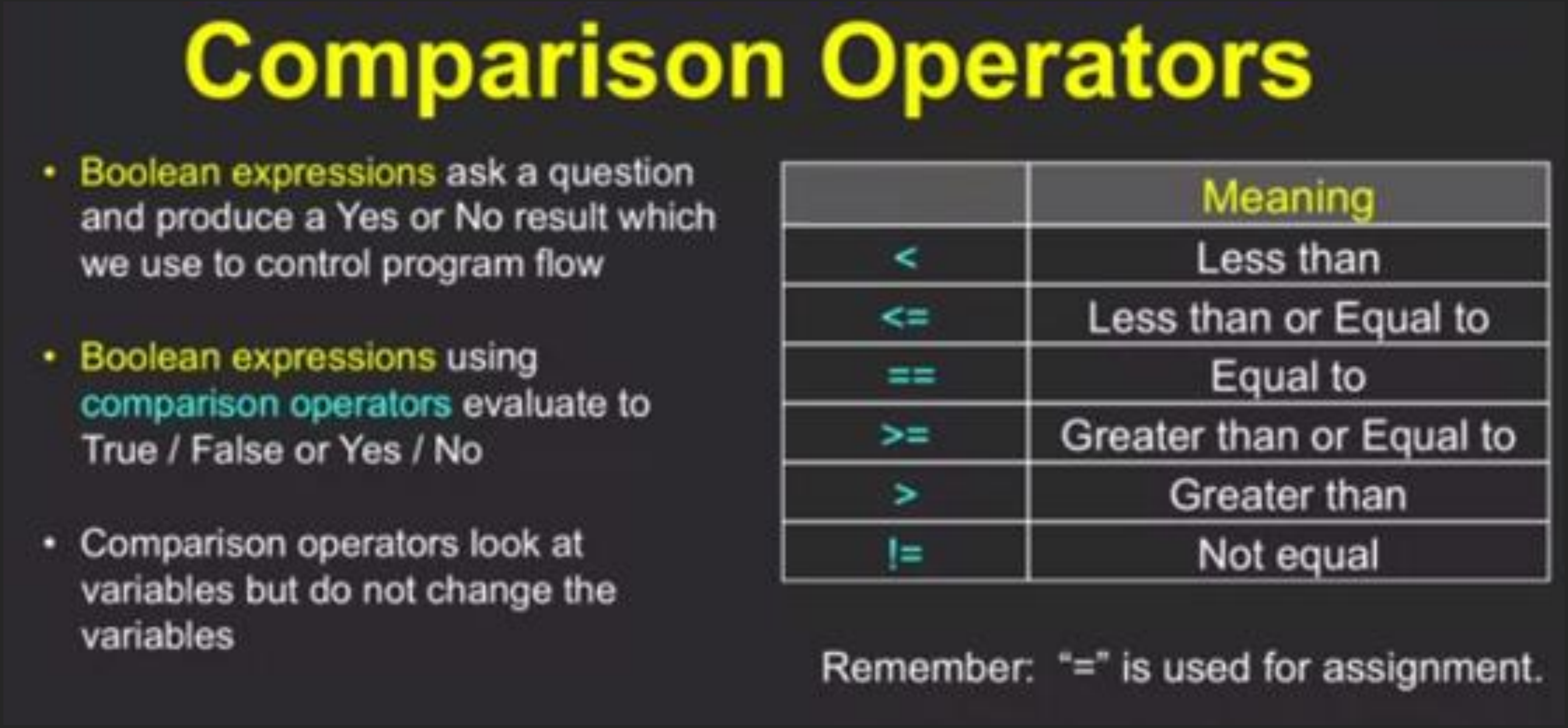
Get Smart: With Java Programming

Yaman Omar Alashqar

```
System.out.println(“WELCOME TO THIS COURSE\n”);
```

LECTURE 3



Comparison Operators

- Boolean expressions ask a question and produce a Yes or No result which we use to control program flow

- Boolean expressions using comparison operators evaluate to True / False or Yes / No

- Comparison operators look at variables but do not change the variables

	Meaning
<	Less than
<=	Less than or Equal to
==	Equal to
>=	Greater than or Equal to
>	Greater than
!=	Not equal

Remember: "=" is used for assignment.

Basic operations on Booleans

!

Not (Flips the output)
!= (Not Equal to) >>> Only 1, 2, or few conditions should not be met
e.g. When we have (A/B) then B should be != 0 (divide by zero)

&&

And
>>> All conditions must be true
e.g. To login your "username" **AND your "password" should be correct**

||

Or
>>> At least one condition must be true
e.g. You can submit your homework either on youtube **OR** on facebook
e.g. To finish the payment you can pay by a credit card **OR** using cash

