2017-2018 Güz Yarıyılı
Balıkesir Üniversitesi
Bilgisayar Mühendisliği Bölümü

BMM3101 PROGRAMLAMA DİLLERİ

If, Function Calling, While, For, Array, 2D Array

Yrd. Doç. Dr. İbrahim KÜÇÜKKOÇ
Web: http://ikucukkoc.baun.edu.tr
Email: ikucukkoc@balikesir.edu.tr

How would you decide if a shirt sale is a good deal or not? Describe the criteria you would look for in a sale and your decision process.

For example: if shirts are discounted 20% or more then that's a good deal, or if I already have a lot of shirts then no deal will interest me, etc.

If the shirts are expensive, then it's only a good deal if it's a big sale with a discount of 40% or more. If the shirts are reasonably priced and I like them, then a good deal is at least 10% off.

if Statement

Test condition

Value varies

Between true and false

Booleans

Always true or false

Great for decision-making!

```
boolean isCold = true; //true or false

if(isCold) {
    //executes ONLY if isCold is true
    System.out.println("It's cold, wear a coat!");

}
```

Print output

```
It's cold, wear a coat!
```

if Statement

Test condition

Value varies

Between true and false

Booleans

Always true or false

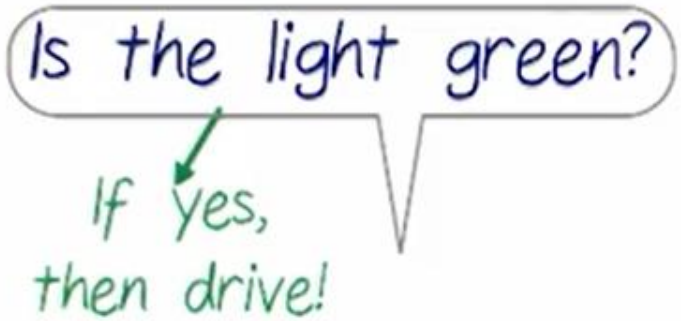
Great for decision-making!

```
boolean isCold = false; //true or false

if(isCold) {
    //executes ONLY if isCold is true
    System.out.println("It's cold, wear a coat!");

}
```

Print output

Self-Driving Car

```
boolean isLightGreen = true; //true or false

if(isLightGreen) {
    // traffic light is green
    System.out.println("Drive!");
}
```



Print output

```
Drive!
```

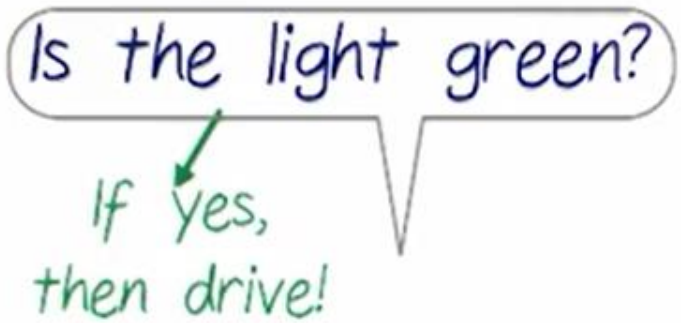
Self-Driving Car

```
boolean isLightGreen = ? ; //true or false
```

false

```
if(isLightGreen) {
    // traffic light is green
    System.out.println("Drive!");
}
```

skip down here



Variable Scope

```
boolean isLightGreen =    ?    ; //true or false


if(isLightGreen) {
    //traffic light is green
    double carSpeed = 100;  //in km/hr
    System.out.println("Drive!");
    System.out.println("Speed is: " + carSpeed);
}
carSpeed = carSpeed - 10;
```

outside

Cannot resolve symbol 'carSpeed'

Error!

carSpeed scope

Block of code where a variable can be used

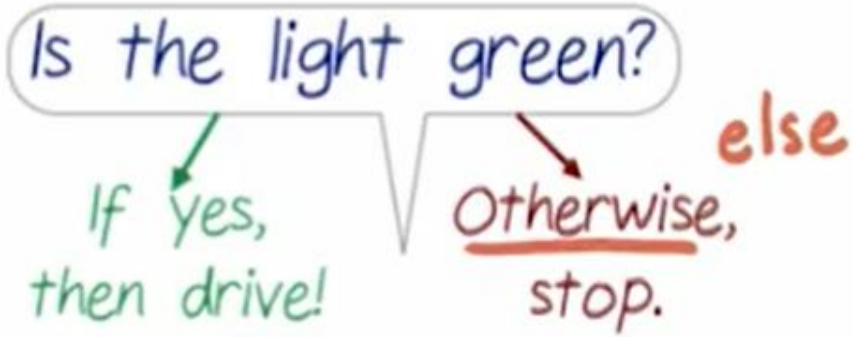
A set of curly braces defines a variable scope

{ scope }

else Statement

```
boolean isLightGreen = false; //true or false

if(isLightGreen) {
    //traffic light is green
    System.out.println("Drive!");
} else {
    //light is NOT green
    System.out.println("Stop.");
}
```

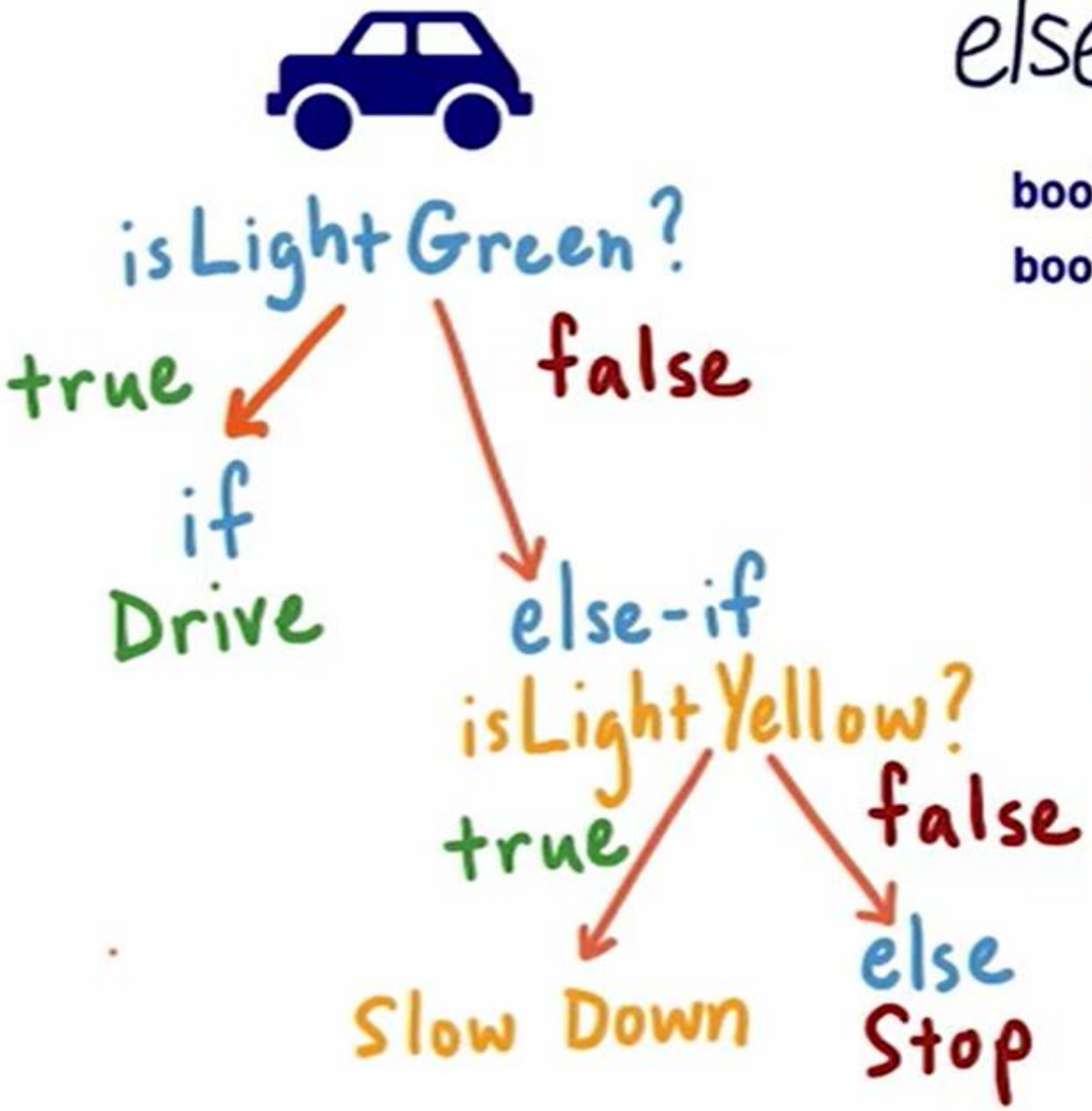
false



Print output

```
Stop.
```

else-if




```java
boolean isLightGreen = false; //true or false
boolean isLightYellow = false; //true or false

if(isLightGreen) {
    //traffic light is green
    System.out.println("Drive!");
} else if(isLightYellow) {
    //light is NOT green but is yellow
    System.out.println("Slow down.");
} else {
    //light is neither green nor yellow
    System.out.println("Stop.");
}
```

Print output

```
Stop.
```

Boolean Expressions

```
int x = 10;
```

Expression

Value

equality check

`x == 9` → false

`x != 9` → true

) used as test conditions

NOT equal

Equality checks

What boolean value do these expressions evaluate to?

Given `int x = 20;` Check true or false for each expression.

