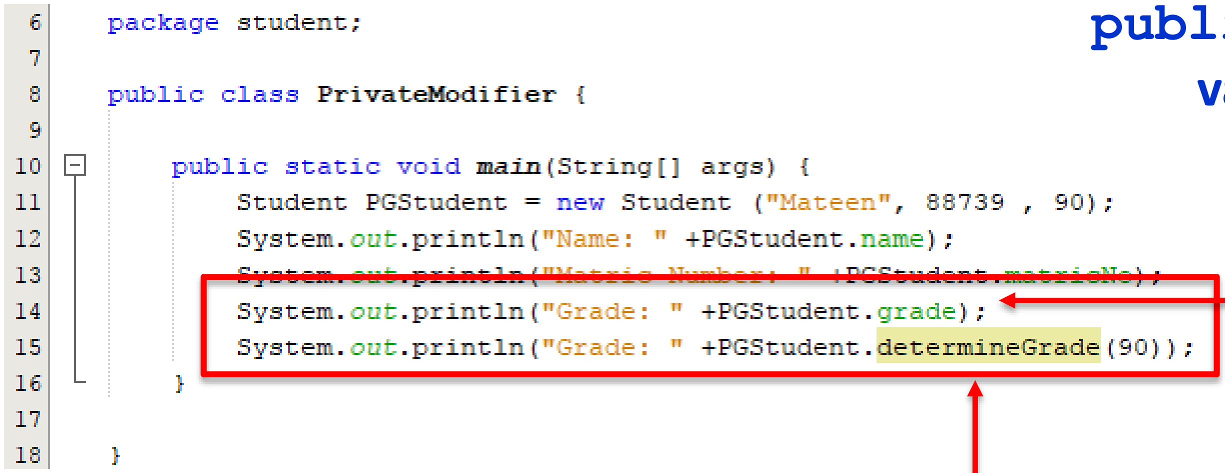


```
    public int getNoOfStudent()
    {
        return noOfStudent;
    }

    public String determineGrade (double mark)
    {
        if (mark > 39)
            grade = "PASS";
        else
            grade = "FAIL";
        return grade;
    }

    public void displayInfo()
    {
        System.out.println("Name:  " +name);
        System.out.println("Matric Number:  " +matricNo);
        System.out.println("Grade:  " +grade);
    }
}
```

determineGrade : public method

PUBLIC MODIFIER : EXAMPLE

Main Class

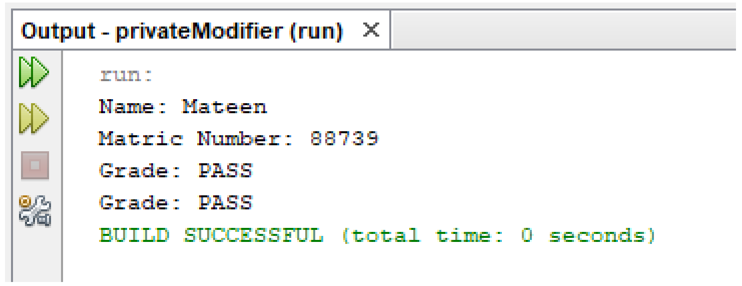


```java
package student;

public class PrivateModifier {

    public static void main(String[] args) {
        Student PGStudent = new Student ("Mateen", 88739 , 90);
        System.out.println("Name: " +PGStudent.name);
        System.out.println("Matric Number: " +PGStudent.matricNo);
        System.out.println("Grade: " +PGStudent.grade);
        System.out.println("Grade: " +PGStudent.determineGrade(90));
    }

}
```