Program to calculate salary based on working hours

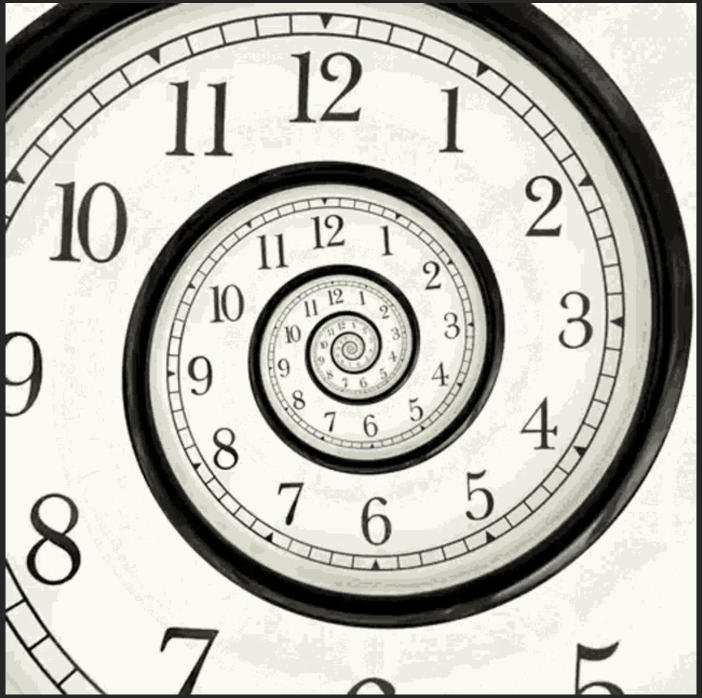
Input: hourly rate, weekly hours (A month is 4 weeks), month overtime (overtime hours are paid one and a half)

Print True if the user has worked for overtime, and false otherwise

Print True if the user earns more than $400 dollars per month

What is your standard hourly pay rate?
How many hours do you work in a week (without overtime)?
How many hours do you work as over time per month?

Read an integer variable named “inHours”
which represents time in hours,
then save the result in minutes
using another variable “inMinutes”



Read an integer variable named “inHours” which represents time in hours, then save the result in minutes in another variable named “inMinutes”

```
 int inHours , inMinutes;

Scanner input = new Scanner(System.in);
System.out.println("Enter Number of Hours: ");
inHours = input.nextInt();

inMinutes = inHours * 60;
System.out.println(" In minutes: "+ inMinutes );
```

WHAT IS A STRING

in Java, a string is:
an object that represents a sequence of characters (letters, spaces, numbers, or symbols).

They are typically used to represent text or speech. Similar to how we represent speech in writing, we surround strings double quotes ("...").

The java.lang.String class is used to create a string object.

- Each variable in Java has a specific type, which determines the size of memory, the range of values that can be stored and the set of operations that can be applied to the variable
- Datatypes can be categorized into two major types:

Primitive

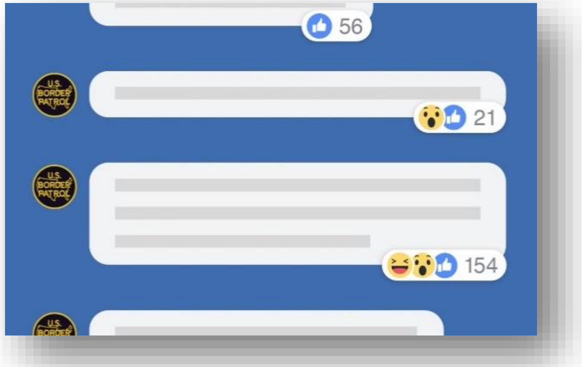
for storing
simple values

EXAMPLES:
Numbers, single character

Reference
Non Primitive

for storing
complex objects

EXAMPLES:
Date , mail message , Post
~~String, Array~~



Non Primitive can implicitly contain multiple primitive data types (but not always)

So far, we have learned primitive data types, which are the simplest types of data with no built-in behavior.

Our programs will also use Strings, which are objects, instead of primitives. Objects have built-in behavior.

So, what can we use strings for?

- To display data that uses text or symbols,
  (like printing our name on the screen, a Facebook post)

- To add or remove text.
  (we can break strings into smaller strings, or combine strings to make longer ones)

- To modify characters.
  (For example, we could capitalize the first letter of every word in a string if wanted to turn it into a title)

The string “22” is different from the numerical value of 22.

While they appear to be the same, a computer would see the first as two characters: '2' and '0', while the second contains the numerical value of 20.

What is the output?

```
System.out.println("RESULT = " + 20 + "5" );
```

The string “22” is different from the numerical value of 22.

```
String intString = "22";
int result = Integer.parseInt(intString);
// TRY result+2;
//TRY intString+2;
System.out.println(result);
```

```
 // Get the double value
double data = 345.32145;

// convert into int
int value = (int)data;

// print the int value
System.out.println(value);
```