Expression		Value			
`x <= 21`	→	true	☐	false	☐
`2*2 > 8`	→	true	☐	false	☐
`5 != x`	→	true	☐	false	☐
`20 == x`	→	true	☐	false	☐

Fill in the blanks to check for two other discount test conditions: 1) age is strictly greater than 60 and 2) person is a student

```
int ticketPrice = 10;
int age = ?;
boolean isStudent = ? ;

if(age <= 15) {
    //age is less than or equal to 15
    ticketPrice = 5;
} else if (          ) {
    //age is greater than 60
    ticketPrice = 5;
} else if (          ) {
    //they are a student
    ticketPrice = 5;
}
```

Test condition 1:

Test condition 2:

Logical Operators

Three main logical operators:

1) AND `3 < 5 && 2 > 15` → false

   true && false

2) OR `3 < 5 || 2 > 15` → true

3) NOT `!(3 < 5)` → false

   NOT true

turns a value into its opposite

Logical Operators

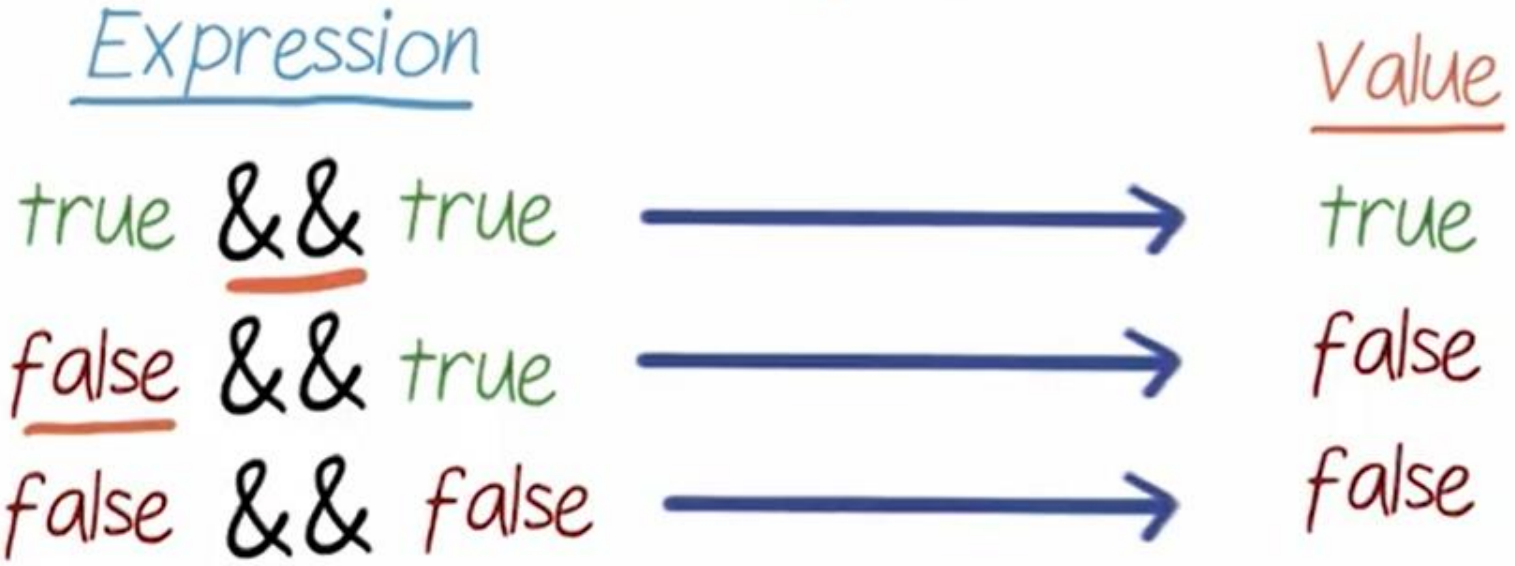
Museum discount cases

```
if( age <= 15 || age > 60 || isStudent ){
    ticket = $5
}
```

1 2 3

More Logical Operators

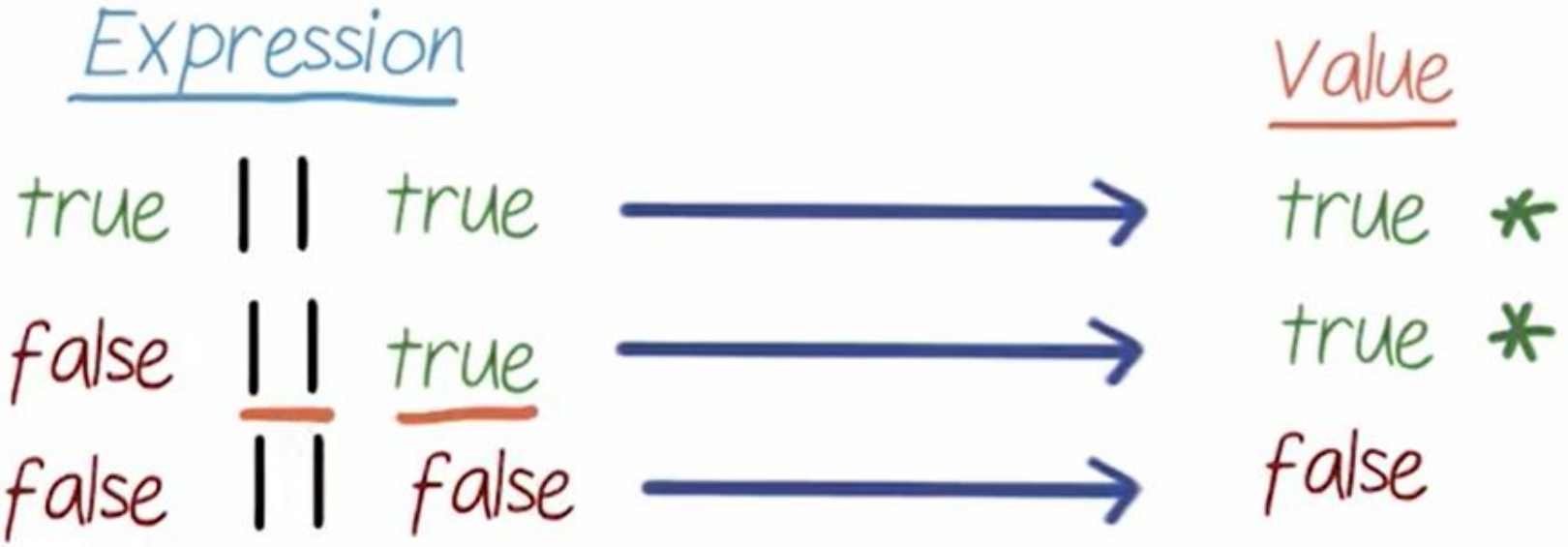
AND operator



AND expressions evaluate to true only when both conditions the operator combines are also true

More Logical Operators

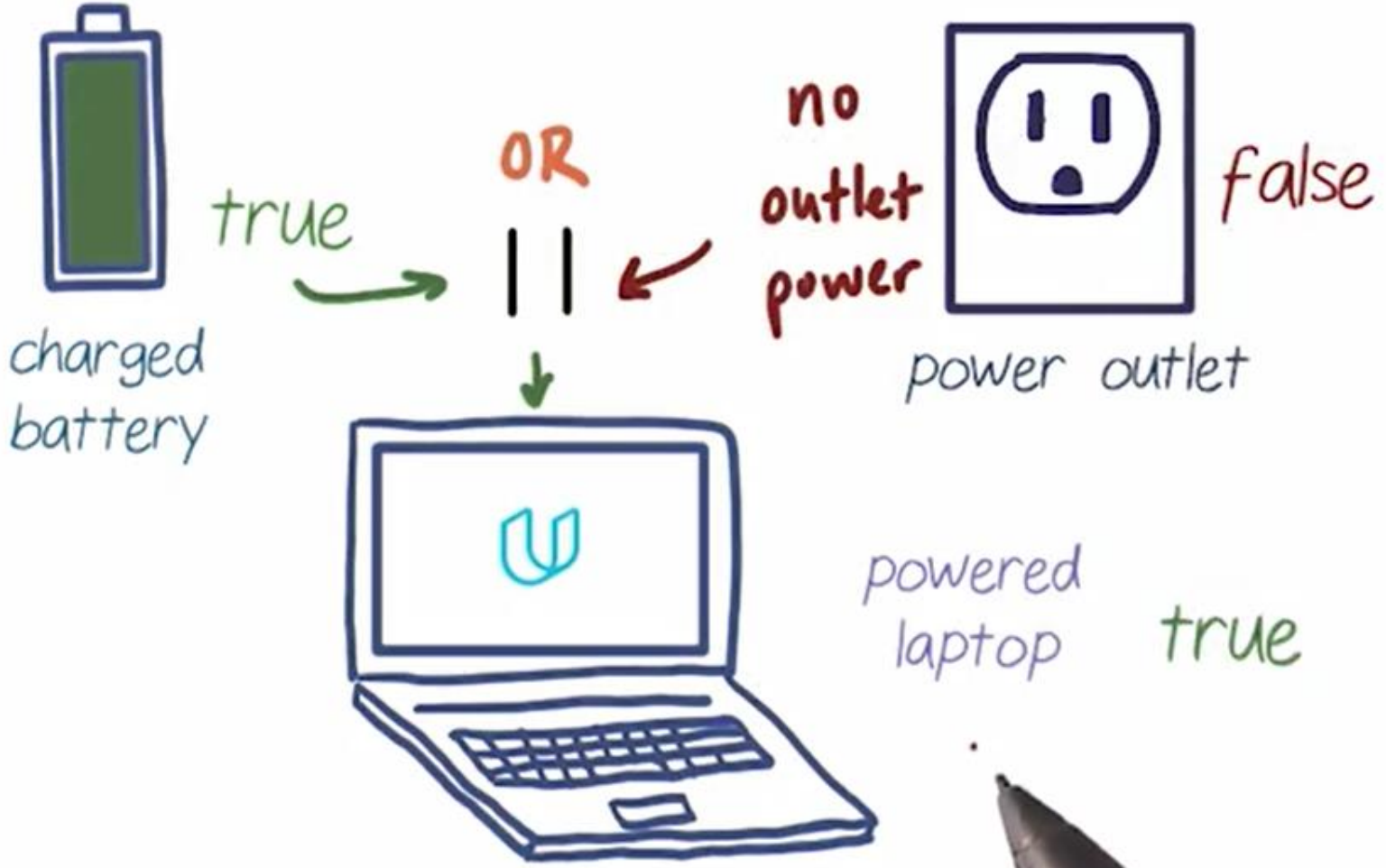
OR operator



OR expressions evaluate to true when at least one of the conditions the operator combines are also true