`grade`: access `public` instance variable

`determineGrade`: invoke `public` method

Output

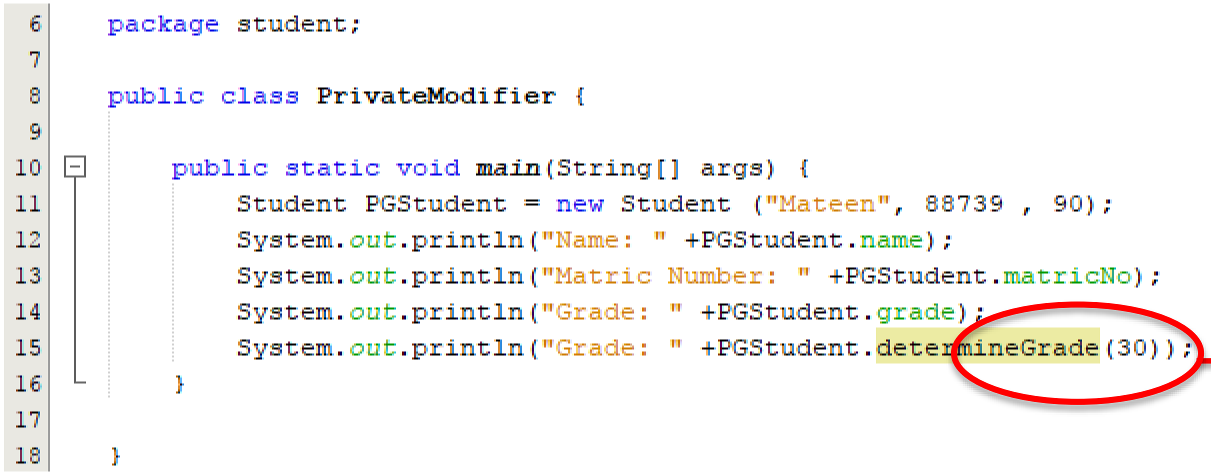


Output - privateModifier (run)

```
run:
Name: Mateen
Matric Number: 88739
Grade: PASS
Grade: PASS
BUILD SUCCESSFUL (total time: 0 seconds)
```

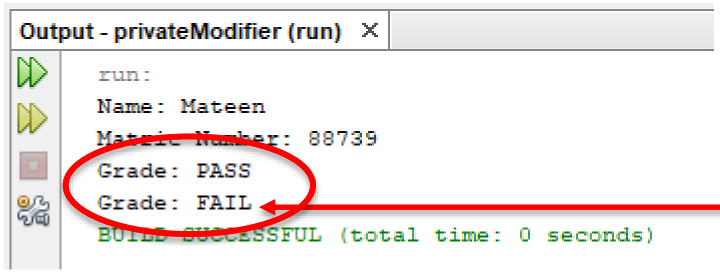
PUBLIC MODIFIER : EXAMPLE ENCAPSULATION ISSUE

Main Class



```java
package student;

public class PrivateModifier {

    public static void main(String[] args) {
        Student PGStudent = new Student ("Mateen", 88739 , 90);
        System.out.println("Name: " +PGStudent.name);
        System.out.println("Matric Number: " +PGStudent.matricNo);
        System.out.println("Grade: " +PGStudent.grade);
        System.out.println("Grade: " +PGStudent.determineGrade(30));
    }

}
```

Output



Output - privateModifier (run)

```
run:
Name: Mateen
Matric Number: 88739
Grade: PASS
Grade: FAIL
BUILD SUCCESSFUL (total time: 0 seconds)
```

PUBLIC MODIFIER : ENCAPSULATION ISSUE

CAN YOU SPOT THE DIFFERENT?

grade variable : **PASS** ⟶ **FAIL**

Why?

Change of **mark** value : **90** ⟶ **30**

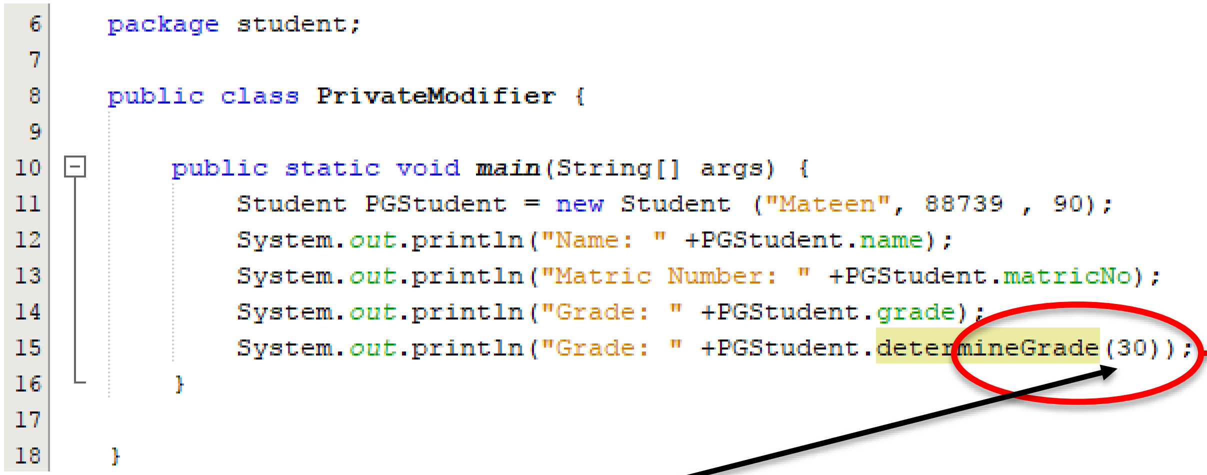
```
System.out.println("Grade:"+PGStudent.determineGrade(30));
```