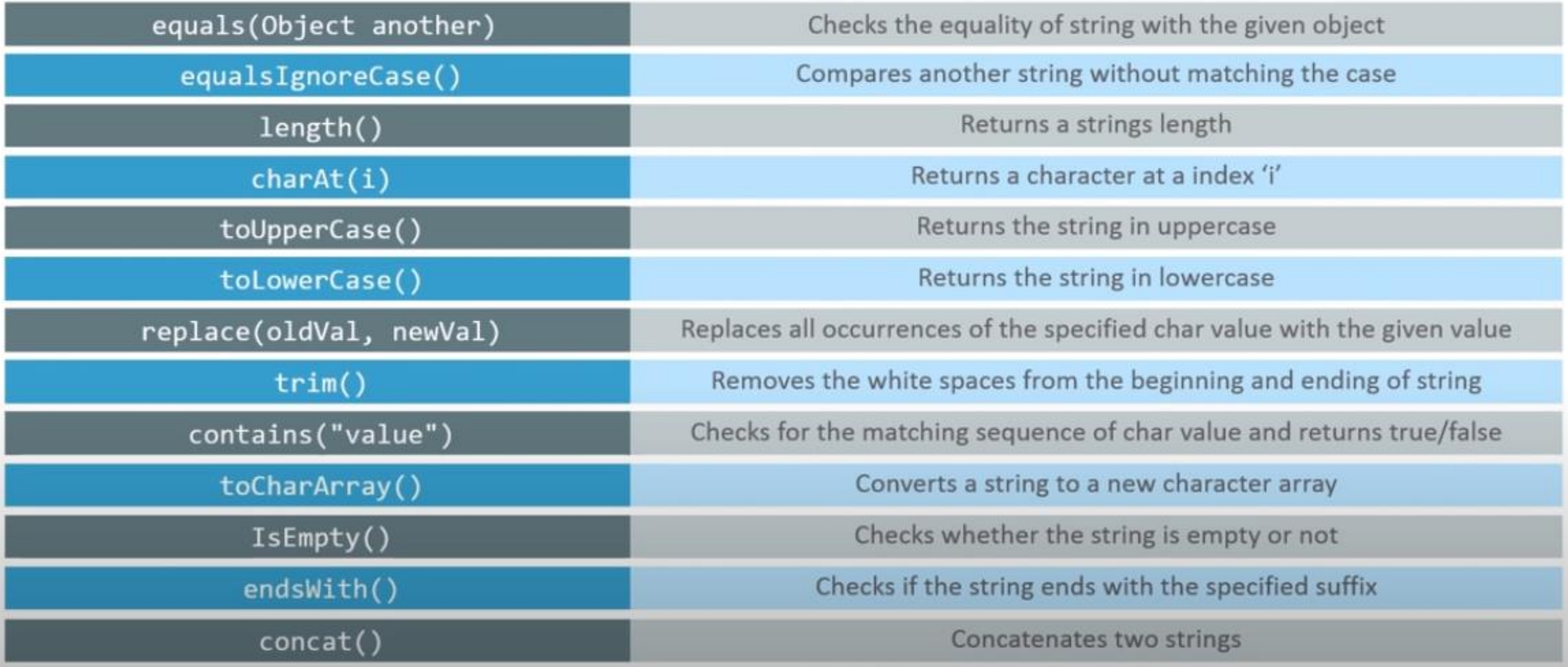
Reading Strings from the Keyboard

- next(): reads a single token.
- nextLine(): reads the whole line.

```
Scanner input = new Scanner(System.in);
s = input.nextLine();
```

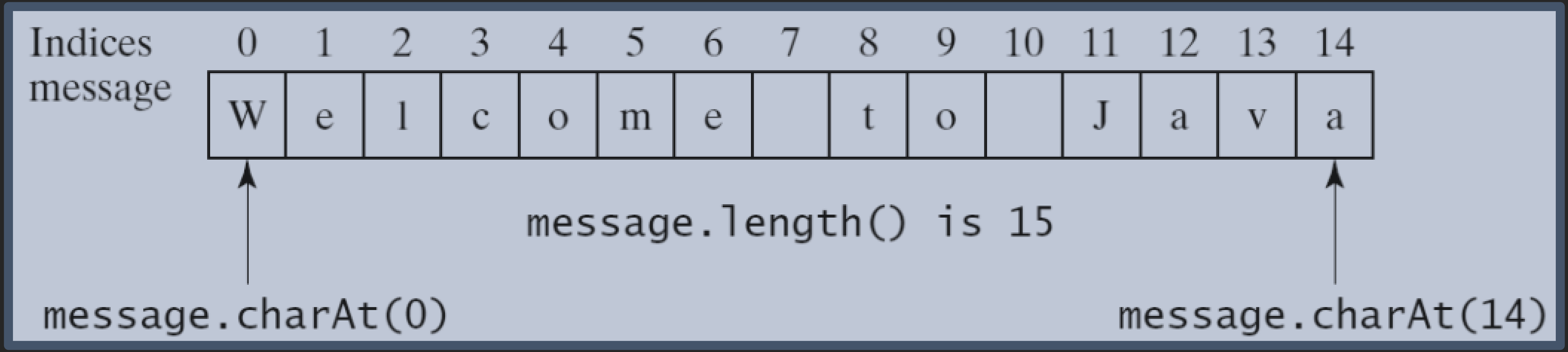
Java String class provides a lot of methods to perform operations on strings

such as compare(), concat(), equals(), split(), length(), replace(), compareTo(), substring() etc.



`equals(Object another)`	Checks the equality of string with the given object
`equalsIgnoreCase()`	Compares another string without matching the case
`length()`	Returns a strings length
`charAt(i)`	Returns a character at a index ‘i’
`toUpperCase()`	Returns the string in uppercase
`toLowerCase()`	Returns the string in lowercase
`replace(oldVal, newVal)`	Replaces all occurrences of the specified char value with the given value
`trim()`	Removes the white spaces from the beginning and ending of string
`contains("value")`	Checks for the matching sequence of char value and returns true/false
`toCharArray()`	Converts a string to a new character array
`IsEmpty()`	Checks whether the string is empty or not
`endsWith()`	Checks if the string ends with the specified suffix
`concat()`	Concatenates two strings