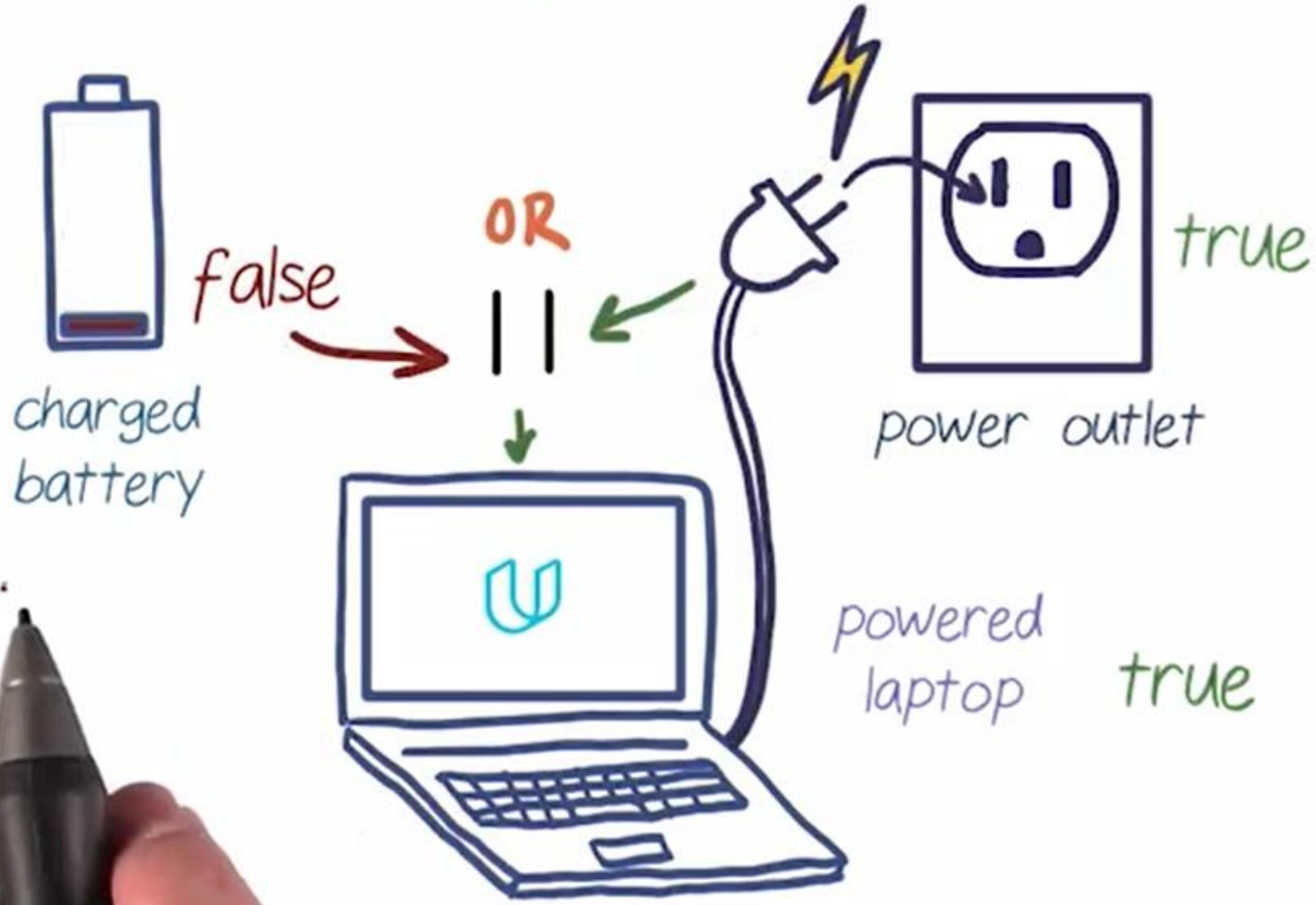
More Logical Operators

OR operator



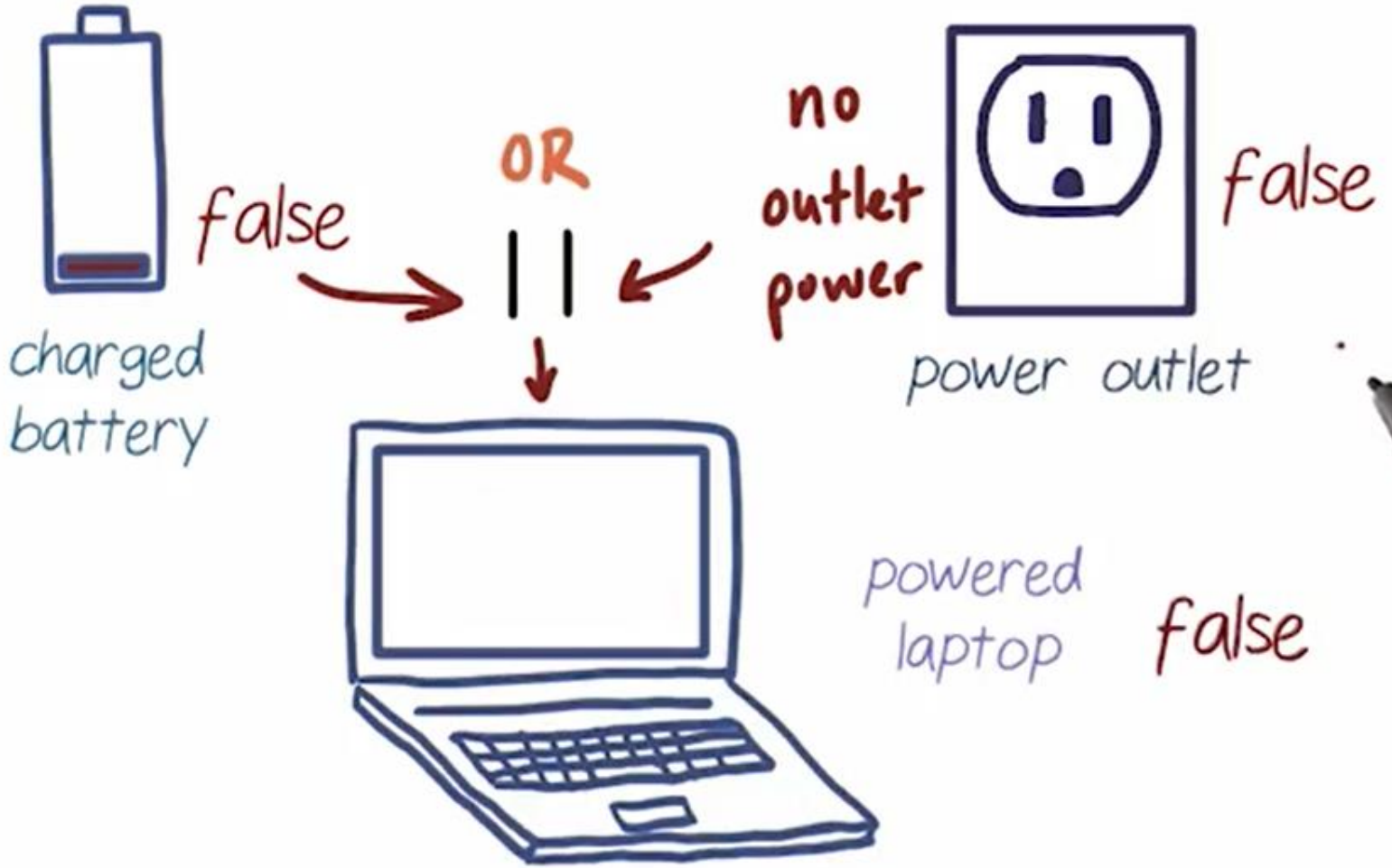
More Logical Operators

OR operator



More Logical Operators

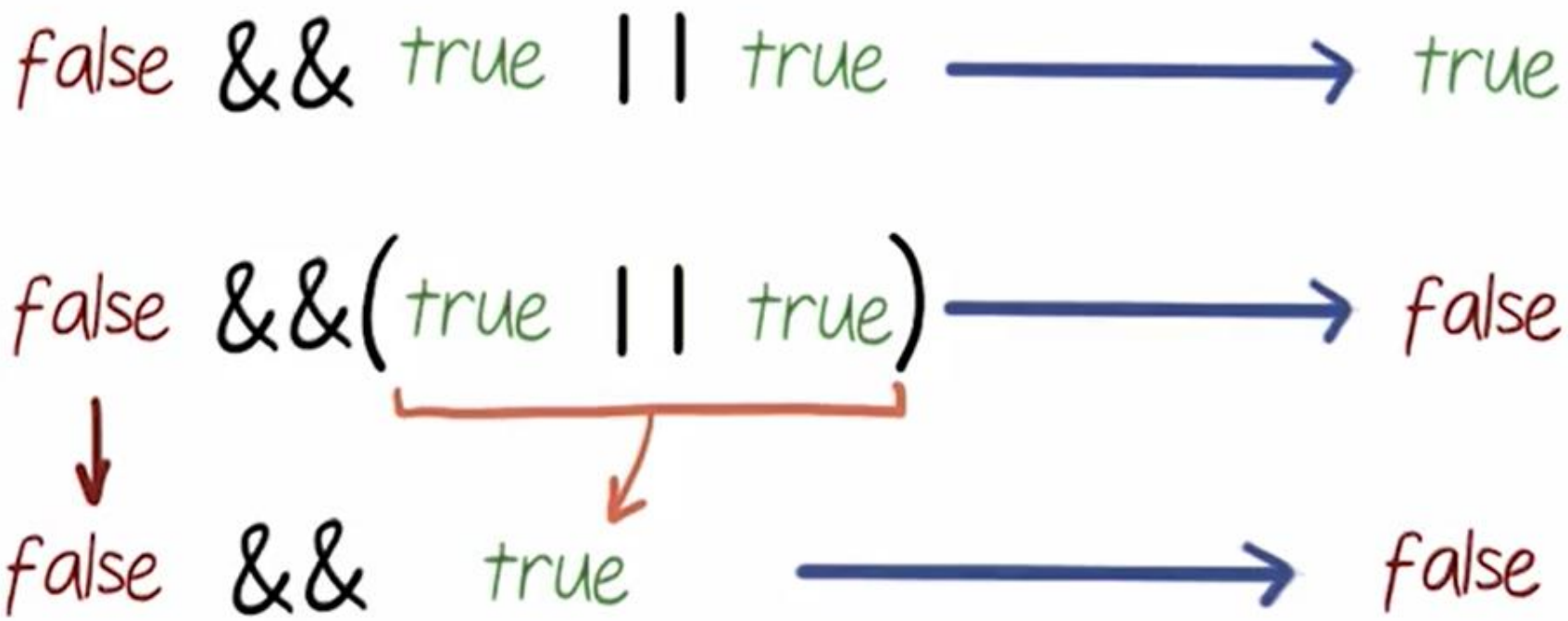
OR operator



More Logical Operators

Using multiple logical operators

&& will evaluate first, then ||



Nested if Statements

An if within an if

action

romance

comedy

horror

```
boolean action =true;
boolean romance =true;
boolean horror =false;
boolean comedy =false;
```

