Java Programing

Introduction to Java

What is Java?

- Java technology is both a programming language and a **platform!**

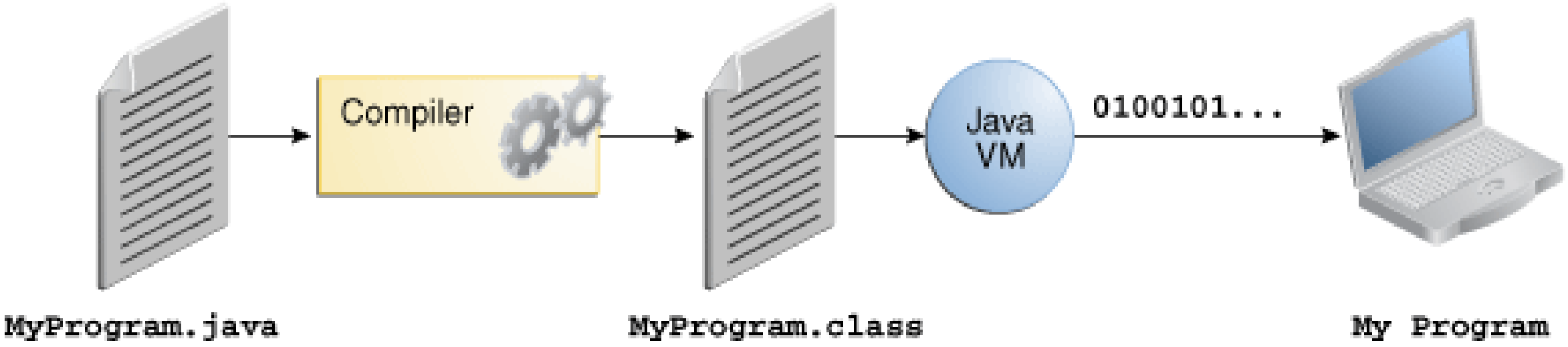
- The Java programming language is a high-level language

- Some if its strong points:
  - Simple
  - Object oriented
  - Multithreaded
  - Architecture neutral
  - Portable

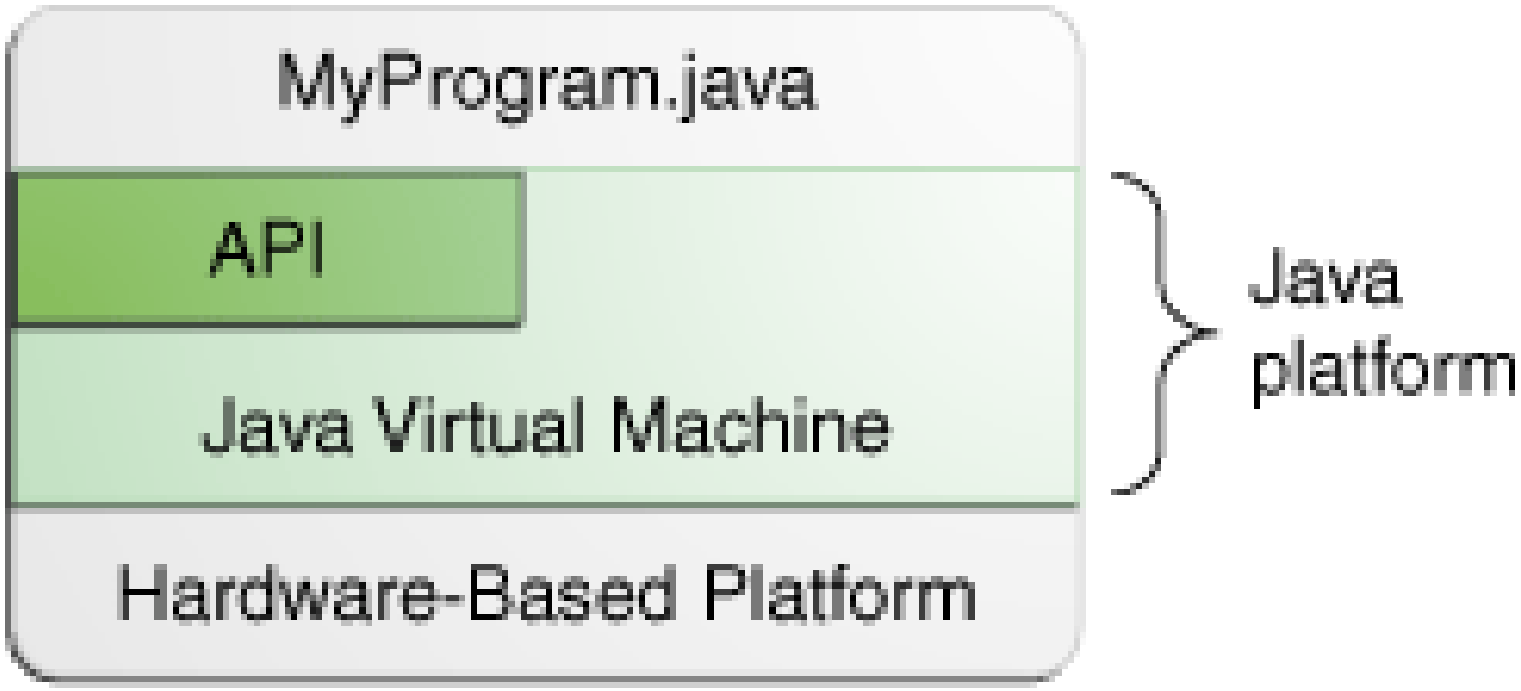
What can we do with Java platform ?