This code is valid since the method is declare as **public** and **grade** variable is not encapsulate in **Student** object

```
public String determineGrade(double mark)
{ // . . . }
```

PUBLIC MODIFIER : EXAMPLE ENCAPSULATION ISSUE

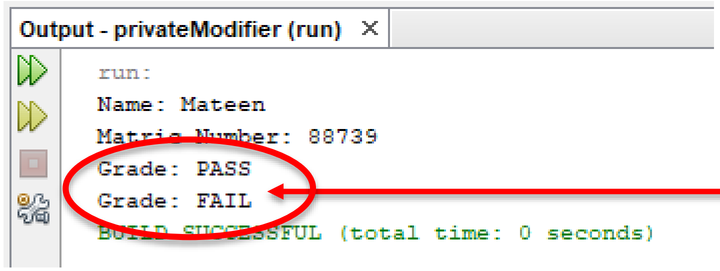
Main Class



```java
package student;

public class PrivateModifier {

    public static void main(String[] args) {
        Student PGStudent = new Student ("Mateen", 88739 , 90);
        System.out.println("Name: " +PGStudent.name);
        System.out.println("Matric Number: " +PGStudent.matricNo);
        System.out.println("Grade: " +PGStudent.grade);
        System.out.println("Grade: " +PGStudent.determineGrade(30));
    }

}
```

The `main ( )` method can **<u>directly change</u>** the `Student` class members

Output



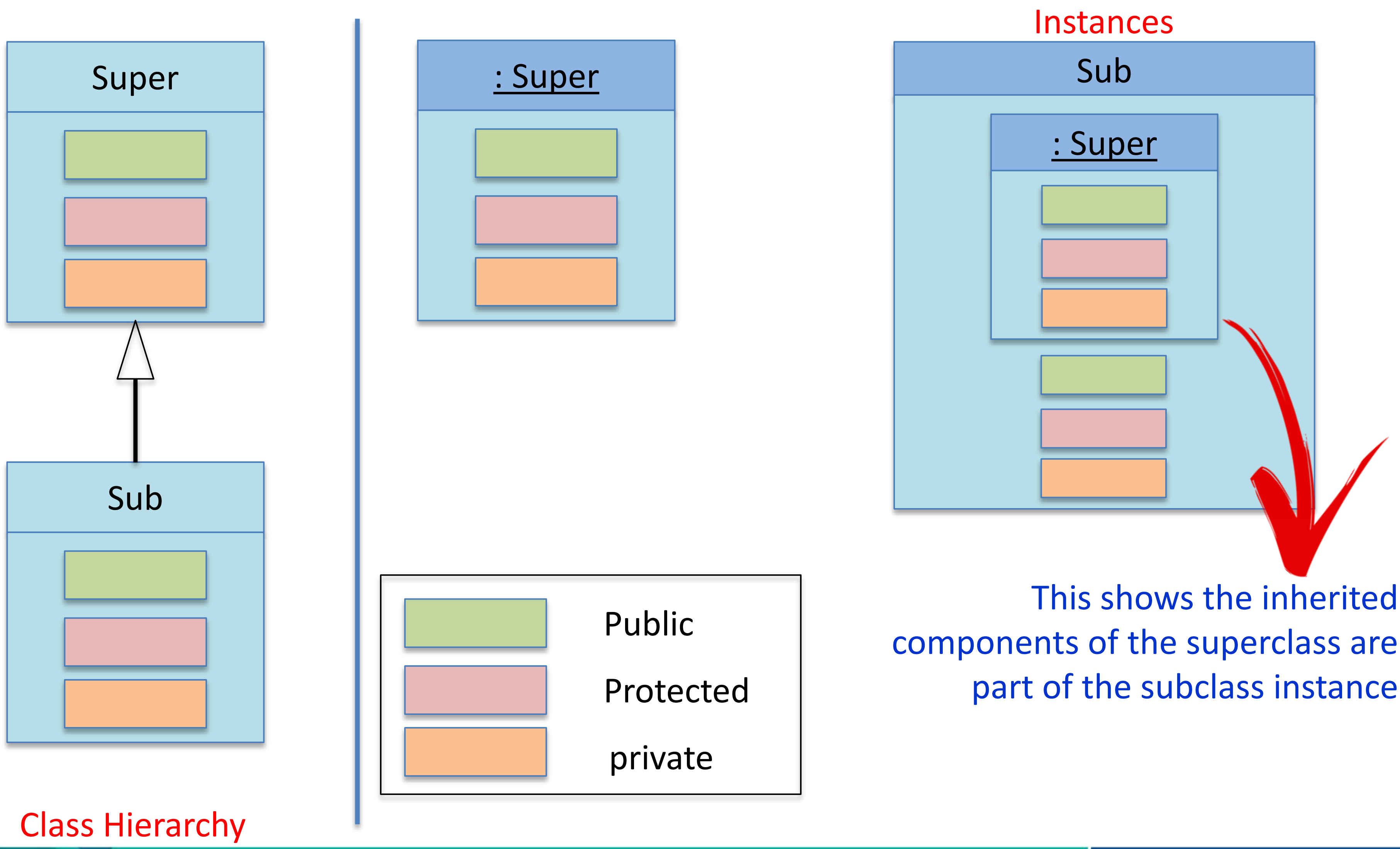
Output - privateModifier (run)

```
run:
Name: Mateen
Matric Number: 88739
Grade: PASS
Grade: FAIL
BUILD SUCCESSFUL (total time: 0 seconds)
```

