THE UNIVERSITY OF
MELBOURNE

Lab5（week6）

Project B Consultation

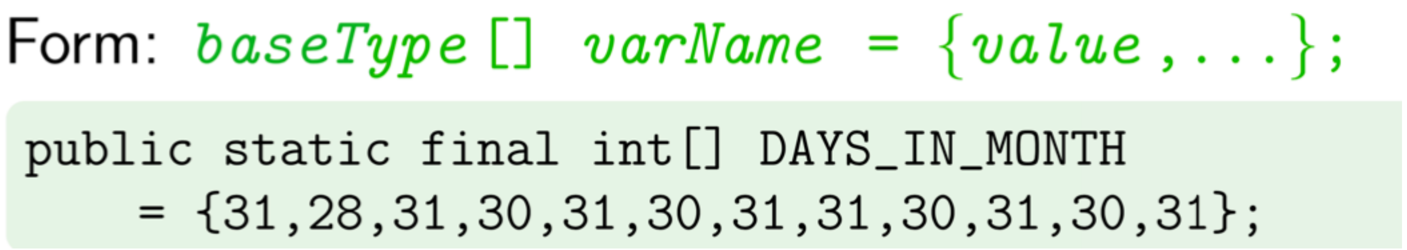
COMP90041 Programming

and software development

Zhe(Zoe) Wang

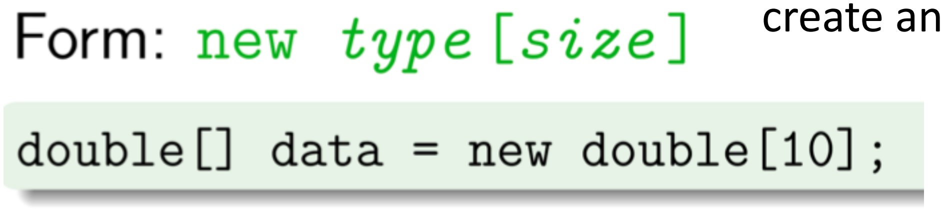
Array

give value to array



Form: *baseType* [] *varName* = {*value*,...};

```
public static final int[] DAYS_IN_MONTH
    = {31,28,31,30,31,30,31,31,30,31,30,31};
```



Form: new *type* [*size*]

```
double[] data = new double[10];
```

create an empty array

type, size

```
int[] bob = {1,2,3}
```

✓

```
int[] sue;
sue = {1,2,3};
```

✗

```
int[] sue;
sue = new int[3];
sue[0] = 1;
sue[1] = 2;
sue[2] = 3;
```

```
int[] sue;
sue = new int[]{1,2,3};
```

iterating arrays

```
for (int i = 0 ; i < a.length ; ++i) ... a[i] ...
```

The Foreach Loop

```
for(elementType name :  array) body
```

print array, assign values, copy…

demo0

Array of objects

demo1

- Can declare array with a class as base type
- *E.g.*, `String[] args`

demo: create a array of Dog

demo: add a new Dog

demo: remove a Dog

demo: edit a Dog

String operations

s.substring()

- `s.substring(i, j)` returns the substring of `s` from character `i` through `j`-1, counting the first char as 0
- E.g`., "smiles".substring(1,5)`, returns `"mile"`

s.indexOf()

- `s.indexOf(s2)` returns the first position of `s2` in `s`

s.equals()