- Create web applications
- Create web services
- Create desktop applications
- Create mobile applications
- Create software for Smart TVs, STBs
- Many other applications

How is a Java application created?



Why call Java a platform?



- Note: Here the term ‘API’ can be rephrased to ‘libraries’

Installing the tools

What tools do we need to learn Java?

- To get started with core Java programing, we will need:
  - Java Development Kit (JDK)
  - Integrated Development Environment (IDE)

Note:

There are several ‘flavours’ of JDK available today, but we would stick to the good old **Oracle’s** JDK!

Again, we have many options when it comes to picking up and IDE, like Eclipse, NetBeans and more modern ones like IDEA. We will use **IDEA**.

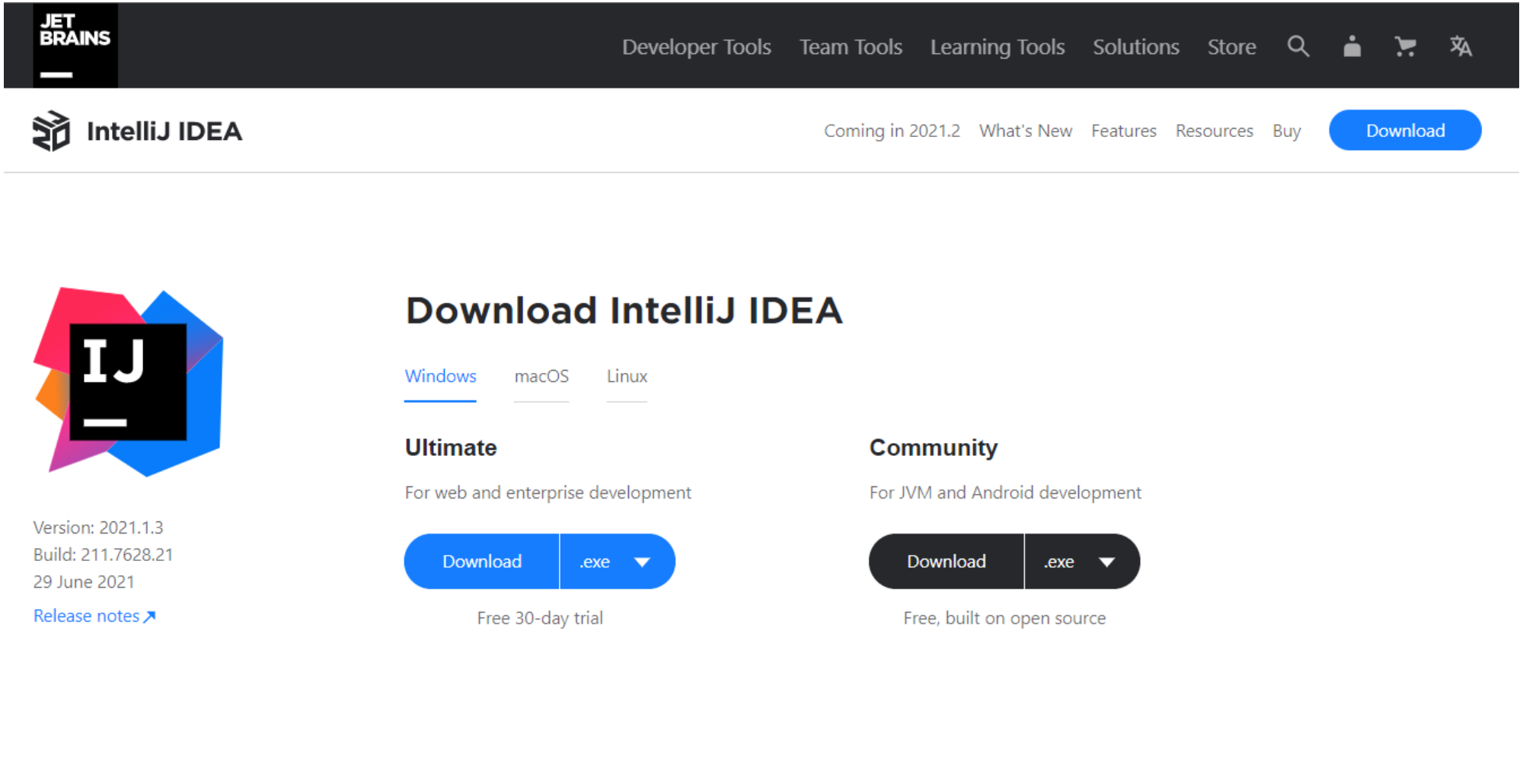
What is JDK exactly?