true

```
if(action && romance) {
    //includes action and romance
    System.out.println("This movie includes action and romance.");
    if(comedy || horror) {
        //everyone's preferences have been met!
        System.out.println("This also includes comedy or horror!");
    }
}
```

false

Print output

```
This movie includes action and romance.
```

Nested if Statements

An if within an if

action

romance

comedy

horror

```
boolean action =true;
boolean romance =true;
boolean horror =false;
boolean comedy =true;
```

true

```
if(action && romance) {
    //includes action and romance
    System.out.println("This movie includes action and romance.");
    if(comedy || horror) {
        //everyone's preferences have been met!
        System.out.println("This also includes comedy or horror!");
    }
}
```

true

Print output

```
This movie includes action and romance.
This also includes comedy or horror!
```

Consider this code for restaurant rating.

```
int rating = ?;
if (rating >= 0 && rating <= 5) {
    //rating is 0-5
    if (rating <= 2) {
        //rating is less than or equal to 2
        System.out.println("What's the reason for your low rating?");
    }
    System.out.println("Thank you for your feedback.");
}
```

What will this print for `rating = 4;`?

Switch Statement

Checks the value of a variable and tests it for equality


```
int passcode = ? ;
String coffeeType;

if (passcode == 555) {
    //espresso passcode
    coffeeType = "Espresso";
} else if (passcode == 312) {
    //vanilla passcode
    coffeeType = "Vanilla latte";
} else if (passcode == 629) {
    //drip coffee code
    coffeeType = "Drip coffee";
} else{
    //unknown passcode
    coffeeType = "Unknown";
}
System.out.println(coffeeType);
```

Switch Statement

Checks the value of a variable and tests it for equality

```
int passcode = 555;
String coffeeType;
switch(passcode) {
    case 555: coffeeType = "Espresso";
        break;
    case 312: coffeeType = "Vanilla latte";
        break;
    case 629: coffeeType = "Drip coffee";
        break;
    default : coffeeType = "Unknown";
        break;
}
System.out.println(coffeeType);
```

Print output

Espresso

```
int passcode = 555;
String coffeeType;
if (passcode == 555) {
    //espresso passcode
    coffeeType = "Espresso";
} else if (passcode == 312) {
    //vanilla passcode
    coffeeType = "Vanilla latte";
} else if (passcode == 629) {
    //drip coffee code
    coffeeType = "Drip coffee";
} else{
    //unknown passcode
    coffeeType = "Unknown";
}
System.out.println(coffeeType);
```

Complete this switch statement by writing:

1) the last case (case 12), so that it prints out **December** and
2) a default case that for any month number not between 1 and 12, will print out **Invalid month**


```
int month = ?; //a number 1-12
String monthString;

switch(month) {
    case 1: monthString = "January";
        break;
    case 2: monthString = "February";
        break;
    .
    .
    .
    case 11: monthString = "November";
        break;

}
System.out.println(monthString);
```

month	
1	January
2	February
3	March
⋮	
11	November
12	December
37	Invalid month

Complete this switch statement by writing:

1) the last case (case 12), so that it prints out **December** and
2) a default case that for any month number not between 1 and 12, will print out **Invalid month**


```
int month = ?; //a number 1-12
String monthString;

switch(month) {
    case 1: monthString = "January";
        break;
    case 2: monthString = "February";
        break;
    .
    .
    //case 12 and a default case below
    case 12: monthString = "December";
        break;
    default: monthString = "Invalid month";
        break;
}
System.out.println(monthString);
```

1) case 12 → `case 12: monthString = "December";`

2) default → `default: monthString = "Invalid month";`

month	
12	December
37	Invalid month

Function Definition

Chorus print code

```
public void chorus() {
    // print out 4 lines of chorus
    System.out.println("Once I had a love and it was a gas");
    System.out.println("Soon turned out had a heart of glass");
    System.out.println("Seemed like the real thing, only to find");
    System.out.println("Mucho mistrust, love's gone behind");
}
```

block of code

public - access modifier

void - return type

function name - how we call our function

Where do these function components belong?

Fill in the blanks with the corresponding letter.

A. <block of code>

B. <function name>

C. <return type>

```
public [ ] [ ] ( ) {
    [ ]
}
```

Function Calling

```
public void chorus() {
    // print out 4 lines
    .
    .
    .
    .
}
```

Function definition

Detailed code
How to use the function
If it returns data

```
chorus();
```

Function call

Use function in separate program
Refer to it by name

Function Calling

Print output

```
Once I had a love and it was a gas
Soon turned out had a heart of glass
Seemed like the real thing, only to find
Mucho mistrust, love's gone behind

Once I had a love and it was a gas
Soon turned out had a heart of glass
Seemed like the real thing, only to find
Mucho mistrust, love's gone behind
```

Calls

```
chorus();
chorus();
```

Define a function called `playMusic()` that prints out:

`Music is playing` when playButton = true and

`Music is paused` when playButton = false.

```
boolean playButton = false;
```

= **true** (when pressed) OR

= **false** (when released)

Define a function called `playMusic()` that prints out:

`Music is playing` when playButton = true and

`Music is paused` when playButton = false.

```
boolean playButton = ? ;

public void playMusic() {
    if (playButton) {
        System.out.println("Music is playing");
    } else {
        System.out.println("Music is paused");
    }
}
```

Parameters and Arguments

Arguments

Specific values passed into our function call
(ex. "New York")

Input parameters

String location

```
public void greeting(String location){
    // greet a specific location
    System.out.println("Hello, " + location);
}
```

Function call: `greeting("New York");`

Print output: `Hello, New York`

What will this print for the following calls?

```
public  void  weatherInterpreter(int temperature) {
    if (temperature > 30) {
        System.out.println("It's hot outside!");
    } else if (temperature < 5) {
        System.out.println("Brr, consider wearing a jacket.");
    } else {
        System.out.println("Not too hot, not too cold.");
    }
}
```

1. `weatherInterpreter(4);` ?

2. `weatherInterpreter(32);` ?

This function checks eligibility to ride a rollercoaster, and requires that a person input their height in cm.

`admission(130.5);` is a valid function call.

Fill in the blank with the type and name of the parameter this expects.

```
public void admission(          ) {

    // checks height reqs
    if(height > 120) {
        System.out.println("You pass the height requirements.");
    }
    else {
        System.out.println("Sorry, you do not pass.");
    }
}
```

Multiple Parameters

Function definition

```
public void printPhoto(int width, int height, boolean inColor) {
    System.out.println("Width = " + width + " cm");
    System.out.println("Height = " + height + " cm");
    if(inColor) {
        System.out.println("Print is full color.");
    } else {
        System.out.println("Print is black and white.");
    }
}
```

Function call

```
printPhoto(30, 40, true);
```

Print output

```
Width = 30 cm
Height = 40 cm
Print is full color.
```

Argument Order

```
public void printPhoto(int width, int height, boolean inColor) {
    System.out.println("Width = " + width + " cm");
    System.out.println("Height = " + height + " cm");
    if(inColor) {
        System.out.println("Print is full color.");
    } else {
        System.out.println("Print is black and white.");
    }
}
```

Function call

```
printPhoto(10, 20, false);
```

Print output

```
Width = 10 cm
Height = 20 cm
Print is black and white.
```

Function call

```
printPhoto(20, 10, false);
```

Print output

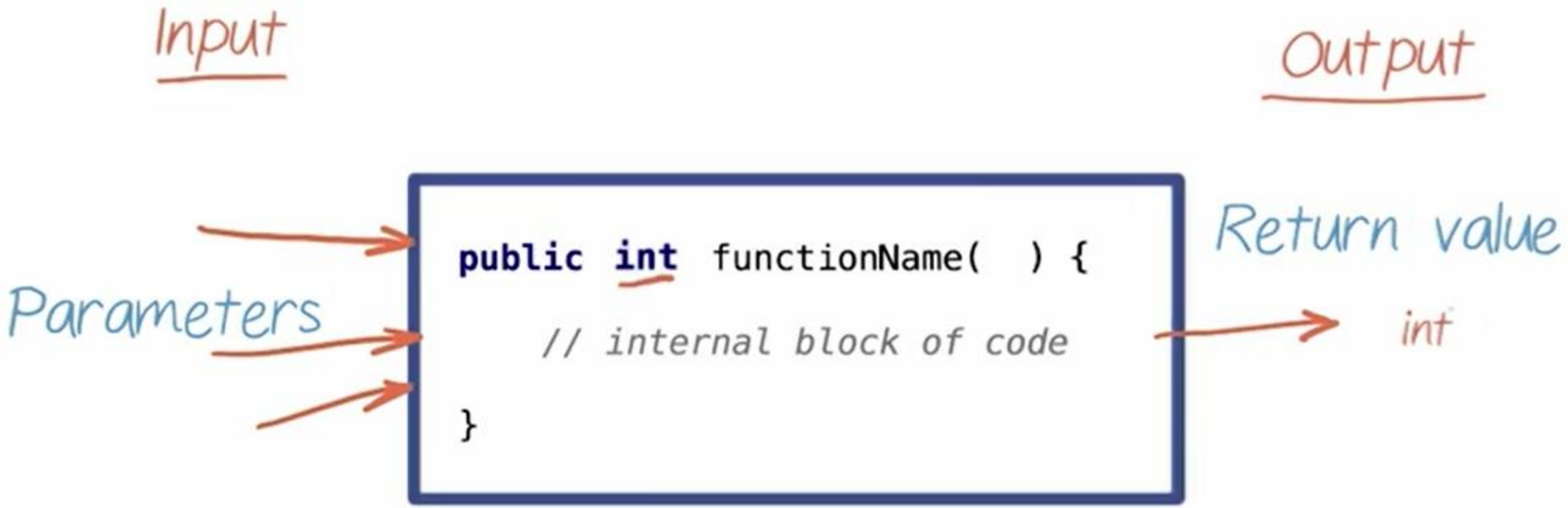
```
Width = 20 cm
Height = 10 cm
Print is black and white.
```