Solution:

`determineGrade` method should encapsulate in a class – `private` it.

ACCESSIBILITY MODIFIER : Graphical Representation

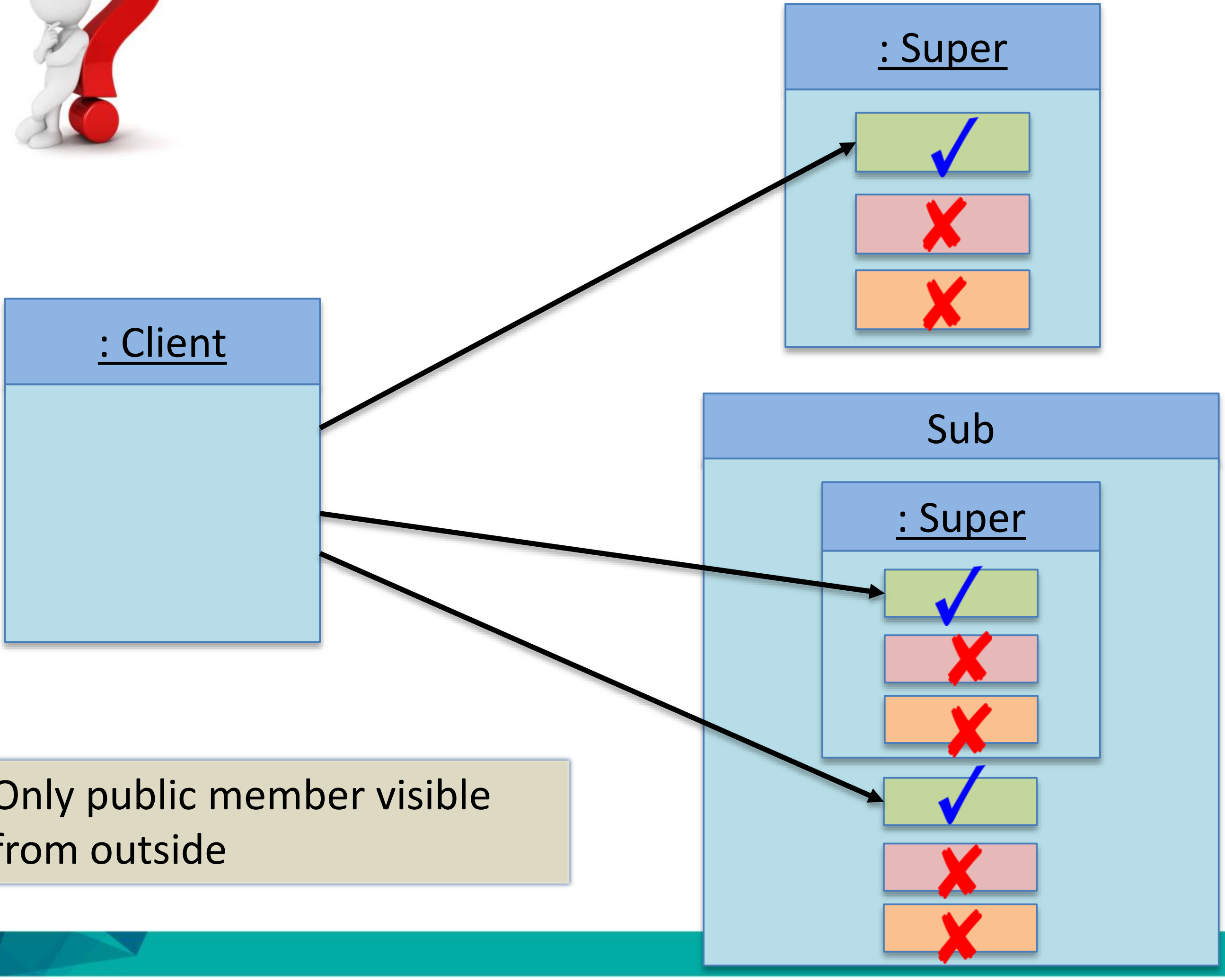


🟩	Public
🟥	Protected
🟧	private

This shows the inherited components of the superclass are part of the subclass instance

ACCESSIBILITY MODIFIER : EFFECT

How These modifiers (public, private and protected) differ?



✓ Accessible

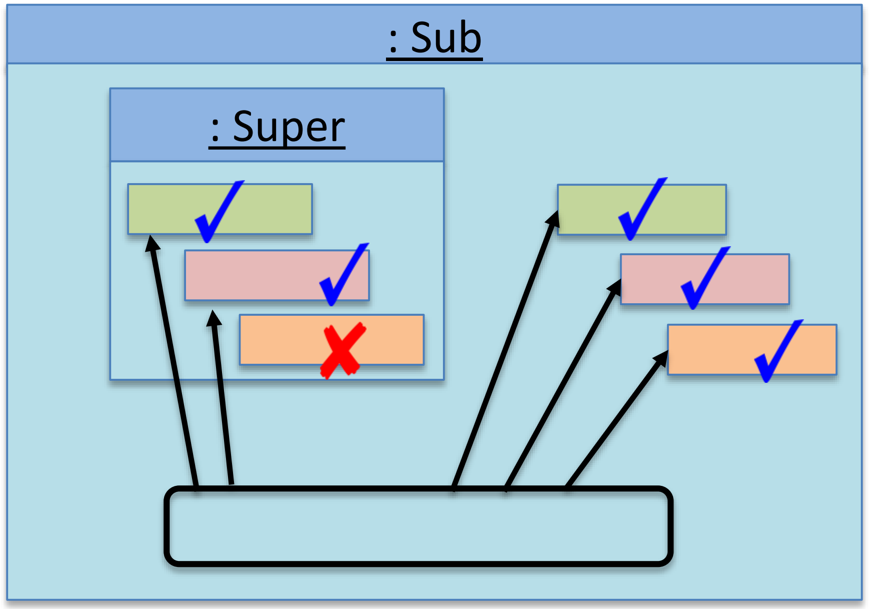
✗ Inaccessible

Only public member visible from outside

ACCESSIBILITY MODIFIER : EFFECT

Accessibility of Super-Class from Sub-Class

From a method of Sub-class, Everything is visible and can be access except the private members of its superclass



Discuss more on the accessibility of each modifier between Superclass and Subclass

ACCESS MODIFIERS TIMETABLE

Access Modifier	Default	Private	Protected	Public
Accessible inside the class	yes	yes	yes	yes
Accessible within the subclass inside the same package	yes	no	yes	yes
Accessible outside the package	no	no	no	yes
Accessible within the subclass outside the package	no	no	yes	yes

