

Monday, Feb. 5

Lecture 5

```
class Person {  
    int age;  
    String nationality;  
    double weight;  
    double height;
```

attribute

```
Person(int a, String n, double w, double h) {  
    this.age = a;  
    this.nationality = n;  
    this.weight = w;  
    this.height = h;  
}
```

constructor

```
double getBMI() {  
    double bmi = this.weight / (this.height * this.height);  
    return bmi;  
}
```

accessor  
(return required)

```
void gainWeightBy(double units) {  
    this.weight = this.weight + units;  
}
```

mutator

return type

change attribute values

boolean park( char c ) {  
    ACCESSO

boolean result = false;  
    if ( c == 'P' || c == 'K') {  
            result = true;  
    }  
    return result;

X  
not complete  
, missing  
return statement

boolean pork (char c) {  
 e.g. 'a'  
 if (c == 'P' || c == 'K') {  
 return true;  
 }  
}  
X  
not  
sample

// missing return statement  
when the if-condition is  
false.

badpan

if (

X

){

not

compar

able

unreachable

statement

return

result

;

badpan

return

result

;

park (char 'P') {

'P' == 'a' || 'C' == 'k' ) {

return tmp;

False;

result =

'C' == 'P' || 'C' == 'K')

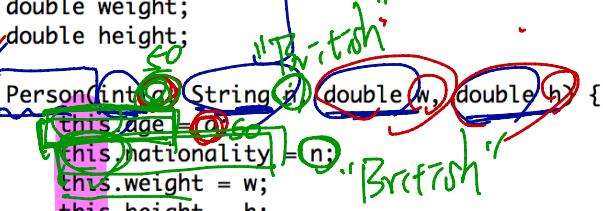
```

class Person {
    int age;
    String nationality;
    double weight;
    double height;

    Person(int a, String n, double w, double h) {
        this.age = a;
        this.nationality = n;
        this.weight = w;
        this.height = h;
    }
}

```

*this.age = 50*



```

    double getBMI() {
        double bmi = this.weight / (this.height * this.height);
        return bmi;
    }
}

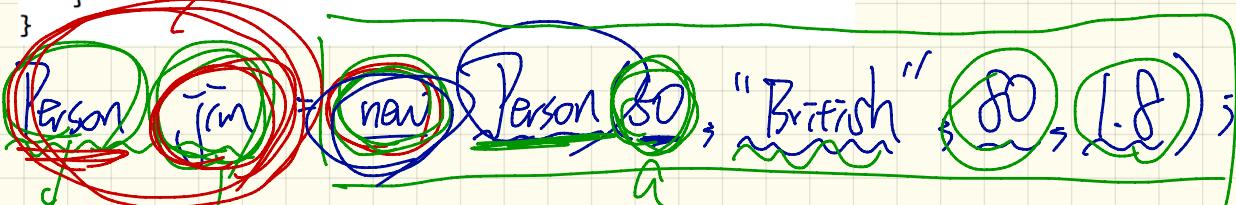
```

*type of Jim is Person*

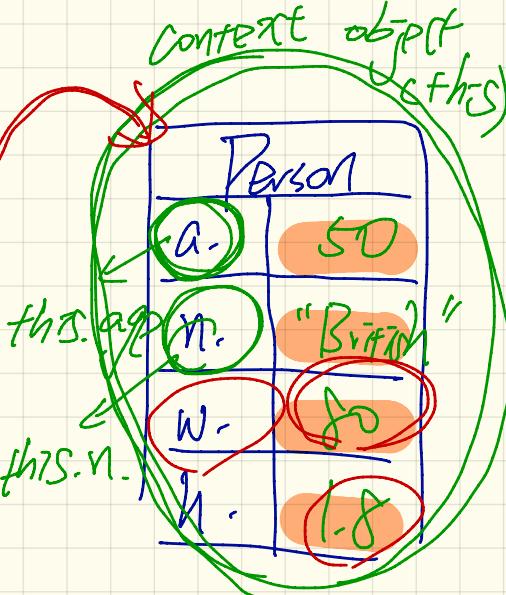
```

    void gainWeightBy(double units) {
        this.weight = this.weight + units;
    }
}