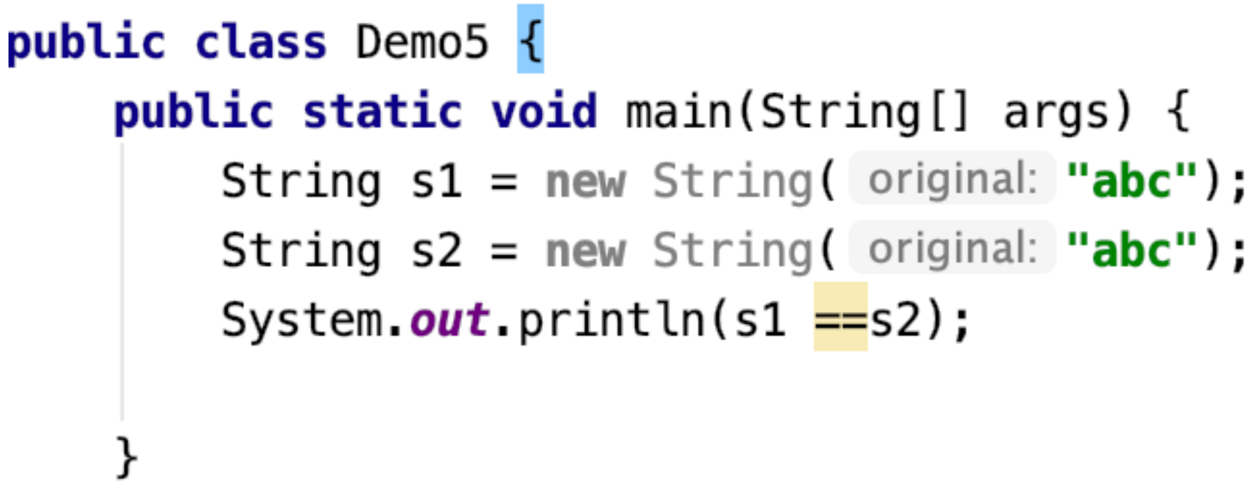
- `s.equals(s2)` returns true if a `s` and `s2` are identical

== VS equals

= = can correctly test two values of a primitive type

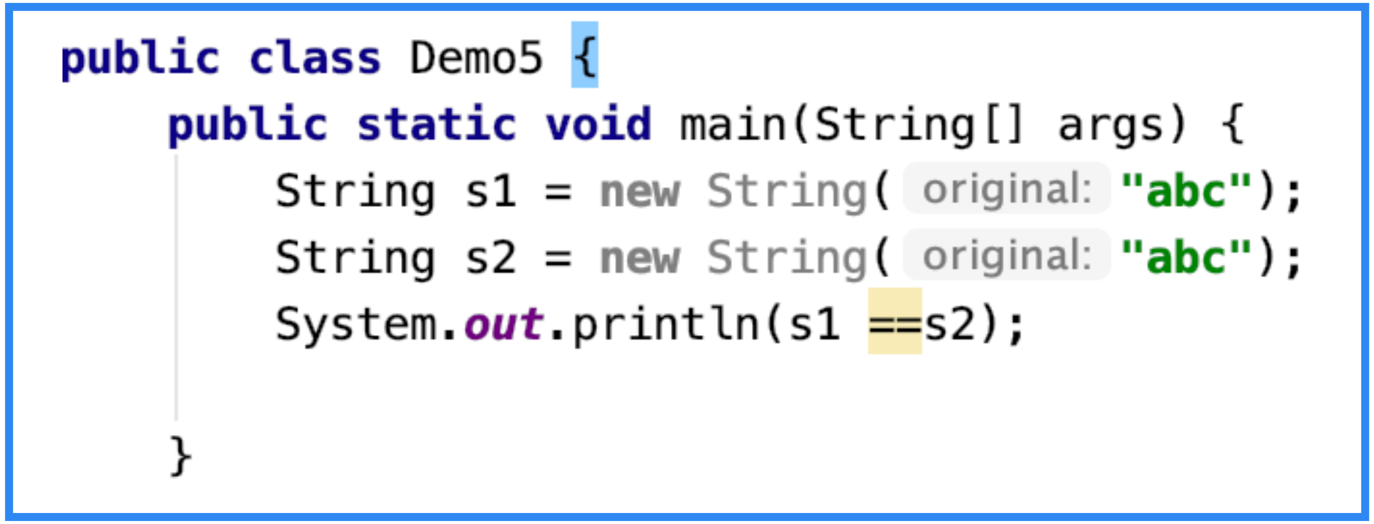
However, when applied to two objects such as objects of the String class, == tests to see if they are stored in the same memory location, not whether or not they have the same value

Do not use = = with Strings!!