ACCESSOR AND MUTATOR

Instance Variable + `private` → A class provides services **to access and modify** data values

Accessor
- Returns the value of a variable
- Syntax: getX
- getters

Mutator
- Changes the value of variable
- Syntax : setX
- setters

ACCESSOR : getInstanceVariable ( )

Syntax

```
public returnType getInstanceVariable ( )
{
      return instanceVariable;
}
```

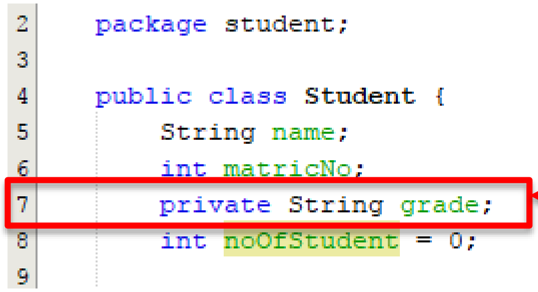
Where;

returnType = The same data type as the **instanceVariable** data type

private instance variable cannot directly access by main class.
Thus, classes provide **public** accessor methods for access purpose

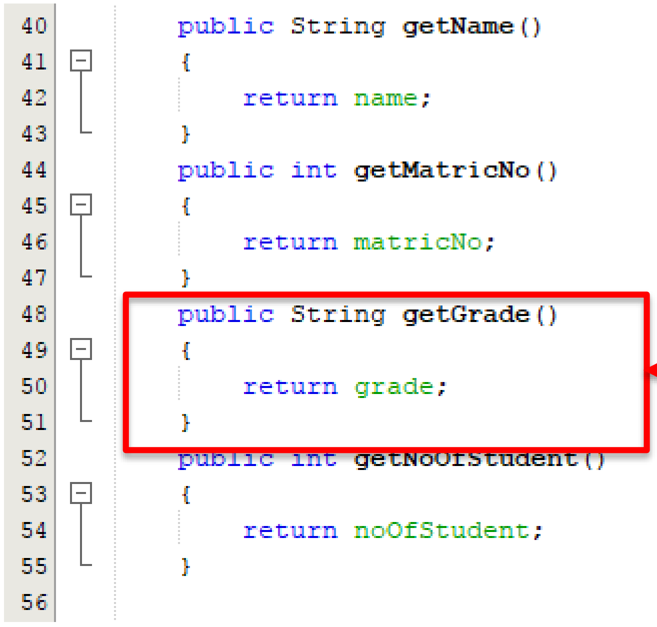
ACCESSOR : `Example`

Student Class



```java
package student;

public class Student {
    String name;
    int matricNo;
    private String grade;
    int noOfStudent = 0;

```

`grade` **:**
private instance variable

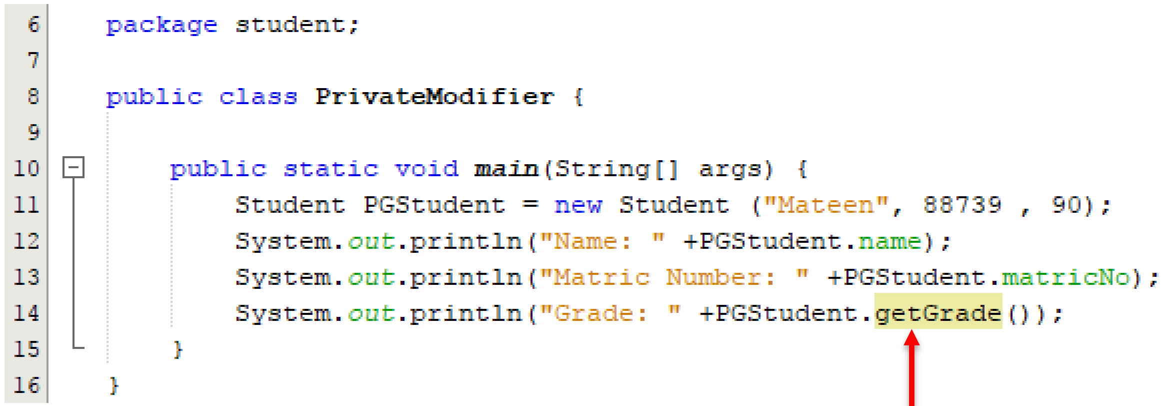


```java
public String getName()
{
    return name;
}
public int getMatricNo()
{
    return matricNo;
}
public String getGrade()
{
    return grade;
}
public int getNoOfStudent()
{
    return noOfStudent;
}