Write a valid function call for the function likePhoto

```
public void likePhoto(int currentLikes, String comment, boolean like) {
    //prints out comment and who is commenting
    System.out.println("Feedback: "+ comment);
    if(like) {
        // increase number of likes by 1
        currentLikes = currentLikes + 1;
    }
    System.out.println("Number of likes: " + currentLikes);
}
```

Write the function call here:

Return Values



Printing does not create output
that we can interact with later on
in separate programs

Tracking Photo Likes

Function definition

Parameters

```
public void likePhoto(int currentLikes, String comment, boolean like) {
    //prints out comment
    System.out.println("Feedback: "+ comment);
    if(like) {
        //increase number of likes by 1
        currentLikes = currentLikes + 1;
    }
    //print out current number of likes
    System.out.println("Number of likes: " + currentLikes);
}
```

Return value

currentLikes can only be reached inside our definition!

Need to output the value of currentLikes

Return Photo Likes

1 Function definition

```
public int likePhoto(int currentLikes, String comment, boolean like) {
    //prints out comment
    System.out.println("Feedback: "+ comment);
    if(like) {
        //increase number of likes by 1
        currentLikes = currentLikes + 1;
    }
    //print out current number of likes
    System.out.println("Number of likes: " + currentLikes);
    //return current likes
    return currentLikes;
}
```

2

Return value

1) a return type - void, int, String, etc.
2) a return statement - line of code that returns a value

Return Photo Likes

Function calls

```
int returnedLikes = likePhoto(0, "Nice color!", true);
```

↳ returns 1

❤ Likes: 1

Return Photo Likes

Function calls

```
int returnedLikes = likePhoto(0, "Nice color!", true);
                                  1
int totalLikes = likePhoto(returnedLikes, "I like this", true);
```

Returned output

```
returnedLikes = 1
```

This code won't return the String we want. Can you identify what's wrong with this code?

- passText isn't a String
- Wrong return type
- Wrong return statement
- Nothing is wrong

```
public void admission( double height ) {
    String passText;
    // checks height reqs
    if(height > 120) {
        passText =  "You pass the height requirements.";
    }
    else {
        passText = "Sorry, you do not pass.";
    }
    return passText;
}
```

Write a function that returns the correct change.

```
makeChange(double itemCost, double dollarsProvided)
```

Function definition

```
public double makeChange(double itemCost, double dollarsProvided) {
    // calculate change
    // change is dollars - item cost
    double change  = dollarsProvided - itemCost;
    return change;
}
```

Function call

```
double returnedChange = makeChange(3.60, 5.75);
```

Returning Random Numbers

```
// random num between 0 and (almost) 1
double randomNumber = Math.random();

// change range to 0 to (almost) 10
randomNumber = randomNumber * 10;
```

→ 0 - 0.999..

← multiply

0 * 10 = 0 min

0.999 * 10 = 9.99 max

Returning Random Numbers

```
// random num between 0 and (almost) 1
double randomNumber = Math.random();

// change range to 0 to (almost) 10
randomNumber = randomNumber * 10;

// cast to integer (ignore decimal part)
// ex. 9.985 becomes 9
int randomInt = (int) randomNumber;
```

multiply

int

Cast

9.985 → 9

1.252 → 1

Casting

Turning one variable type into another

(ex. double to an int)

0.5728 → 0

Rolling Dice

```
public int rollDice() {
    // random num between 0 and (almost) 1
    double randomNumber = Math.random();

    // change range to 0 to (almost) 6
    randomNumber = randomNumber * 6;

    // shift range up one
    randomNumber = randomNumber + 1;

    // cast to integer (ignore decimal part)
    // ex. 6.998 becomes 6
    int randomInt = (int) randomNumber;

    // return statement
    return randomInt;
}
```

range of 6 total numbers

range = 1 - (almost) 7

casting!

range = 1 - 6 (integers)

Use the parameter **int** sides to change our code and produce the correct random roll range each time.

Roll range = 1–8

Roll range = 1–10

```
public int rollDice(int sides) {
    // random num between 0 and (almost) 1
    double randomNumber = Math.random();

    // change range to 0 to the number of sides
    randomNumber = randomNumber * sides;

    // shift range up one
    randomNumber = randomNumber + 1;

    // cast to integer (ignore decimal part)
    // ex. 6.998 becomes 6
    int randomInt = (int) randomNumber;

    // return statement
    return randomInt;
}
```

Change the multiplier!

ALARM

```
/**
 * makes a single beep sound
 */
public void beep()


/**
 * @return true if alarm is on, false if off
 */
public boolean checkAlarm()


/**
 * Keep beeping until snoozed
 */
public void alarm()
```

ALARM

```
public void alarm(){
    boolean on = checkAlarm();
    if(on){
        beep();
        on = checkAlarm();
    }
    if(on){
        beep();
        on = checkAlarm();
    }
    if(on){
        beep();
        on = checkAlarm();
    }
    ...
}
```

WHILE LOOPS

```
public void alarm(){
    boolean on = checkAlarm();
    if(on){
        beep();
        on = checkAlarm();
    }
    if(on){
        beep();
        on = checkAlarm();
    }
    if(on){
        beep();
        on = checkAlarm();
    }
    ...
}
```

```
while(on){
    beep();
    on = checkAlarm();
}
```

1 Googol =

10000000000000000000000000000000000
000000000000000000000000000000000000
000000000000000000000000000000

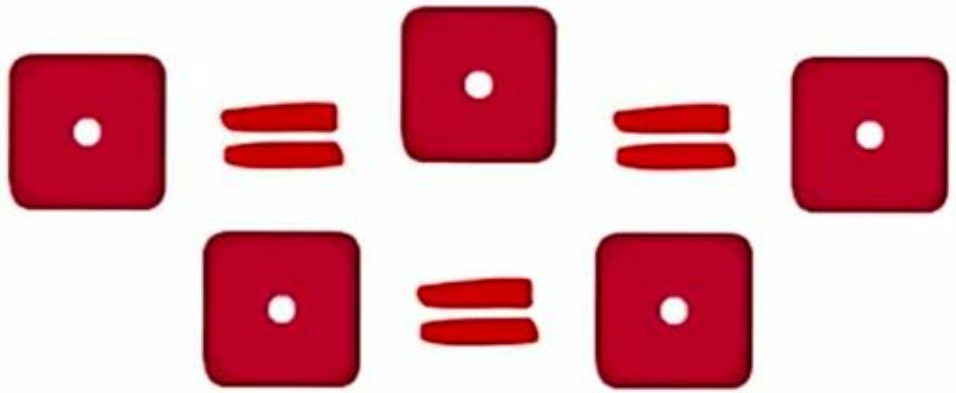
```
long largest = 9223372036854775807

String googol = "100000000000000000000000000000000000000000000..."
```

What's the condition ?

```
String googol = "1";
int len = googol.length();
while(len < 101){
   googol = googol + "0";
   len = googol.length();
}
```

googol	len	len < 101
"1"	1	true
"10"	2	true
"100"	3	true
"1000"	4	true
"100.."	..	true
"100.."	..	true
"10^97"	98	true
"10^98"	99	true
"10^99"	100	true
"10^100"	101	false

ROLL A YAHTZEE



```
public int keepRolling(){
    int dice1 = rollDice();
    int dice2 = rollDice();
    int dice3 = rollDice();
    int dice4 = rollDice();
    int dice5 = rollDice();
    int count = 1;
    while(!(dice1 == dice2 && dice2 == dice3 &&
            dice3 == dice4 && dice4 == dice5)){
        //we need to re-roll
        dice1 = rollDice();
        dice2 = rollDice();
        dice3 = rollDice();
        dice4 = rollDice();
        dice5 = rollDice();
        count = count + 1;
    }
    return count;
}
```

FOR LOOPS

```
public void raiseAlarm (int numOfWarnings){
    int i = 1;                                  (1) loop counter
    while (i <= numOfWarnings) {                (2) loop condition
        System.out.println("Warning");
        i++;                                    (3) loop increment
    }
}
```

While loop

```
public void raiseAlarm(int numOfWarnings){
              (1)          (2)          (3)
    for(int i = 1; i <= numOfWarnings ; i++ ){
        System.out.println("Warning");
    }
}
```

For loop

LOOP COUNTERS

```
for(int i = 1; i <= 3  ; i++){
    i = i-1;
    System.out.println(i);
}
```

i	i <= 3
1	true
0	
1	true
0	
1	true
0	
...	...

Print output
0
0
0
...

Infinite loop !!!