Strings are a linear sequence of characters



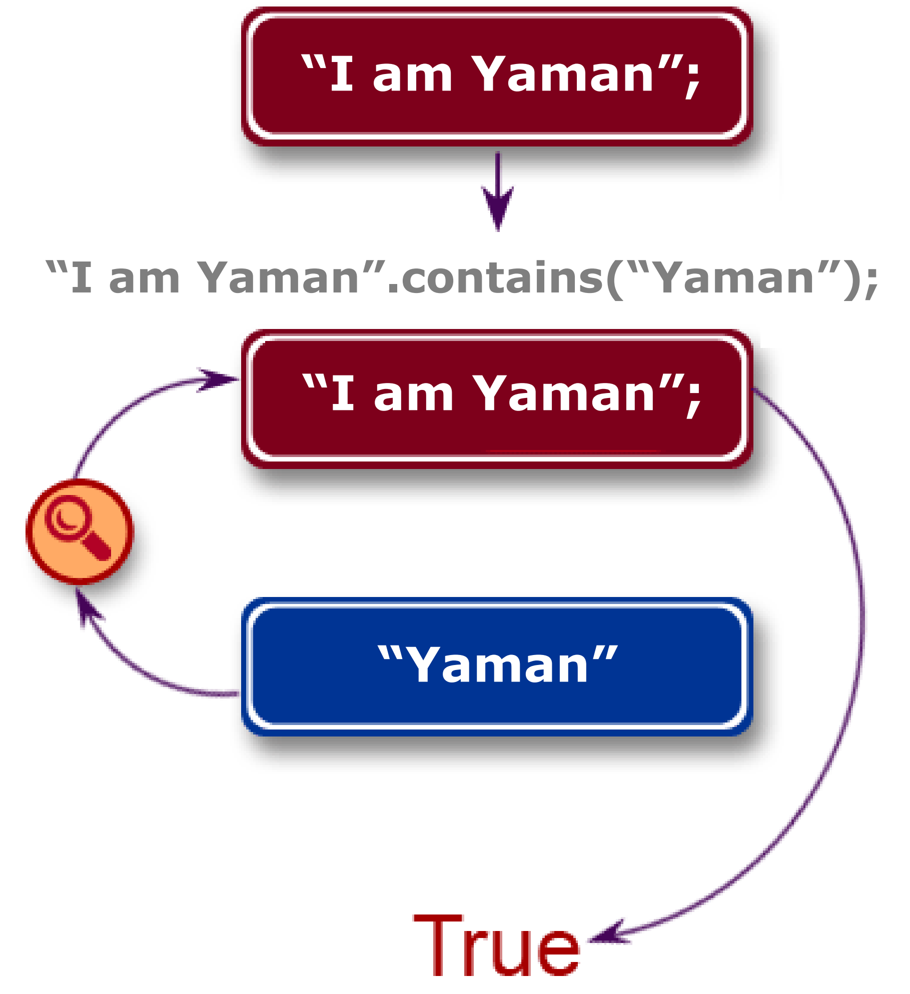
Indices	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
message	W	e	l	c	o	m	e		t	o		J	a	v	a

int length()	returns string length (count of total number of characters)

```
String s1=“Yaman";
String s2=“Cyber Deal";
System.out.println("string length is: "+s1.length());
System.out.println("string length is: "+s2.length());
```

contains()

```
String s1 = “I am Yaman”;
s1.contains(“Yaman");
```



“I am Yaman”;

“I am Yaman”.contains(“Yaman”);

“I am Yaman”;

“Yaman”

True

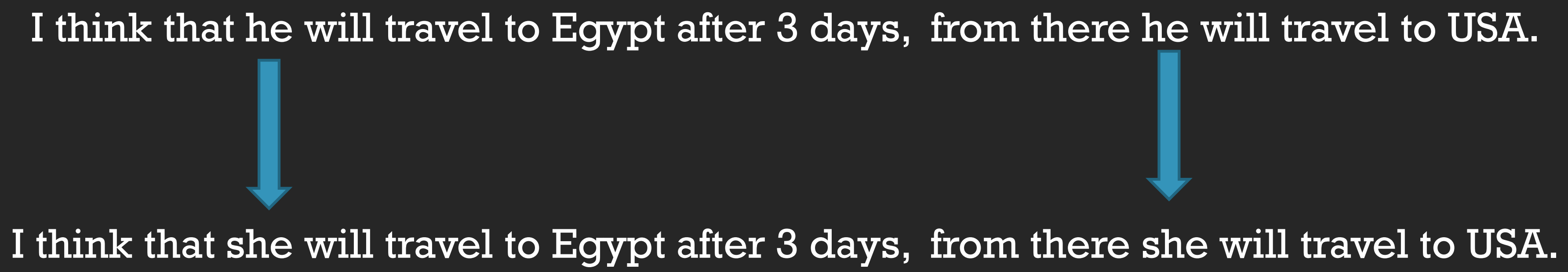
contains()

```
boolean keyWordCar = post.contains(“mercedes") || post.contains(“bmw") || post.contains(“audi") ;

boolean keyWordApp = post.contains(“android") || post.contains(“ios") || post.contains(“app") ;

boolean keyWordLang = post.contains(“english") || post.contains(“course") || post.contains(“learn") ;
```

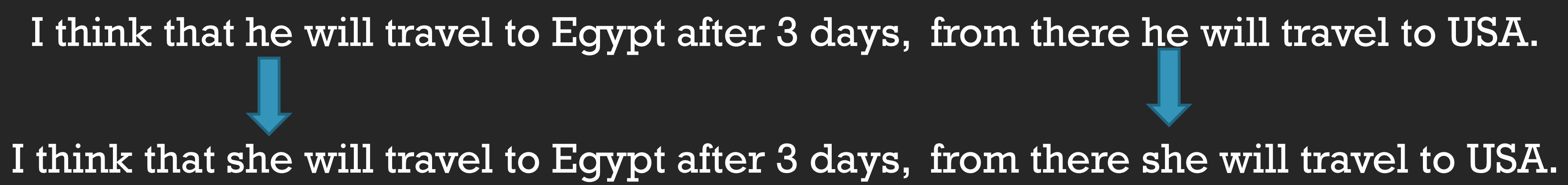
replace(s1,s2)



I think that he will travel to Egypt after 3 days, from there he will travel to USA.