```

`Public` **accessor method to access** **`private`** **instance variable** **`grade`**

ACCESSOR : Example

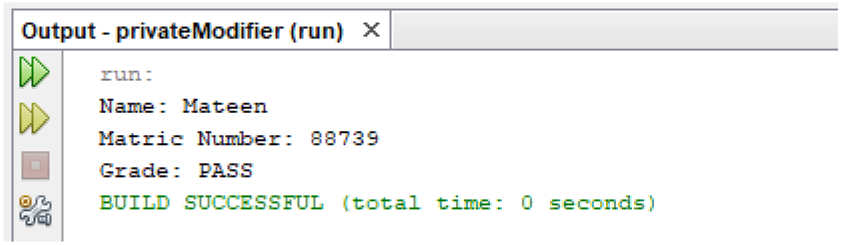
Main Class



```
package student;

public class PrivateModifier {

    public static void main(String[] args) {
        Student PGStudent = new Student ("Mateen", 88739 , 90);
        System.out.println("Name: " +PGStudent.name);
        System.out.println("Matric Number: " +PGStudent.matricNo);
        System.out.println("Grade: " +PGStudent.getGrade());
    }
}
```

PGStudent.getGrade( ) – invoke accessor method

Output



Output - privateModifier (run)

```
run:
Name: Mateen
Matric Number: 88739
Grade: PASS
BUILD SUCCESSFUL (total time: 0 seconds)
```

MUTATOR : setInstanceVariable ( )

Syntax

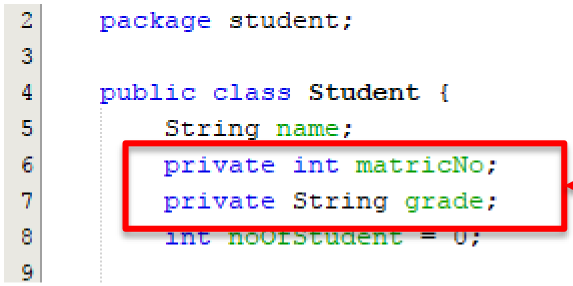
```
public returnType setInstanceVariable (datatype
               newValue )
{
          // assign newValue to instance variable
}
```

Where;

returnType = **void**

MUTATOR : Example

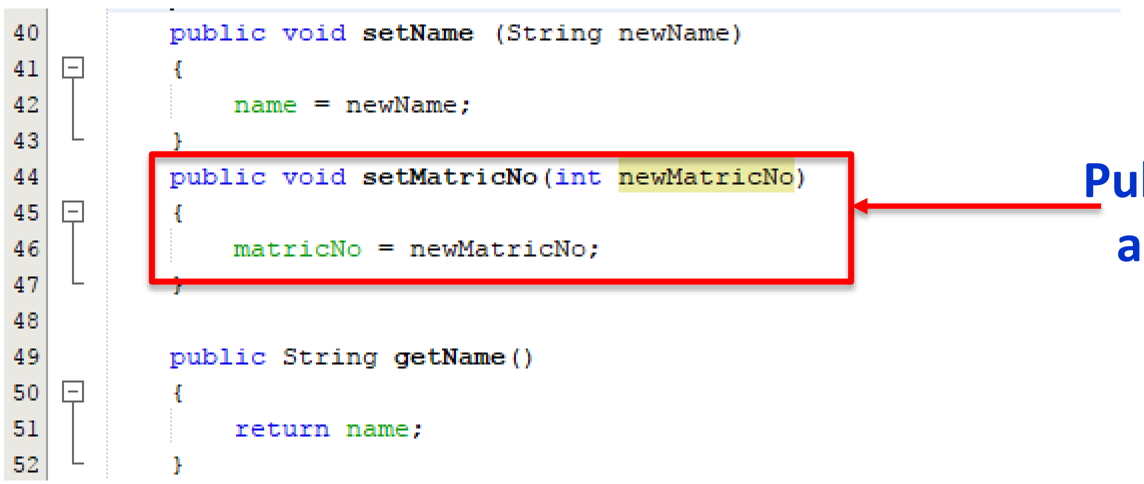
Student Class



```
package student;

public class Student {
    String name;
    private int matricNo;
    private String grade;
    int noOfStudent = 0;