LOOP COUNTERS

```
for(int i = 1;  i <= 3  ; i++){
    System.out.println(i*2);
}
```

i	i <= 3
1	true
2	true
3	true
4	false

Print output

```
2
4
6
```

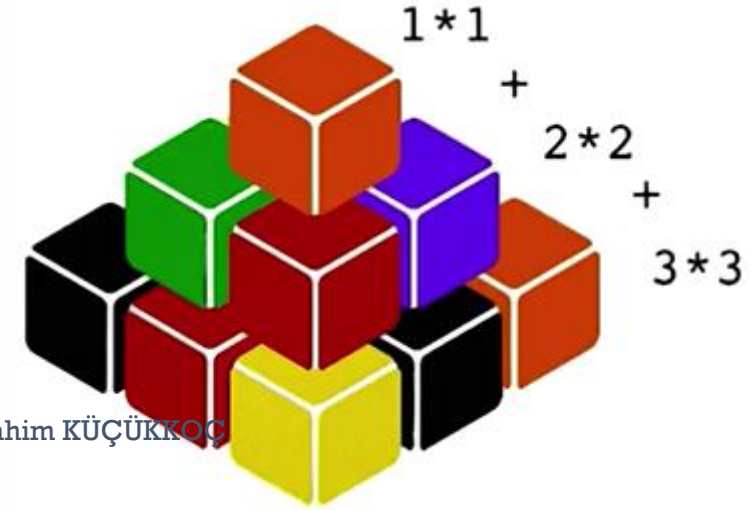
Counting the blocks

```
public int countBlocks(int levels){
    int total = 0;
    for(int i=1; i<=levels; i++){
        total = total + (i*i);
    }
    return total;
}
```

3 (levels)

3 (levels)

i	i<=3	(i*i)	total
1	true	1	1
2	true	4	5
3	true	9	14
4	false		



LOOP COUNTERS

```java
/**
 * Adds the numbers 15 + 16 + .. + 20
 * @return the total sum
 */
public int addNumbers() {
    int sum = 0;
    for (int i = 15; i <= 20; i++) {
        sum = sum + i;
    }
    return sum;
}
```

sum	i	i <= 20
0	15	true
15	16	true
31	17	true
48	18	true
66	19	true
85	20	true
105	21	false

LOOP COUNTERS

While loop

```
int i=0;
while(i<=10){
    System.out.println(i);
    i++;
}
```

For loop

```
for (int j = 0; j <= 10; j++) {
    System.out.println(j);
}
```

LOOP COUNTERS

Instruction	Short-hand	Effect
`i = i+1;`	`i++;`	10 → 11
`i = i-1;`	`i--;`	10 → 9
`i = i+5;`	`i += 5;`	10 → 15
`i = i-6;`	`i -= 6;`	10 → 4
`i = i*3;`	`i *= 3;`	10 → 30
`i = i/2;`	`i /= 2;`	10 → 5

Java that uses for-loop, string

```
public class Program {
    public static void main(String[] args) {

        String value = "art";

        // Loop from 0 to length() of the string.
        for (int i = 0; i < value.length(); i++) {
            // Get letters with charAt method.
            char letter = value.charAt(i);
            System.out.println(letter);
        }
    }
}
```

Output

```
a
r
t
```


FOR LOOP STEPS

```java
/**
 * Print the even numbers between 2 and 10
 */
public void evenNumbers(){
    for (int i = 2; i <= 10; i+=2) {
        System.out.println(i);
    }
}
```

i	i <= 10
2	true
4	true
6	true
8	true
10	true
12	false

Print output

```
2
4
6
8
10
```

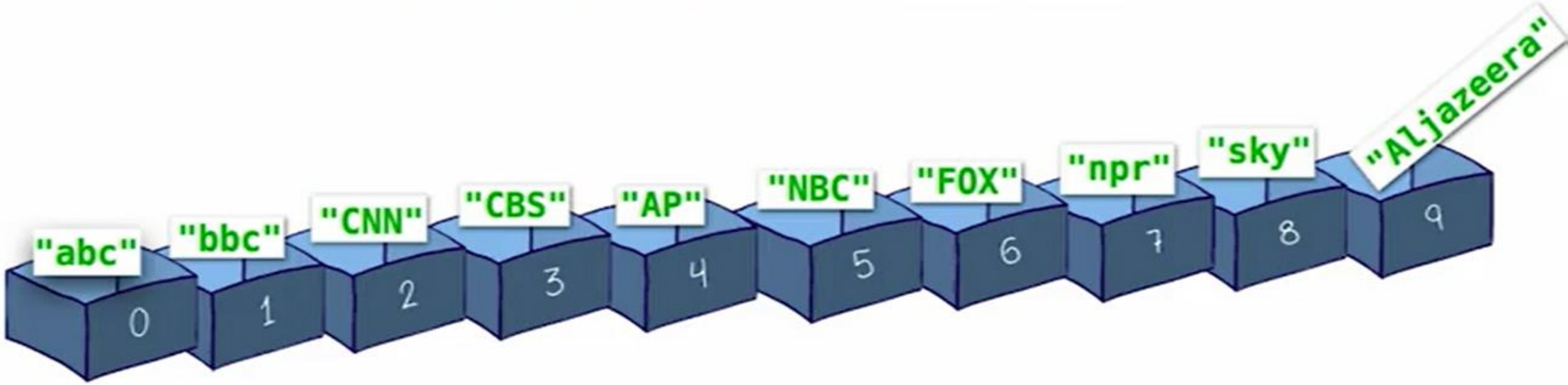
WIFI SEARCH

```java
/**
 * WiFi search
 */
public void searchWifi(){
    for(int i = 1;i <=10; i++) {
        boolean wifiAvailable = checkWifi();
        if (wifiAvailable) {
            System.out.println("Wifi found");
            break;
        }
    }
}
```

ROLL A SIX

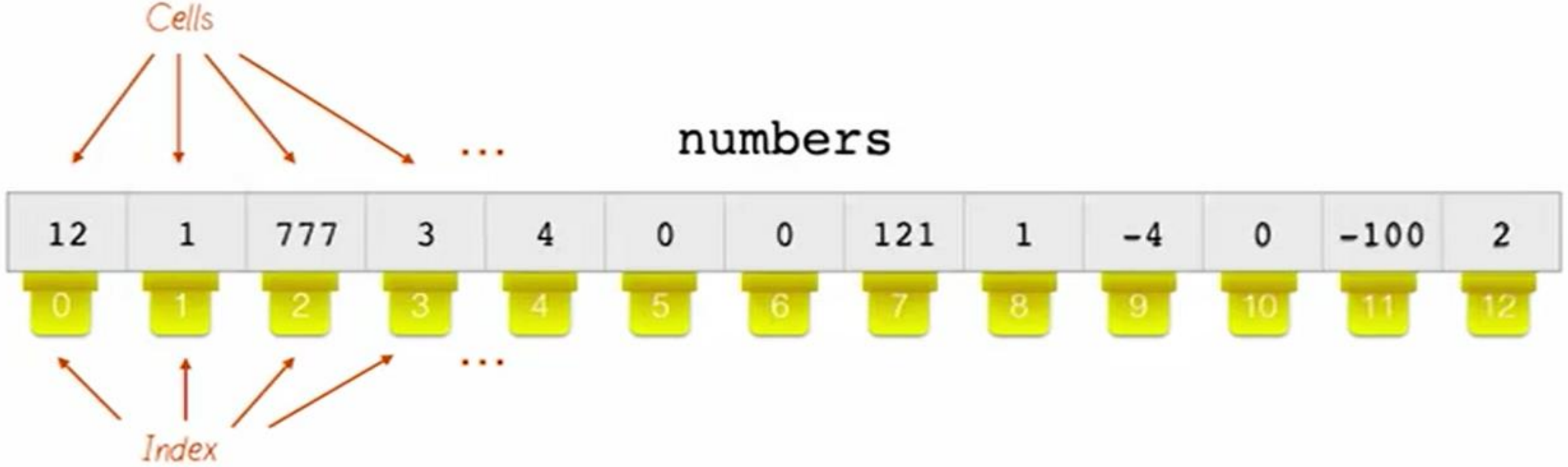
```
/**
 * Rolls a dice till you get a 6 then you win
 * but if you get a 3 you lose
 */
public boolean rollASix(){
    int dice = rollDice();
    while(dice!=6){
        dice = rollDice();
        if(dice == 3)
            break;
    }
    if(dice == 6)
        return true;
    else
        return false;
}
```

ARRAY OF STRINGS




```
newsOutlet

newsOutlet[0]  ⟶  "abc"
newsOutlet[1]  ⟶  "bbc"
newsOutlet[2]  ⟶  "CNN"
```

ARRAY OF INTEGERS




```
int sum = numbers[0]+numbers[3];
```

sum ⟶ 15

ARRAY OF INTEGERS

numbers


```
int [] numbers ={12,1,777,3,4,0,0,121,1,-4,0,-100,2};
System.out.print(numbers[0]);
System.out.print(numbers[9]*numbers[12]);
```

Print output

```
12
-8
```

ARRAY OF DOUBLES


```
double [] fracNumbers = {4.0, 6.2, 3.75, 5.1, 9.99};
```

Pick a random press

newsOutlet

0	"abc"
1	"bbc"
2	"CNN"
3	"CBS"
4	"AP"
5	"NBC"
6	"FOX"
7	"npr"
8	"sky"
9	"Aljazeera"

PRESS CONFERENCE

```
String [] newsOutlet = {"abc","bbc","CNN","CBS","AP","NBC","FOX","npr","sky","Aljazeera"};

double lucky = Math.random();
//Multiply by 10 to get a fractional number between 0 - 10
lucky *= 10;

//cast to integer to get an integer number between 0 - 9
int luckyIndex = (int) lucky;

System.out.print(newsOutlet[luckyIndex]);
```

lucky	lucky*10	luckyIndex	Print output
0.423	4.23	4	"AP"
0.987	9.87	9	"Aljazeera"
0.611	6.11	6	"FOX"
0.123	1.23	1	"bbc"

Arrays and Loops

```
public double calculateAverage(double [] temperatures){
    int size = temperatures.length;
    double total = 0;
    for(int i=0; i<size; i++){
        total += temperatures[i];
    }
    double average = total/size;
    return average;
}
```

Number of items in the array

loop counter: (0, 1, 2 ...)