I think that she will travel to Egypt after 3 days, from there she will travel to USA.

replace(s1,s2)



I think that he will travel to Egypt after 3 days, from there he will travel to USA.

I think that she will travel to Egypt after 3 days, from there she will travel to USA.

```
String he = "I think that he will travel to Egypt after 3 days, from there he will travel to USA.";
System.out.println( he.replace("he" , "she"));
```

\b allows you to perform a “whole words only” search using a regular expression in the form of \bword\b.

concat	

```
String firstName = “Yaman”;
String lastName = “Alashqar”;
String fullName =
```

```
String firstName = “Yaman”;
String lastName = “Alashqar”;
String fullName = firstName + lastName; //YamanAlashqar
//DON’T FORGET TO ADD A SPACE IN BETWEEN
```

POINTS	CONCAT() METHOD	+ OPERATOR
Definition	A concat() method is method to combine two strings .	+ operator used to concatenate any number of strings.
Number of arguments	In concat() method, takes only one argument of string and concatenate it with another string.	In + operatortakes any number of arguments and combines all strings.
Type of arguments	concat() method takes arguments of string type only.	+ operator takes any type of argument and converts it to string type and then combine them.
Creates new string	concat() takes concatenates two strings and return new string object only string length is greater than 0, otherwise it returns same object..	+ operatorcreates a new string object every time irrespective of length of string.
NullPointer Exception	In concat() method raises NullPointer Exception when string is concatenated with null .	+ operator concatenates string with without any error.
Performance	concat() method is better than + operator because it creates a new object only when the string length is greater than zero(0), so it uses less amount of memory.	+ operator always a creates a new string irrespective of length of string therefore it takes more memory.

```
String a = "1";
String b = "aa2aa";
String c = "123";

boolean isDigits = a.matches("\\d"); //T
boolean isDigits = b.matches("\\d"); //F
boolean isDigits = c.matches("\\d"); //F
```

```
c.matches("[0-9]+")
```

firstWord

```
String sentence = "Cyber Deal String, To Test";
String firstWord = sentence.replaceAll(" .*", "");

System.out.println(firstWord);
```

```
String sentence = "Cyber Deal String, To Test";
String firstWord = sentence.replaceAll(" ", "");
```

```
firstWord= sentence.substring(0, sentence.indexOf(" "));

firstWord = sentence.replaceAll(" .*", "");  //To delete the rest of the sentance
```

Is it a valid email?

- Format: name@domain.com
- Length should be 10 or more and less than 20
- No Spaces

```
String s1=“yaman@gmail.com";
String s2=“yaman.com”;
String s3=“Random Text”;
String s4=“yaman@ gmail.com; //CHALLENGE
```

Is it a valid email?

- Format: name@domain.com
- Length should be 10 or more and less than 20
- No Spaces

```
String s1=“yaman@gmail.com";
String s2=“yaman.com”;
String s3=“Random Text”;
String s4=“yaman@ gmail.com; //CHALLENGE

boolean test1 , test2, test3 , isValid;
test1 = s1.contains("@");
test2 = s1.contains(".com");
test3 = s1.length() <= 10;

isValid = test1 && test2 && test3;

        System.out.println(isValid );
```

```
String a = "This is a small string to search for the word Yaman, in all cases";
```

Should return true if we search for:

- Yaman
- yaman
- YamAn
- yAman
- YAMAN

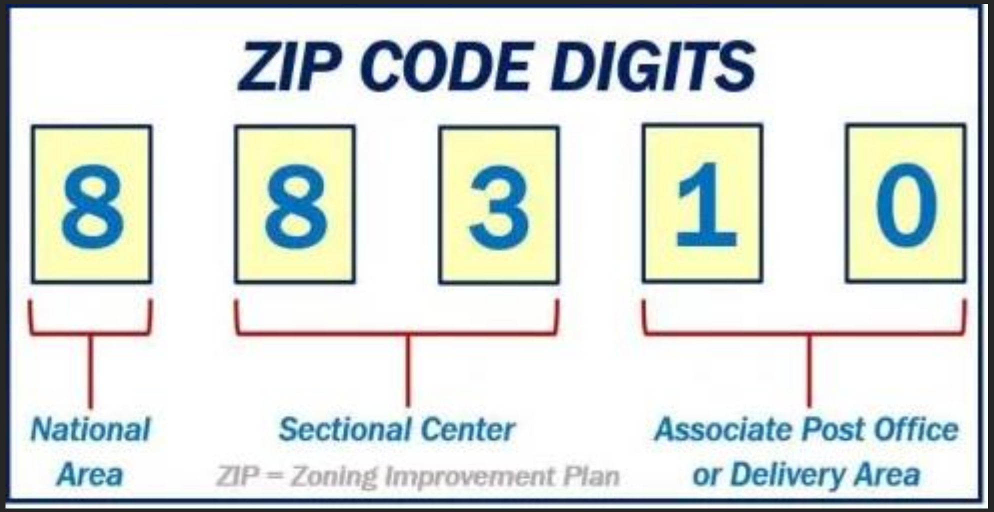
```
public static void main(String[] args) {

    String a = "This is a small string to search for the word Yaman, in all cases";
    Scanner input = new Scanner(System.in);
    String t = input.nextLine();
    boolean c = a.toLowerCase().contains(t.toLowerCase());
    System.out.println("RESULT " + c );

  }
```

INPUT: String a = "yaman“;
OUTPUT: a = “Yaman”;

```
String a = "yaman";
a =  a.substring(0, 1).toUpperCase() + a.substring(1);
System.out.println(a);
```


ZIP CODE DIGITS
8
8
3
1
0
National Area
Sectional Center
Associate Post Office or Delivery Area
ZIP = Zoning Improvement Plan

Save each part in a single string and the print each one on a single line

Print if it is a valid tweet, then print the length.