```



*variable*  
→ store address of Jim Person object



```

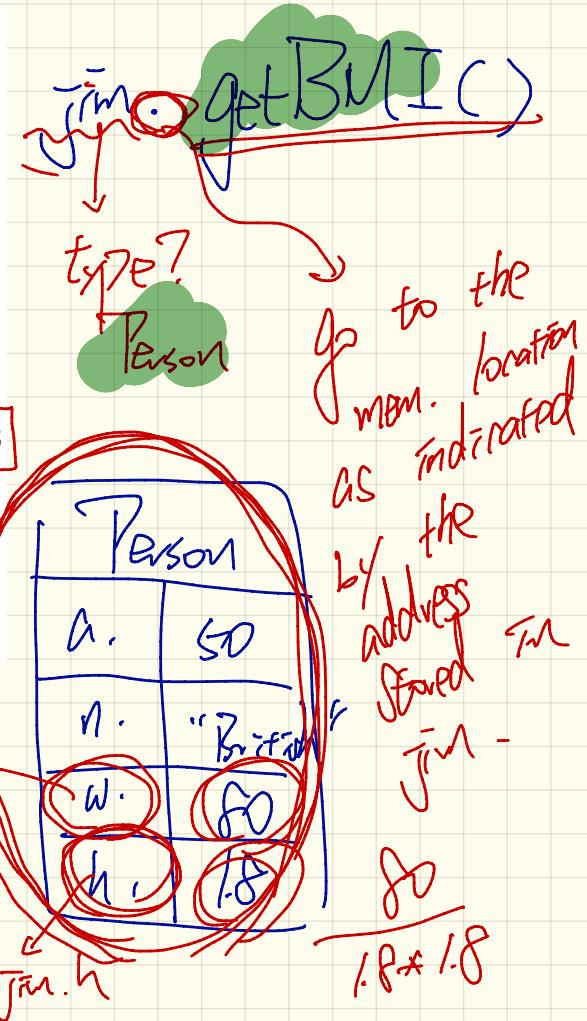
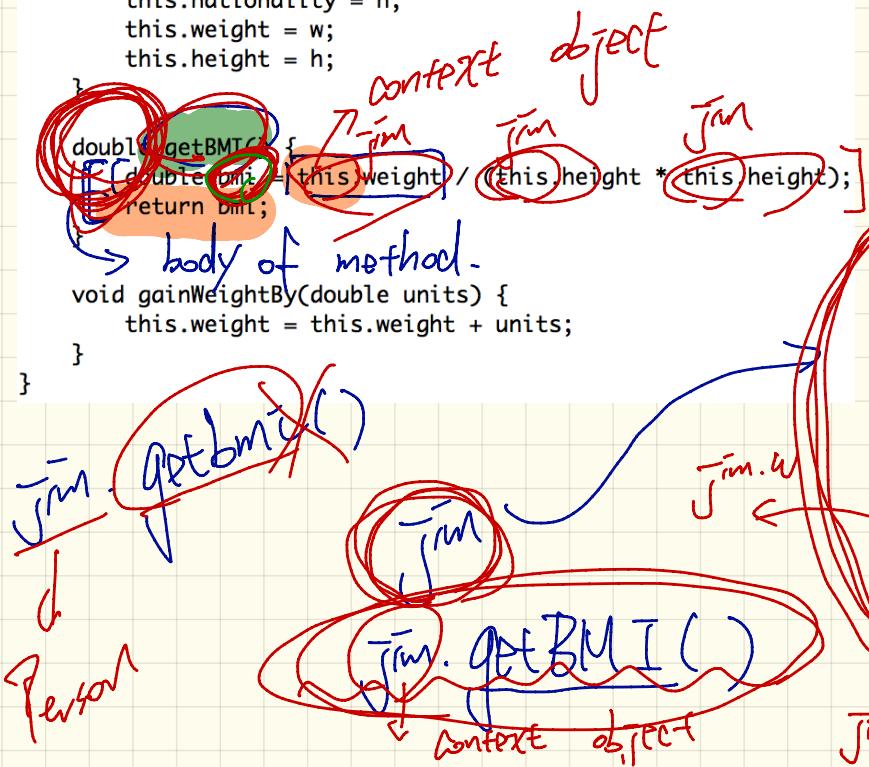
class Person {
    int age;
    String nationality;
    double weight;
    double height;

    Person(int a, String n, double w, double h) {
        this.age = a;
        this.nationality = n;
        this.weight = w;
        this.height = h;
    }

    double getBMI() {
        double bmi = this.weight / (this.height * this.height);
        return bmi;
    }

    void gainWeightBy(double units) {
        this.weight = this.weight + units;
    }
}