Arrays and Loops

{75,73,72,80}

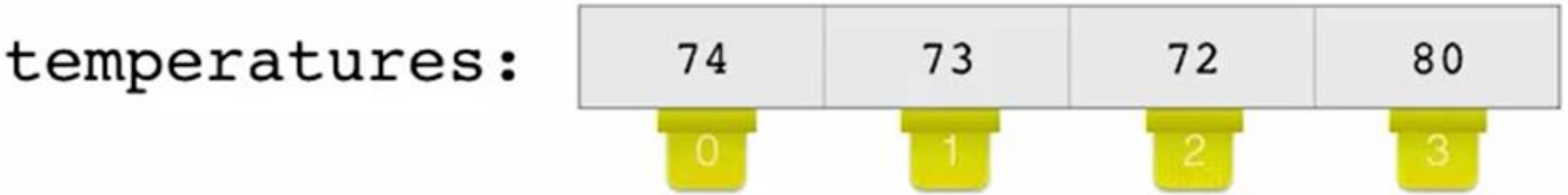
```
public double calculateAverage(double [] temperatures){
    int size = temperatures.length;
    double total = 0;
    for(int i=0; i<size; i++){
        total += temperatures[i];
    }
    double average = total/size;
    return average;
}
```

4

300 / 4 = 75

i	temperatures[i]	total
0	75	75
1	73	148
2	72	220
3	80	300
4		

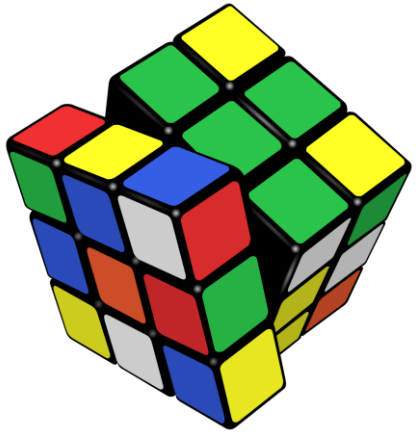
Array bounds




```
int size = temperatures.length;
```
← 4

```
System.out.println(temperatures[10]);
```
Error!

ArrayIndexOutOfBoundsException



Leaderboard	
Trish McCallister	7.85
Jayden McNeill	7.32
Lucas Etter	4.9
Michal Pleskowicz	6.22
Jiayu Wang	5.4
Kevin Costello	7.3
Pavan Ravindra	5.19

ARRAY SEARCH

```java
/**
 * Searches an array of speed
 * for the fastest (smallest) value.
 * @param speed array of speeds
 * @return double the fastest speed found
 */
public double search(double[] speed){
    int size = speed.length;
    double min = speed[0];
    for(int i=1 ; i<size ; i++){
       if(speed[i]<min){
           min = speed[i];
       }
    }
    return min;
}
```

STRING ARRAY SEARCH

Write a function that searches an array of Strings and finds the longest string

```
public String findLongestName(String [] names){

}
```

REMEMBER:

`name.length()` returns the length of a string

STRING ARRAY SEARCH

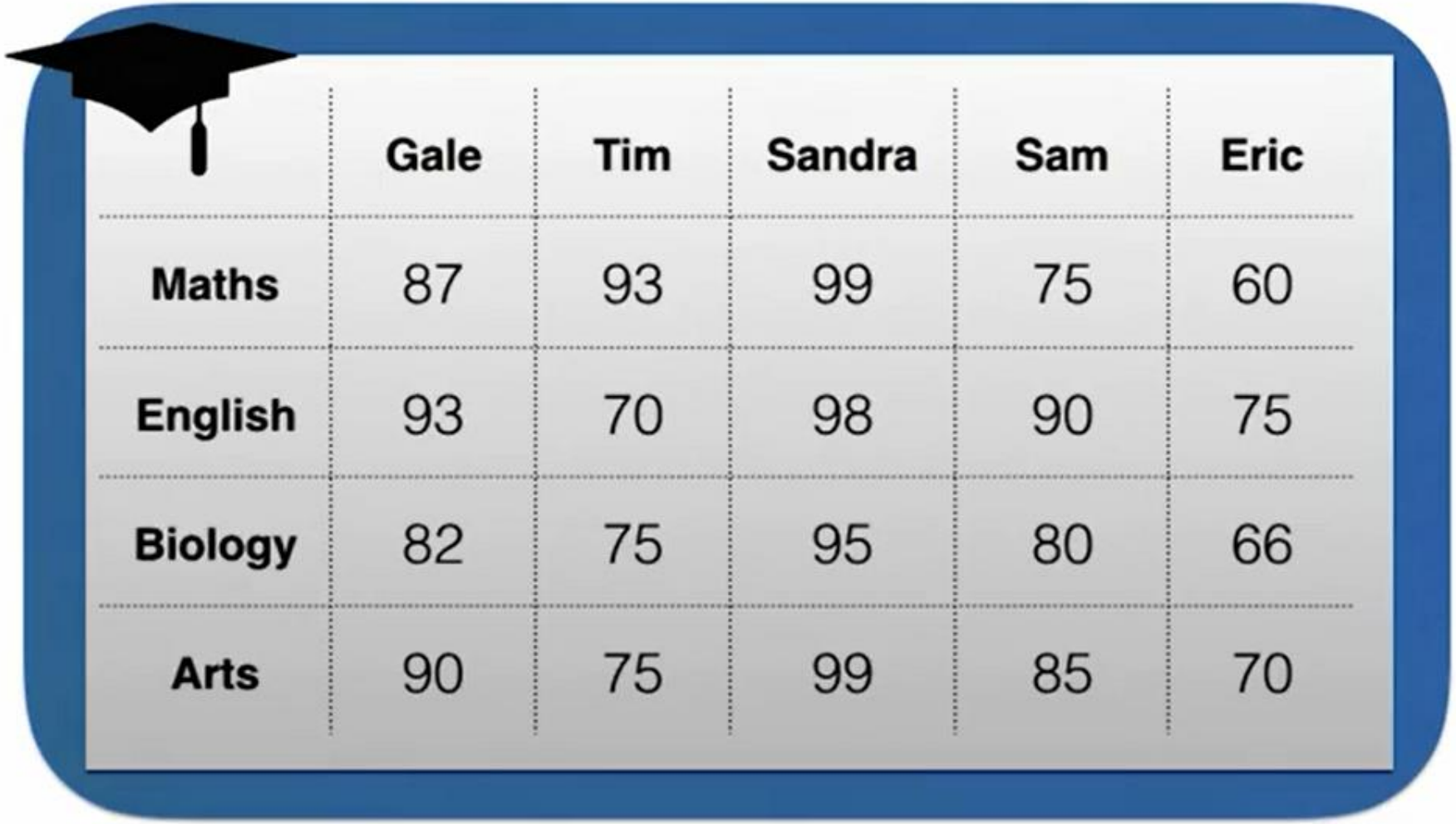
```java
/**
 * Searches an array of names
 * for the longest name.
 * @param names array of names
 * @return String: the longest name found
 */
public String findLongestName(String [] names){
   int size = names.length;
   String longestName = names[0];
   for(int i=1; i<size; i++){
     if(names[i].length() > longestName.length()){
         longestName = names[i];
     }
   }
   return longestName;
}
```

TIE BREAKERS ?

```java
public String findLongestName(String [] names){
    int size = names.length;
    String longestName = names[0];
    for(int i=1; i<size; i++){
        if(names[i].length() > longestName.length()){
            longestName = names[i];
        }
    }
    return longestName;
}
```

longestName ?

- "Kimberli"
- "Apolonia"
- Both of them
- None of them



	Gale	Tim	Sandra	Sam	Eric
Maths	87	93	99	75	60
English	93	70	98	90	75
Biology	82	75	95	80	66
Arts	90	75	99	85	70

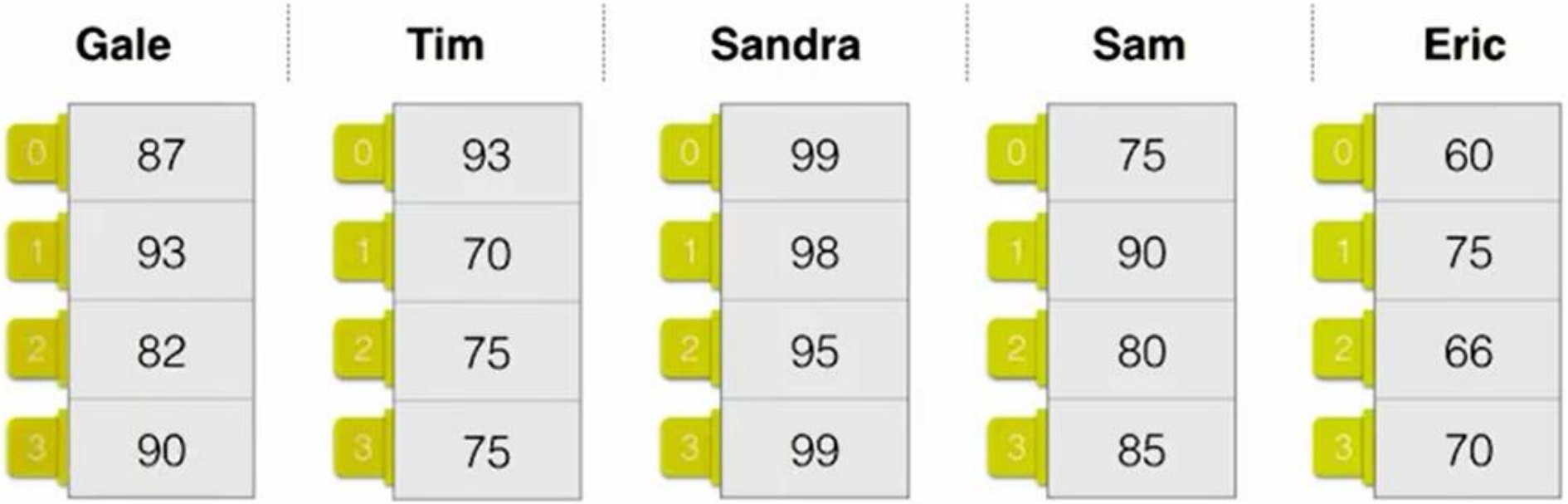
int []