- JDK (Java Development Kit)
  - Java Compiler
  - JRE (Java Runtime Environment)
    - JVM (Java Virtual Machine)
    - Other Libraries required by JRE

Installing IntelliJ IDEA

Download and install community edition

https://www.jetbrains.com/idea/download/#section=windows



Getting started with Java tools

A quick overview of IntelliJ IDEA

- **Help** menu: About (Version)
- Main Settings : File -> Settings
- **Project** panel: Hide/Move
- Project root: Open in -> Explorer
- Project structure: Different folders and files
- Code editor: Main.Java
- **Run** menu: To run the project (Other options)
- **Run** panel: To see output (can move around)

A handy tool - JShell

- JShell is an REPL tool (Read – Evaluate – Print-Loop)
- Useful for beginners to get a quick hold of language syntax
- Does not need a class or method to run
- Interactive way to evaluate functions
- Can be used by senior programmers as well
- Can be used on a terminal without IDE as well

Accessing JShell from IDEA

- Go to **Terminal** panel (if not shown, use Alt+F12 to open)
- Type JShell and press enter
- We can type any expression or function
- use /exit to quit

Java language basics

What is an expression?

- An *expression* is a construct made up of variables/literals, operators, and method invocations that evaluates to a single value

- 5 * 2
- 5 * 2 + 2
- (5 * 2) + 2 // more clear with parenthesis
- 5 * (2 + 2) // BODMAS applicable in Java as well

What is a variable?

- A variable is a name give to a memory location
- The programmer can keep varying (assigning) it’s value
- Declaration and assignment can be done separately
  - int x
  - x = 5
- Or together
  - int x = 5
- Value can be changed any time
  - x = 10

What are the rules to name a variable?

- Variable names are case-sensitive
- It cannot begin with number
- It cannot be a keyword
- It must not contain spaces
- It can use underscore or dollar symbol

Variable naming best practices

- It is recommended to begin with a letter and not _ or $
- Use full words instead of short forms
- Camel case is a common standard (amountDue)
- Pascal case also exists (AmountDue)
- So does snake case (amount_due)
- All uppercase is also not recommended for variables

What is a primitive datatype?

Primitive types are special data types built into the language; they are not objects created from a class

Primitive Data Types

1. byte
2. short
3. int
4. long
5. float
6. double
7. Boolean
8. char

Non primitive datatypes

1. String
2. Array
3. Others

Primitive datatypes sizes

Data Type	Size in bytes	Range
byte	1	-128 to 127
short	2	-32,768 to 32,767
int	4	-2,147,483,648 to 2,147,483,647
long	8	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
float	4	precision of 7 decimal digits
double	8	precision of 15 decimal digits
boolean	Not defined	true or false
char	2	for single unicode character

Primitive datatypes defaults

DATA TYPE	DEFAULT VALUE (FOR FIELDS)
byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0d
char	'\u0000'
String	null
boolean	false

A heads-up on default values

The compiler never assigns a default value to an uninitialized **local** variable, Accessing an uninitialized local variable will result in a compile-time error!