```

matricNo :

private instance variable



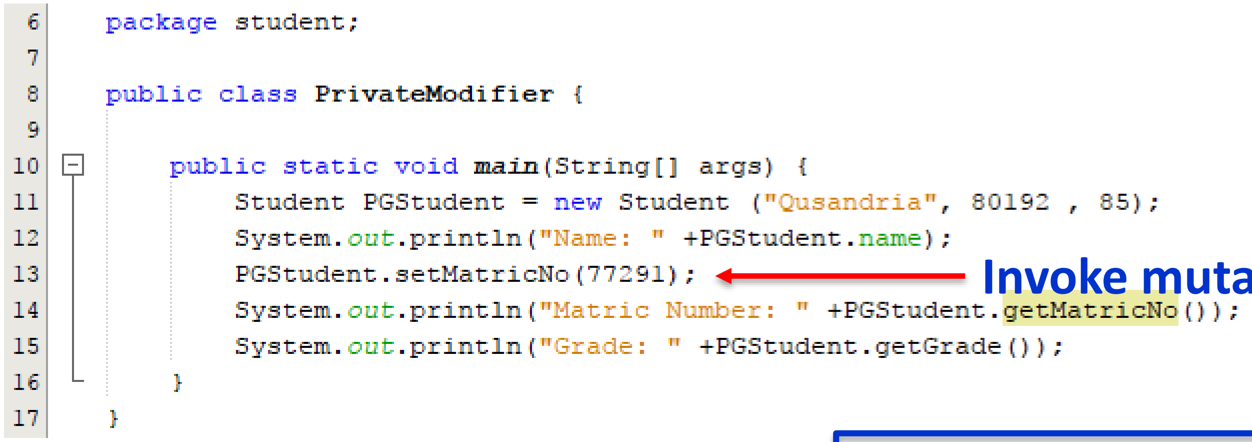
```
    public void setName (String newName)
    {
        name = newName;
    }
    public void setMatricNo(int newMatricNo)
    {
        matricNo = newMatricNo;
    }

    public String getName()
    {
        return name;
    }
```

setMatricNo :

Public mutator method to access private instance variable

MUTATOR : Example

Main Class

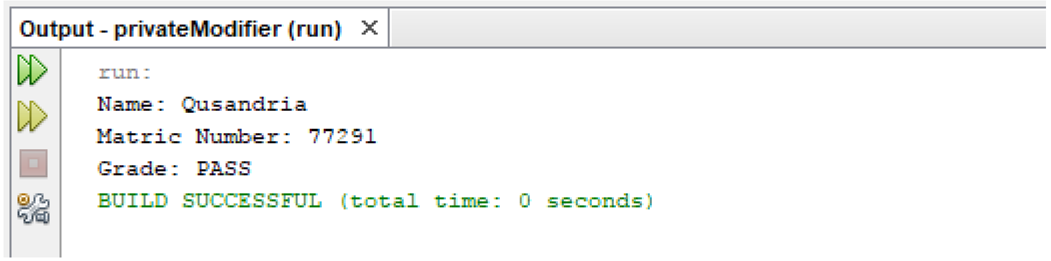


```
package student;

public class PrivateModifier {

    public static void main(String[] args) {
        Student PGStudent = new Student ("Qusandria", 80192 , 85);
        System.out.println("Name: " +PGStudent.name);
        PGStudent.setMatricNo(77291);
        System.out.println("Matric Number: " +PGStudent.getMatricNo());
        System.out.println("Grade: " +PGStudent.getGrade());
    }
}
```

Invoke mutator method

CAN YOU SPOT THE DIFFERENCE??

Output



Output - privateModifier (run)

```
run:
Name: Qusandria
Matric Number: 77291
Grade: PASS
BUILD SUCCESSFUL (total time: 0 seconds)
```

Universiti Malaysia PAHANG
Engineering • Technology • Creativity

Author Information

Dr. Nor Saradatul Akmar Binti Zulkifli

Senior Lecturer
Faculty of Computer Systems & Software Engineering
Universiti Malaysia Pahang