4

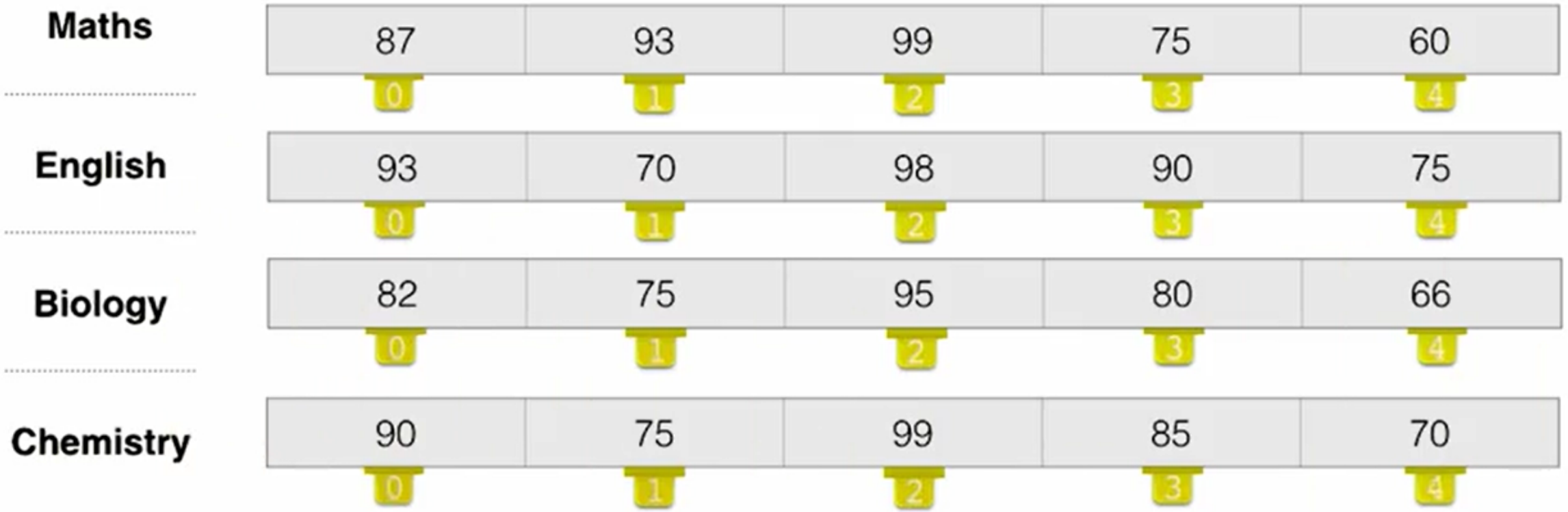
5

= 20 Grades

Array per student



Array per subject



Looping over 2D Arrays

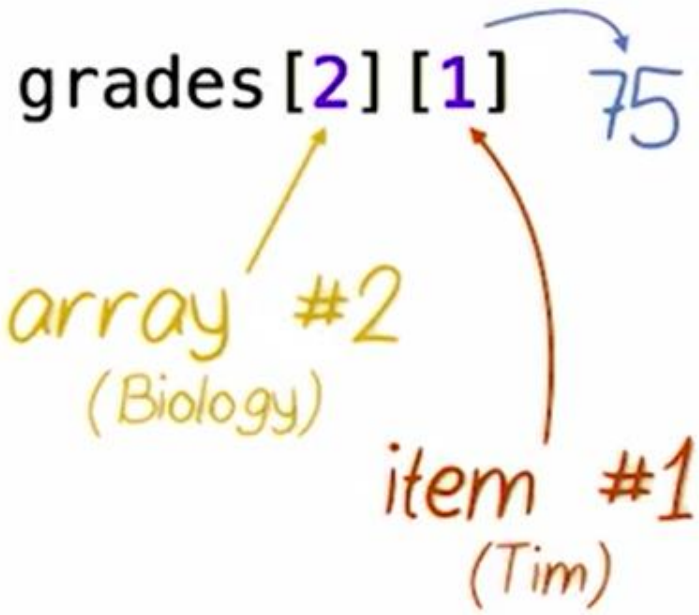
	Gale (0)	Tim (1)	Sandra (2)	Sam (3)	Eric (4)
Maths (0)	87	93	99	75	60
English (1)	93	70	98	90	75
Biology (2)	82	75	95	80	66
Arts (3)	90	75	99	85	70

```
for(i=0; i<5; i++) {
    grades[1][i]
}
```

English

2D Array

	Gale (0)	Tim (1)	Sandra (2)	Sam (3)	Eric (4)
Maths (0)	87	93	99	75	60
English (1)	93	70	98	90	75
Biology (2)	82	75	95	80	66
Arts (3)	90	75	99	85	70



Looping over 2D Arrays

```
public static double englishAverage(int [][] grades){
    int total = 0;
    for(int i=0; i<5; i++) {
        total += grades[1][i];
    }
    double average = total/5.0;
    return average;
}
```

Loop over all students in the english array

```
grades[1][i]
```

English Every student

Looping over 2D Arrays

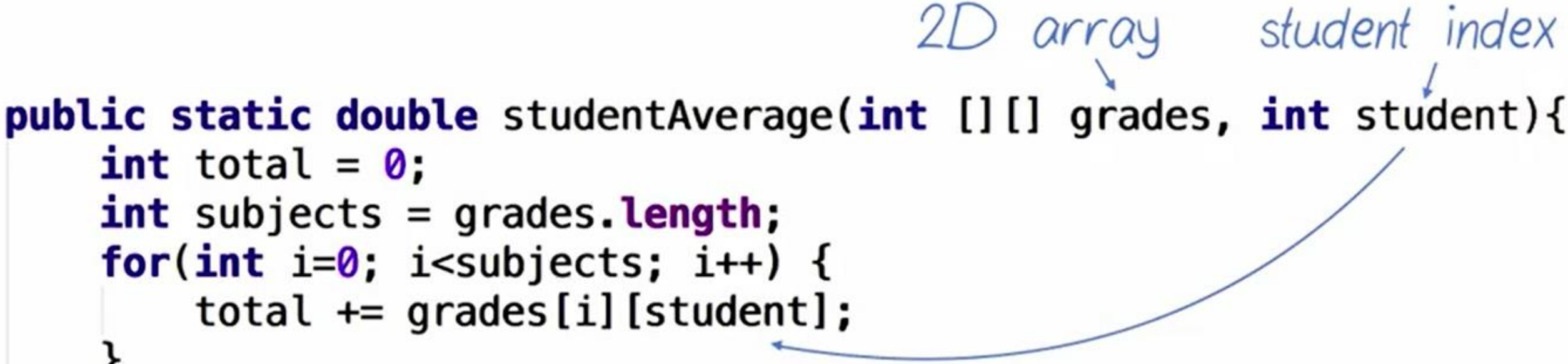
```
public static double sandraAverage(int [][] grades){
    int total = 0;
    for(int i=0; i<4; i++) {
        total += grades[i][2];
    }
    double average = total/4.0;
    return average;
}
```

Loop over all grades under Sandra

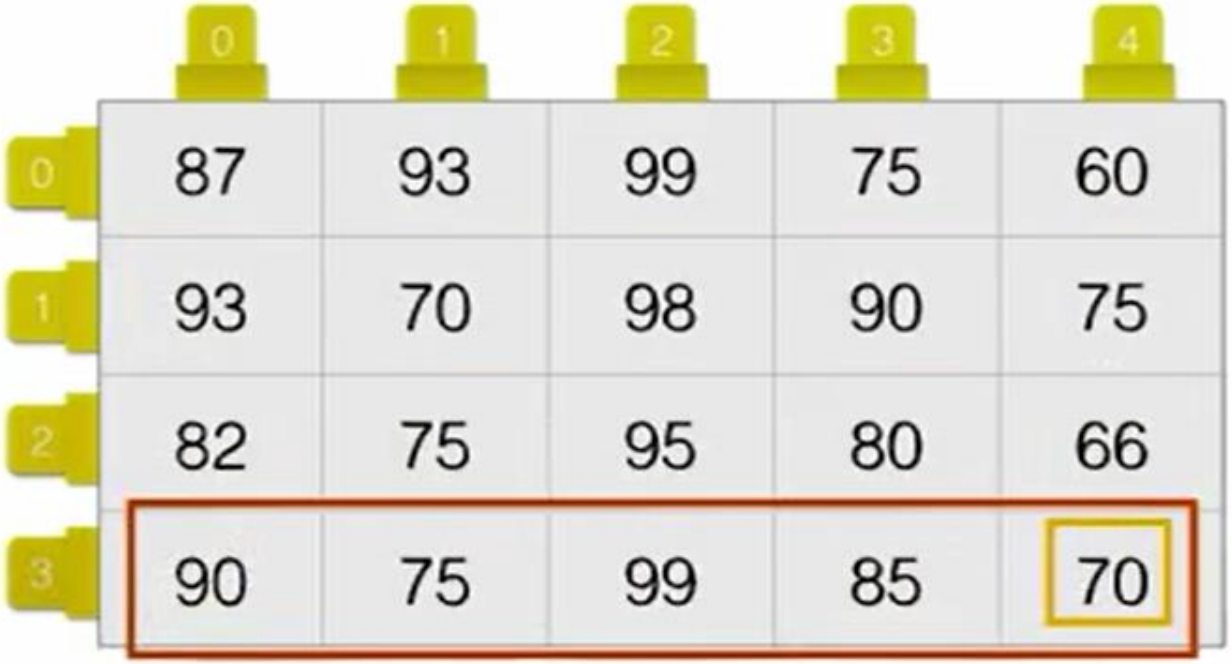
grades[i][2]

Every Subject Sandra

Looping over 2D Arrays



Nested loops with 2D arrays



	0	1	2	3	4
0	87	93	99	75	60
1	93	70	98	90	75
2	82	75	95	80	66
3	90	75	99	85	70

```
int total = 0;
for(int i=0; i<4; i++){
    for (int j=0; j<5; j++){
        total += grades[i][j];
    }
}
```

References

- Udacity Java Programming Lectures, https://in.udacity.com/course/intro-to-java-programming--cs046