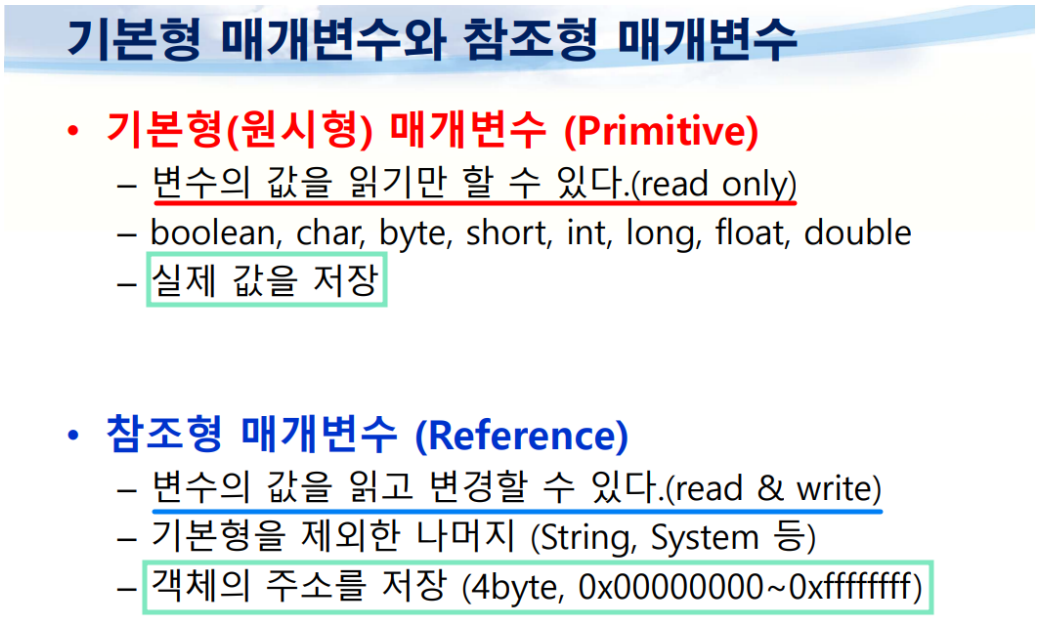
12.29 木 _ 매개 변수



반환 값이 없어 에러!!

```
package test;

class Bar{
  static int doSomething() {     => 정상 작동 시키려면 int 를 void 로 변경해야 한다
      //  반환 값이 있어야 하는데 없다..;;;;
  }
}

public class test4 {

  public static void main(String[] args) {


  }

}
```

```
package test;
```

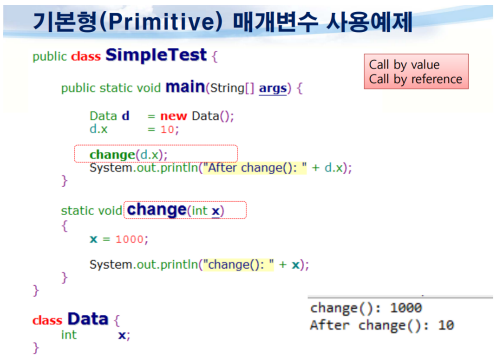
```
class Bar{
  static void doSomething(int argA) {
    System.out.println(argA);
  }
}

public class test4 {

  public static void main(String[] args) {
    Bar.doSomething(3);

  }

}
```

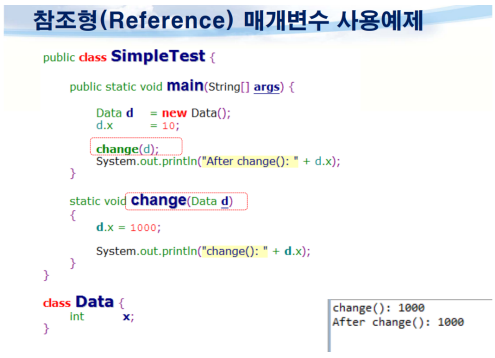
스레드 - 일꾼

main 문부터 하나하나 읽어가면서 코드 실행시킨다

⭐ 인자 값은 매개 변수에 **복사**가 되어서 들어온다 (call by value)