```

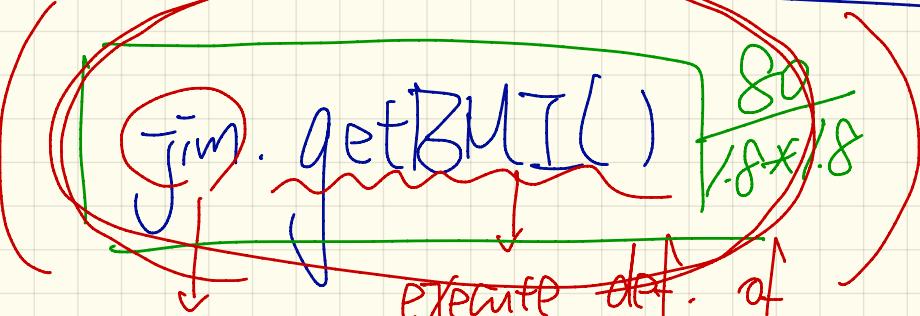


Store/Catch return value

from an accessor method call

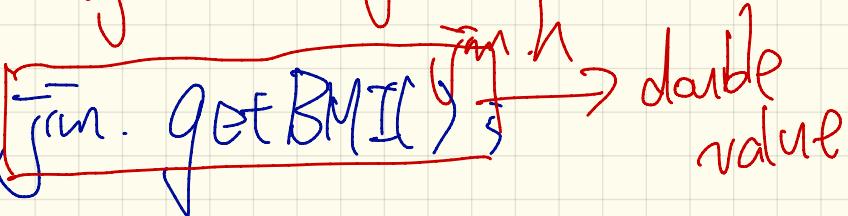
① `Println ( Jim.getBMI() )`

execute def. of



context object `getBMI` according to  
`Jim.w` and ()

② double result = Jim.getBMI() → double value



```
class Person {  
    int age;  
    String nationality;  
    double weight;  
    double height;
```

```
Person(int a, String n, double w, double h) {  
    this.age = a;  
    this.nationality = n;  
    this.weight = w;  
    this.height = h;  
}
```

```
double getBMT() {  
    double bmi = this.weight / (this.height * this.height);  
    return bmi;  
}
```

```
void gainWeightBy(double units) {  
    this.weight = this.weight + units;  
}
```

body of implementation.

→ go to wherever  
↑ Jim points  
Jim gainWeightBy(10)

mutator method  
(no return value  
to be used).  
so  
object

Jim. weight + 10

Person	
a.	30
n.	"British"
w.	80
h.	1.8

Jim. weight

accessor method call

~~println ( jm. getBMI() );~~

double result = jm. getBMI();

mutator method call

~~X println ( jm. gainWeightBy(10) );~~ void

does  
not  
compile

~~X double~~

result = jm. gainWeightBy(10);

```

class Person {
    int age;
    String nationality;
    double weight;
    double height;

    Person(int a, String n, double w, double h) {
        this.age = a;
        this.nationality = n;
        this.weight = w;
        this.height = h;
    }
}

```

```

double getBMI() {
    double bmi = this.weight / (this.height * this.height);
    return bmi;
}

```

```

void gainWeightBy(double units) {
    this.weight = this.weight + units;
}
}

```

↓  
Jrn

Person	
a.	50
n.	"Brī"
w.	80
h.	1.8

90

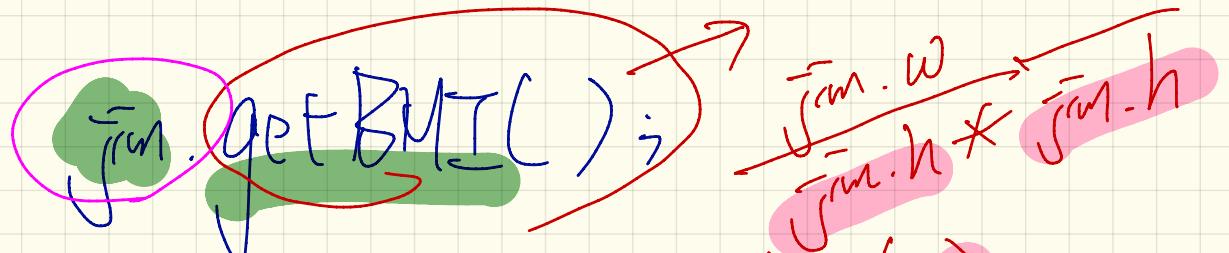
① Person jrn = new Person(50, "Brī", 1.8);

② println(jrn.getBMI());