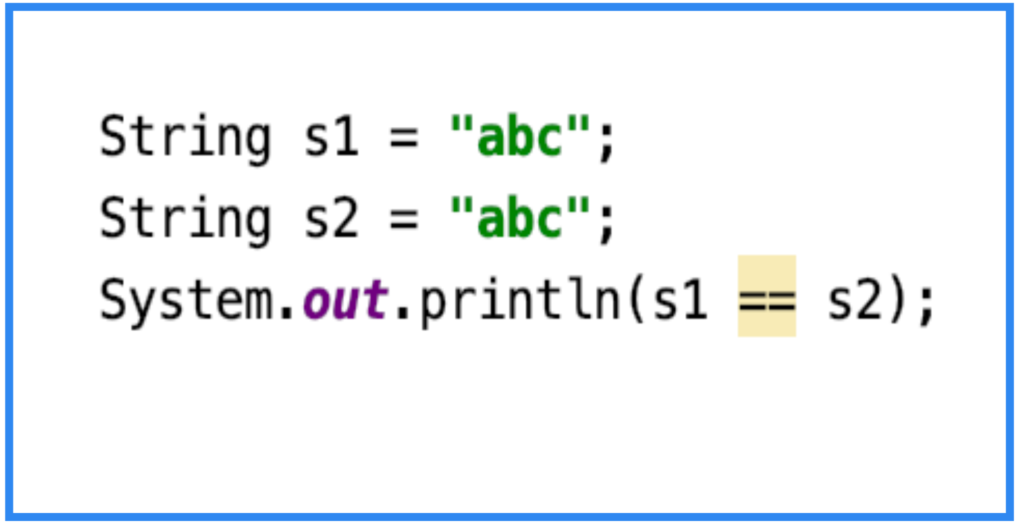


```
public class Demo5 {
    public static void main(String[] args) {
        String s1 = new String( original: "abc");
        String s2 = new String( original: "abc");
        System.out.println(s1 ==s2);

    }
```

== VS equals



```
public class Demo5 {
    public static void main(String[] args) {
        String s1 = new String( original: "abc");
        String s2 = new String( original: "abc");
        System.out.println(s1 ==s2);

    }
```



```
String s1 = "abc";
String s2 = "abc";
System.out.println(s1 == s2);
```