Rule #1 MAX=280 CHARACTERS

Rule #2 NOT ALLOWED WORDS = { “OIL , GAS” }

Mastercard 5434-4012-3454-7891

Mastercard 5434401234547891

Recived From Mastercard ****7891

```
String s = "5434-4012-3454-7891";
       String s2 = "";
        String s3 = "";

~~      //  s2 = s.replaceFirst("-", "");~~
~~       // s2 = s2.replaceFirst("-", "");~~
~~       // s2 = s2.replaceFirst("-", "");~~
               s2 = s.replace("-", "");  //or s.replaceAll
       s3 = "****" + s2.substring(s2.length()-4);

       System.out.println(s2);
System.out.println(s3);
```

CV BUILDER

This system will take little information from the user
and will provide him with a fully developed CV in return

A recent graduate with a degree in ---- from ---- with a GPA of %% seeking an entry-level position in your company. I have a clear, logical mind with a practical approach to problem-solving and a drive to see things through to completion. I have % years of experience in ----- . I am eager to learn, I enjoy overcoming challenges.

```
Scanner input = new Scanner(System.in);
System.out.println("ENTER YOUR NAME");
String a = input.nextLine();
String base = "My name is " + a;
System.out.println("ENTER YOUR CITY");
a = input.nextLine();
base = base + ", I live in " + a;
System.out.println("ENTER YOUR AGE");
a = input.nextLine();
base = base + ", and I am  " + a + " years old.";
System.out.println(base);
```

Travel: travel, vacation, holiday, hotels, airport, landmarks, world, Europe, flights

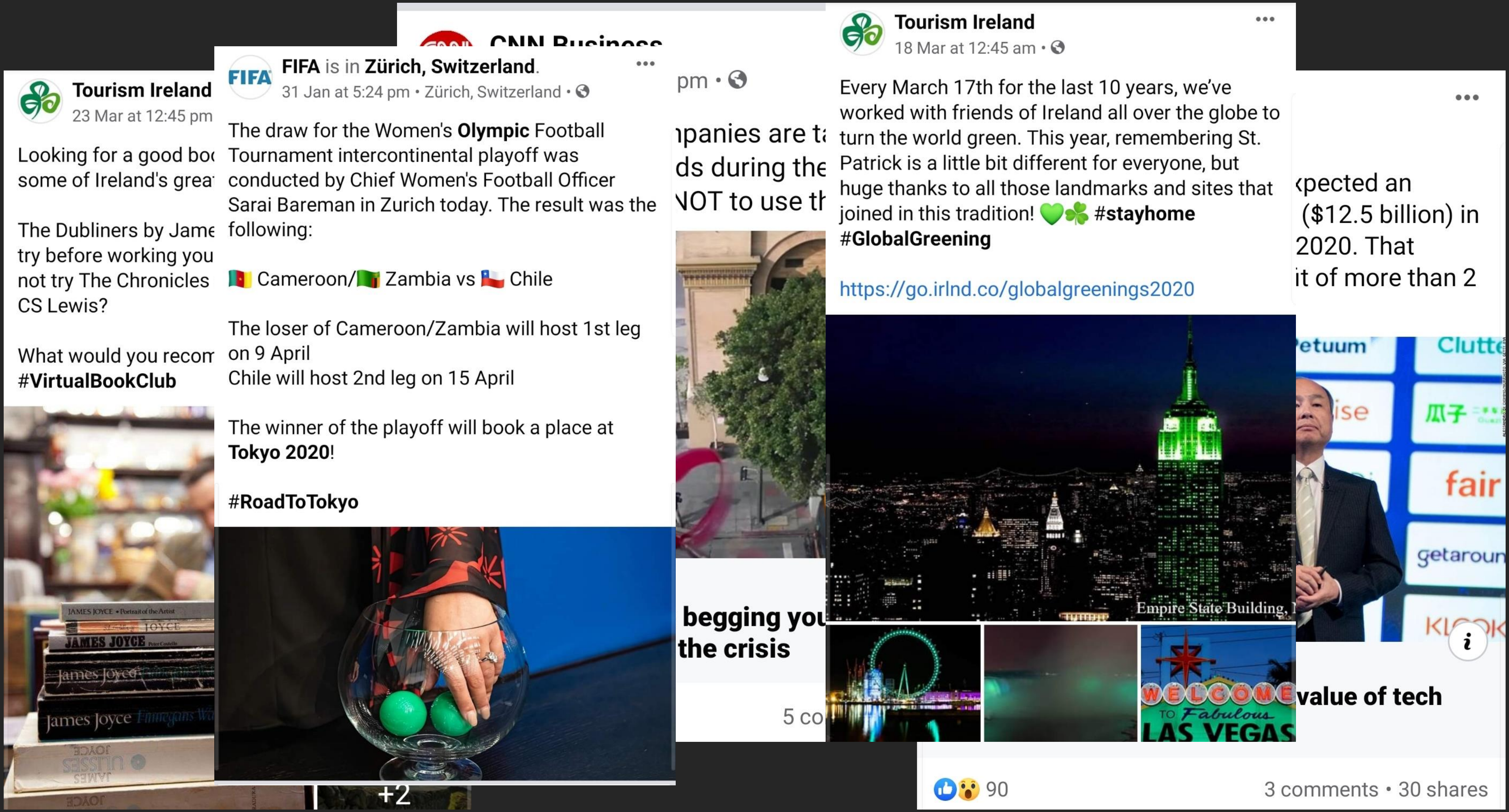
Sports: football, game, match, tennis, swimming, Rafael Nadal, boxing, baseball