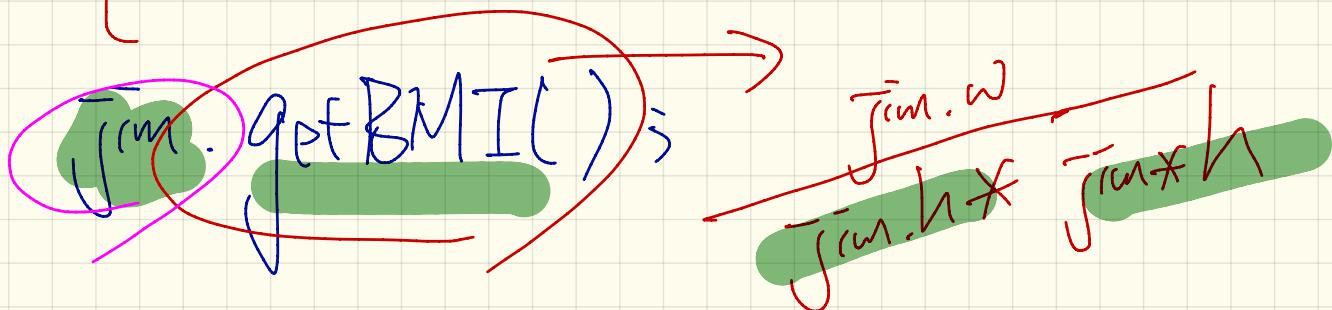
③ jrn.gainWeightBy(10);

④ println(jrn.getBMI());

90 / 1.8<sup>2</sup>



$\bar{jm} \cdot \text{gainWeightBy}(f_0)$  is increased by  $f_0$ .



class PersonTester {  
 main (...) {  
 this.gainWeightBy(10);  
 }  
}

PersonTester

}  
} Person jim = new Person(...);  
jim.m();  
Context object

class Person {  
 void gainWeightBy(...) {  
 }  
 }  
}  
} m() {  
 this.gainWeightBy(10);  
}

# For Loop

```
for( ; ; )
```

body  
of  
loop

}

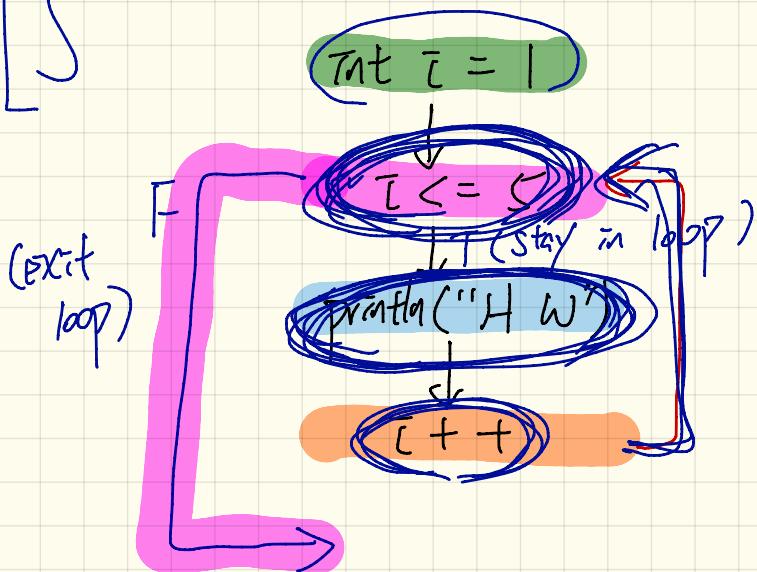
Initialization  
(once)

- boolean expression
- Stay condition (SC)
- as long as SC is true,  
loop repeating

} {

→ update  
(at the end  
of each  
repetition)

for (int i = 1; i <= 5; i++) {  
     cout << "Hello World";  
 }  
 } // loop counter



i	i <= 5	Action
1	T	Print C++
2	T	Print C++
3	T	Print C++
4	T	Print C++
5	T	Print C++
6	F	

~~infinite loops~~  
~~never terminating~~

① [for (int i = 1 ; i <= 5 ; i++) {  
    printf("H W");  
}

---

\_\_\_\_\_

② [int i = 1 ;  
for ( ; i <= 5 ; i++) {  
    printf("H W");  
}

③

```
int i = 1;
for ( ; i <= 5; ) {
    printf("H W");
}

```

The code is a C program. It starts with an integer variable *i* initialized to 1. A *for* loop runs as long as *i* is less than or equal to 5. Inside the loop, the *printf* function is called with the string "H W". After the loop, there is a closing brace for the *for* loop and another closing brace for the entire code block.

A red rectangular box highlights the expression *i++* inside the *for* loop condition.

$$\textcircled{1} = \textcircled{2} = \textcircled{3}$$

④

int  $i = 1;$

for(  $i \leq 5;$  ) {

$i++;$

    printf( "H %c", i );

④

$\neq$  ③

```
for (int i = 0; i < 10; i++)  
    {  
        cout << "Welcome";  
    }  
}
```

Q1. How many times cout will be  
executed?

(Q2) How many times cout will be  
checked?