demo5

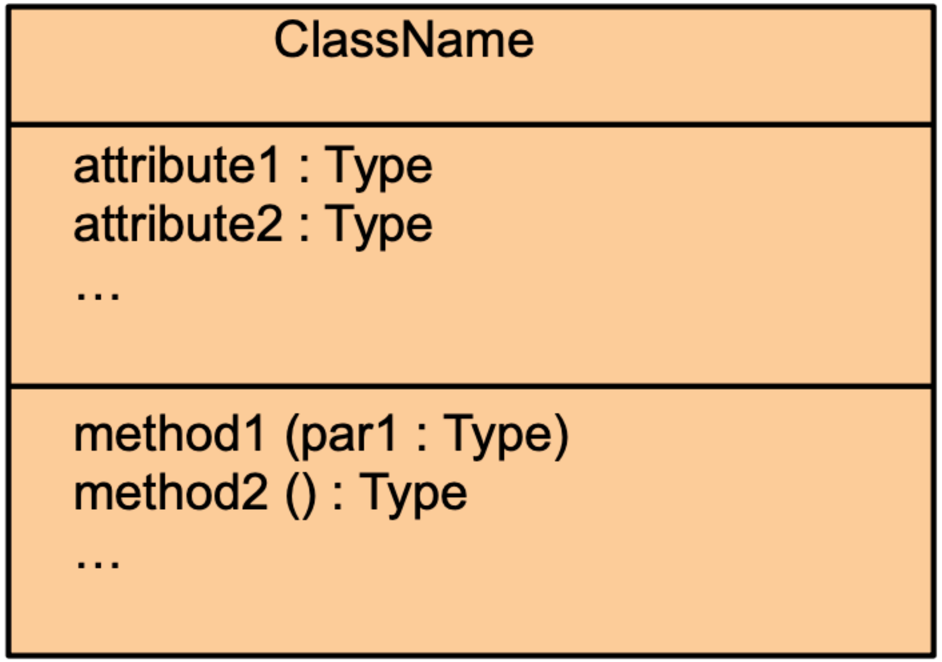
String operations

s.split()

split (String regex, int limit)

demo2

UML



ClassName

attribute1 : Type
attribute2 : Type
…

method1 (par1 : Type)
method2 () : Type
…

- Each name is preceded by a character that specifies its access type:
  - A minus sign (-) indicates private access
  - A plus sign (+) indicates public access
  - A sharp (#) indicates protected access
  - A tilde (~) indicates package access

printf – formatted output

- Form:

```
System.out.printf(format-string, args...);
```

- *E.g.*:

```
System.out.printf("Average:  %5.2f", average);
```

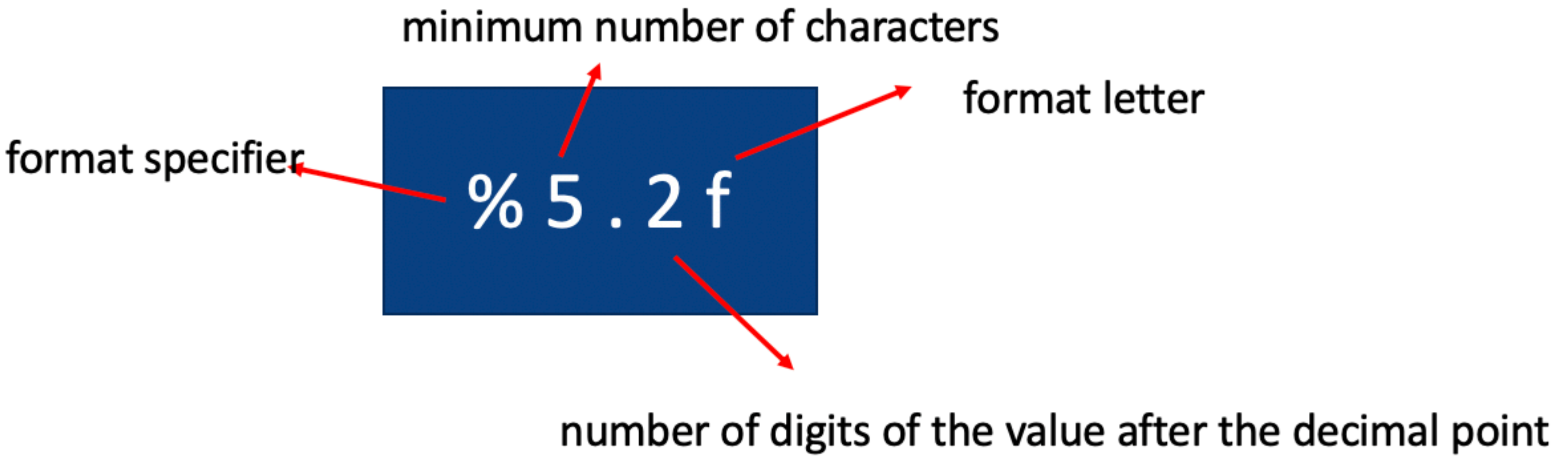
  - If the number if negative, the value will be left-justified, otherwise right-justified

demo2

x =5.7889

print as

5.78



minimum number of characters

format letter

format specifier

%5.2f

number of digits of the value after the decimal point

THE UNIVERSITY OF
MELBOURNE

Thank you