Business: company, product, management, manufacturing, business, planning, profit

Question: How, What, Where, ?

```
boolean keyWordCar = post.contains(“mercedes") || post.contains(“bmw") || post.contains(“audi") ;

boolean keyWordApp = post.contains(“android") || post.contains(“ios") || post.contains(“app") ;
```



Tourism Ireland
23 Mar at 12:45 pm

Looking for a good boo
some of Ireland's grea

The Dubliners by Jame
try before working you
not try The Chronicles
CS Lewis?

What would you recom
#VirtualBookClub

FIFA is in **Zürich, Switzerland**.
31 Jan at 5:24 pm • Zürich, Switzerland •

The draw for the Women's **Olympic** Football Tournament intercontinental playoff was conducted by Chief Women's Football Officer Sarai Bareman in Zurich today. The result was the following:

🇨🇲 Cameroon/🇿🇲 Zambia vs 🇨🇱 Chile

The loser of Cameroon/Zambia will host 1st leg on 9 April
Chile will host 2nd leg on 15 April

The winner of the playoff will book a place at **Tokyo 2020**!

#RoadToTokyo

CNN Business
pm •

npanies are ta
ds during the
NOT to use th

begging you
the crisis

5 co

Tourism Ireland
18 Mar at 12:45 am •

Every March 17th for the last 10 years, we've worked with friends of Ireland all over the globe to turn the world green. This year, remembering St. Patrick is a little bit different for everyone, but huge thanks to all those landmarks and sites that joined in this tradition! 💚☘️ **#stayhome** **#GlobalGreening**

https://go.irlnd.co/globalgreenings2020

xpected an
($12.5 billion) in
2020. That
it of more than 2

value of tech

👍😮 90

3 comments • 30 shares

Every March 17th for the last 10 years, we've worked with friends of Ireland all over the globe to turn the world green. This year, remembering St. Patrick is a little bit different for everyone, but huge thanks to all those landmarks and sites that joined in this tradition! #stayhome #GlobalGreening

Looking for a good book? How about these from some of Ireland's great writers:

The Dubliners by James Joyce is a good one to try before working your way up to Ulysses! Or why not try The Chronicles of Narnia by Belfast born CS Lewis?

What would you recommend? #StayHome #VirtualBookClub

The draw for the Women's Olympic Football Tournament intercontinental playoff was conducted by Chief Women's Football Officer Sarai Bareman in Zurich today. The result was the following:

R Cameroon/Ri Zambia vs lu Chile
The loser of Cameroon/Zambia will host I st leg on 9 April
Chile will host 2nd leg on 15 April

The winner of the playoff will book a place at Tokyo 20201

#RoadToTokyo

At least two large companies are taking an unusual tack in their ads during the coronavirus: they're urging people NOT to use their products

The Japanese company said it expected an operating loss of 1 .35 trillion yen ($12.5 billion) in its fiscal year through March 31 , 2020. That compares with an operating profit of more than 2 trillion yen the previous year.

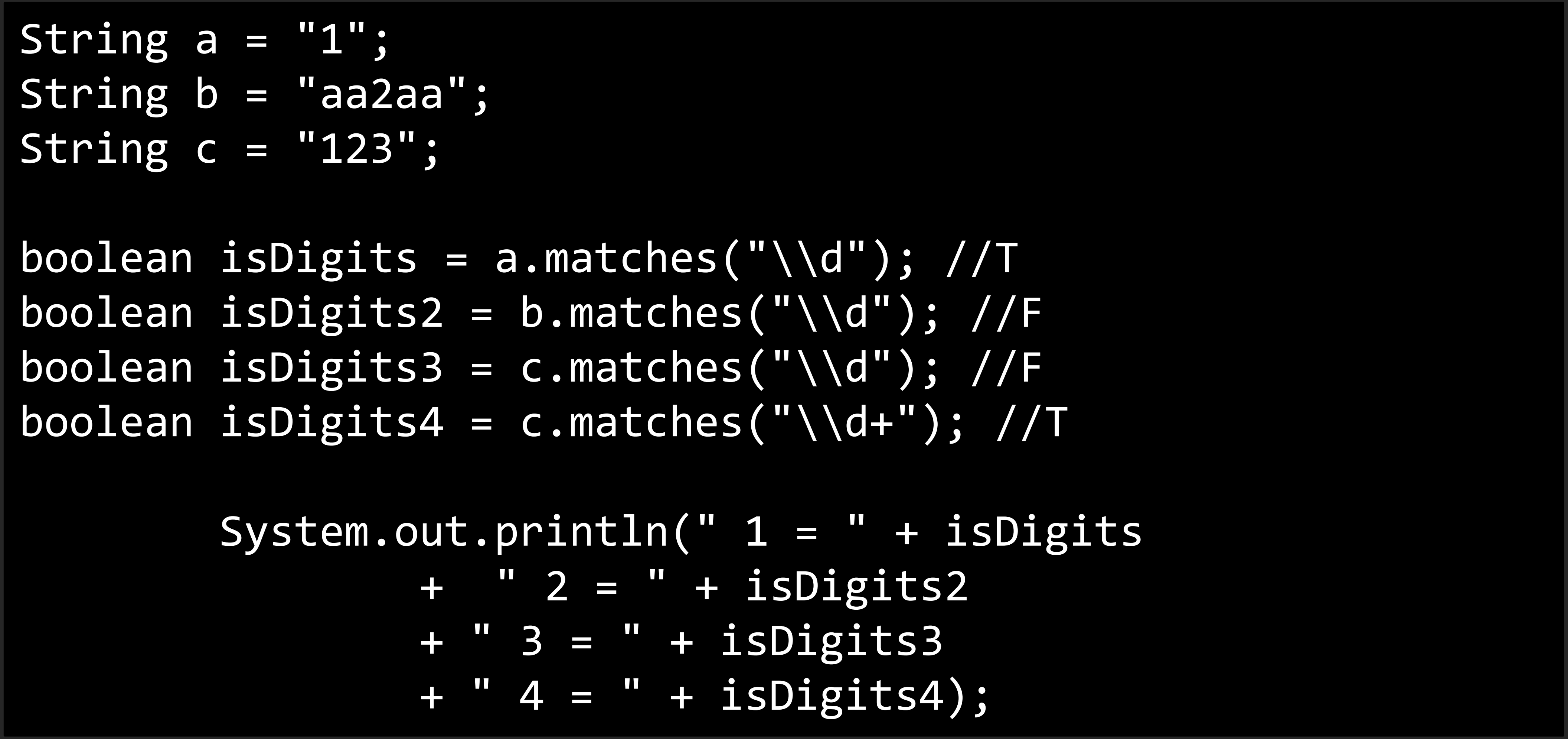
Casting Problem

```
String a = "1";
String b = "aa2aa";
String c = "123";



System.out.println("A = " + Integer.parseInt(a));
System.out.println("B = " + Integer.parseInt(b));
System.out.println("C = " + Integer.parseInt(c));
```