매서드를 호출할 때 넣어주는 input값 자체를 인자 값이라고 한






```
package test;

class Bar{
  static void doSomething(int argA) {
    argA = 5;
    System.out.println(argA);
  }
```

```
}

public class test4 {

  public static void main(String[] args) {

    int x =3;

    Bar.doSomething(x);  // 메소드가 종료되면 호출한 이곳으로 다시 돌아온다

    System.out.println(x);

  }

}
```

Call by reference

```
package test;

class Bar{
  static void doSomething(Foo argFooObj) {
    argFooObj.x = 10;

  }
}

class Foo{

  int x =3;
}


public class test4 {

  public static void main(String[] args) {

    Foo f1 = new Foo();

    Bar.doSomething(f1);  // 메소드가 종료되면 호출한 이곳으로 다시 돌아온다

    System.out.println(f1.x);

  }

}
```

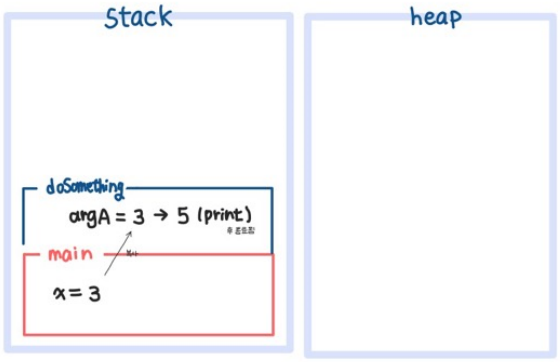
Call by value - 프리미티브

콜바이 레퍼런스 → 레퍼런스 값이 넘아간다

```
package test;

class Bar{
  static void doSomething(Foo argFooObj , int argY) {
    argY =20 ;
    argFooObj.x = 10;

  }
}

class Foo{

  int x =3;
}


public class test4 {

  public static void main(String[] args) {

    Foo f1 = new Foo();

    int y =10 ;

    Bar.doSomething(f1,y);  // 메소드가 종료되면 호출한 이곳으로 다시 돌아온다

    System.out.println(f1.x + ":"+y);

  }

}
```

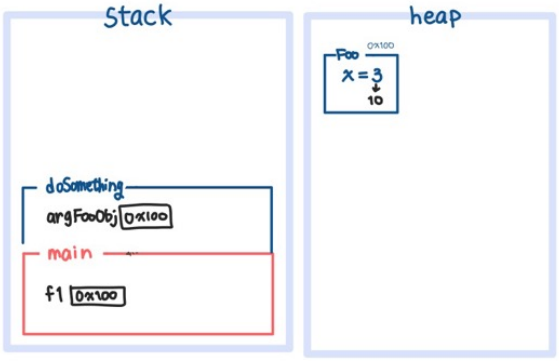
```
class Bar {
    static void doSomething (int argA) {
        argA = 5;
        sysout (argA)
    }
}
public class MyProject {
    public static void main (String args[]) {
        int x = 3;
        Bar. doSomething (x);
        sysout (x);
    }
}
```

Call by Value




```
class Bar {
    static void doSomething (Foo argFooObj) {
        argFooObj.x = 10;
    }
}
class Foo {
    int x = 3;
}
public class MyProject {
    public static void main (String args[]) {
        Foo f1 = new Foo();
        Bar. doSomething (f1);
        sysout (f1.x); // 10이 출력됨!
    }
}
```

Call by reference



Deep copy vs Shallow copy

```
package Test;


// Method


class Bar {
  int x = 3;
}

// Deep copy vs Shallow copy
public class MyProject {
  public static void main(String args[]) {

    Bar b1 = new Bar();
    b1.x = 10;
    Bar b2 = b1; // Shallow copy

    // <<----
    Bar b3 = new Bar();
    b3.x = b1.x;
    // ---->> Deep copy
  }
}
```

```
package Test;

// Method

class Bar {
  int x = 3;
}

// Deep copy vs Shallow copy
public class MyProject {
  public static void main(String args[]) {

    Bar b1 = new Bar();
    b1.f1 = new Foo();
    b1.k = new int[5];

    // b1을 Deep copy 하라~!
    Bar b2 = new Bar();
    b2.f1 = b1.f1;
    b2.k = b1.k;
  }
}
```