- Fix this problem (casting a string)
- Replace and non digit with a “”



```
String a = "1";
String b = "aa2aa";
String c = "123";

boolean isDigits = a.matches("\\d"); //T
boolean isDigits2 = b.matches("\\d"); //F
boolean isDigits3 = c.matches("\\d"); //F
boolean isDigits4 = c.matches("\\d+"); //T

        System.out.println(" 1 = " + isDigits
                +  " 2 = " + isDigits2
                + " 3 = " + isDigits3
                + " 4 = " + isDigits4);
```

Convert a String of numbers to an integer
(but be carful if there is anything other than digits)

```
String a = "1";
String b = "aa2aa";
String c = "123";

// The [^0-9] expression is used to find any character that is NOT a digit.

b  = b.replaceAll("[^0-9]", "");
System.out.println("b = " + b); // b = 2
System.out.println("b = " + Integer.parseInt(b));
```

Convert a String of numbers (with a floating point) to an integer (but be carful if there is anything other than digits)

```
String a = “22.44";

if (b.contains(".")) {
       a = a.substring(0, a.indexOf("."));
}
       a  = a.replaceAll("[^0-9]", "");

System.out.println("b = " + Integer.parseInt(b));
```

REGEX

A regular expression is a sequence of characters that forms a search pattern. When you search for data in a text, you can use this search pattern to describe what you are searching for.

A regular expression can be a single character, or a more complicated pattern.

Regular expressions can be used to perform all types of text search and text replace operations.

```
System.out.println("-231232213".matches("^\\+[1-9]\\d{1,14}$"));
System.out.println("+231232213".matches("^\\+[1-9]\\d{1,14}$"));
System.out.println("+2A31232213".matches("^\\+[1-9]\\d{1,14}$"));
System.out.println(Pattern.matches(".s", "as"));//true (2nd char is s)
System.out.println(Pattern.matches(".s", "mk"));//false (2nd char is not s)
System.out.println(Pattern.matches(".s", "mst"));//false (has more than 2 char)
System.out.println(Pattern.matches(".s", "amms"));//false (has more than 2 char)
System.out.println(Pattern.matches("..s", "mas"));//true (3rd char is s)
System.out.println("? quantifier ....");
System.out.println(Pattern.matches("[amn]?", "a"));//true (a or m or n comes one time)
System.out.println(Pattern.matches("[amn]?", "aaa"));//false (a comes more than one time)
System.out.println(Pattern.matches("[amn]?", "aazzta"));//false (a comes more than one time)
System.out.println(Pattern.matches("[amn]?", "am"));//false (a or m or n must come one time)

System.out.println("+ quantifier ....");
System.out.println(Pattern.matches("[amn]+", "a"));//true (a or m or n once or more times)
System.out.println(Pattern.matches("[amn]+", "aaa"));//true (a comes more than one time)
System.out.println(Pattern.matches("[amn]+", "aammmnn"));//true (a or m or n comes more than
once)
System.out.println(Pattern.matches("[amn]+", "aazzta"));//false (z and t are not matching pattern)
System.out.println(Pattern.matches("\\d", "abc"));//false (non-digit)
System.out.println(Pattern.matches("\\d", "1"));//true (digit and comes once)
System.out.println(Pattern.matches("\\d", "4443"));//false (digit but comes more than once)
System.out.println(Pattern.matches("\\d", "323abc"));//false (digit and char)
```

/*Create a regular expression that accepts alphanumeric characters only.
Its length must be six characters long only.*/

/*Create a regular expression that accepts 10 digit numeric characters starting with 7, 8 or 9 only.*/