Problem

Print the area of a triangle, given that it's base is 2 units and height is 1.5 units

- Solve it in JShell
- First use variables of double datatype
- Then redo the problem with variables of float datatype

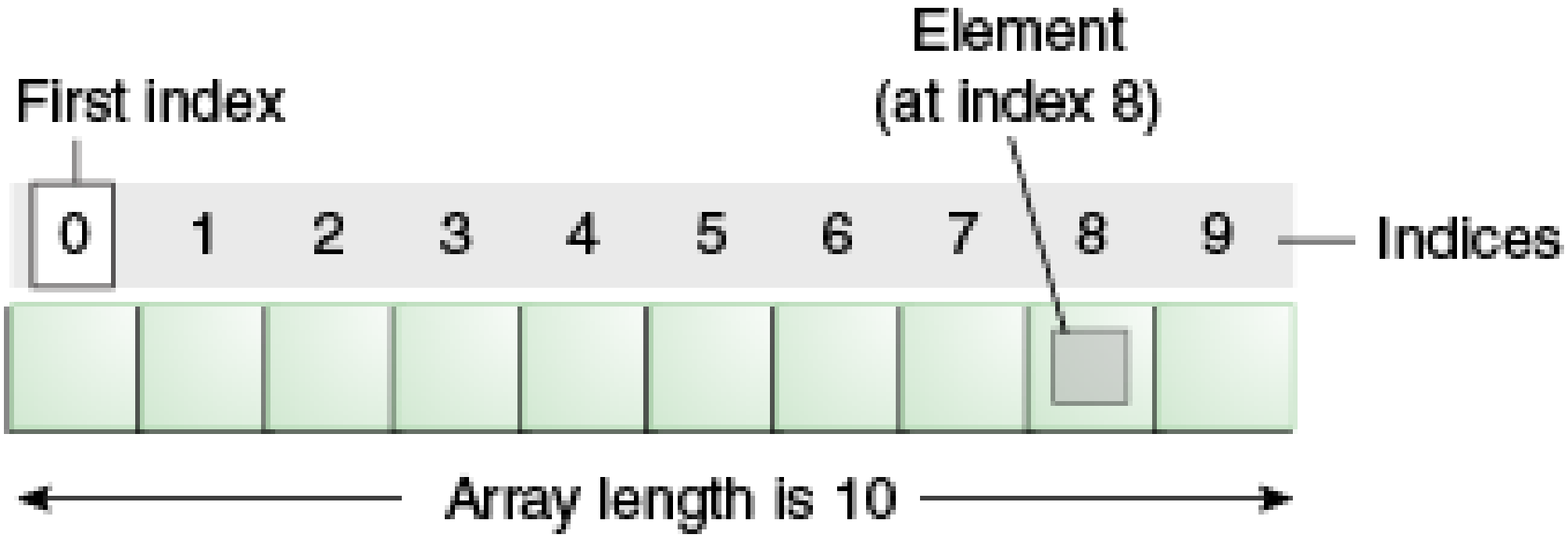
Note: You will be in for some surprises!

Learnings from the problem

- Integer divisions resulted in integer result
- Use *f* suffix for floating type literals
- Assigning larger type to a smaller one results in error
- A combination of variable, literals and operators resulted in one line of code which we called **statement**
- A statement forms a complete unit of execution

What is an array?

- An *array* is a container (object) that holds a fixed number of values of a **single type**



How to create an array?

- The length of an array is established when the array is created and its length is fixed after that
- int[] marks (Declaration)
- marks = new int[6] (Construction)
- int[] marks = new int[6] (usually both are combined)
- Same concept holds good for other datatypes as well

How to initialize and access array items

- int[] marks = {60,70,75,80,90} (initialize)
- marks.length (prints 5)
- marks[0] (access first element)
- marks[4] (access last element)
- marks[5] (error)

Good news!

Let's start using IntelliJ IDEA!

We may still utilize JShell when required!

A heads up on control flow statements

- The statements inside a program is executed from top to bottom, in the order that they appear
- **Control flow statements** can break up this flow of execution by using looping, decision making, and branching
- This allows the developer to **conditionally** execute a particular piece of code
- If-else, for loop, while loop, switch, and return, break, continue are the control flow statements available in Java

Problem

- Give below are the number of people in four families
- Display which family has even and which has odd number of people

Family No	No of people
1	2
2	3
3	4
4	1

Learnings from the problem

1. Nested Expressions
2. For loop basics
3. Operators ++, %, ==
4. If-else condition
5. First program using IntelliJ Idea

Problem

- Given the scores (out of 100) of a student in five of his subjects shown
  - Print all his scores
  - Find out in how many subjects he has scored 70 and above
  - Find out if he has scored full in any subject

Subject	Marks
Physics	75
Chemistry	70
Math	100
Biology	65
English	85

Learnings from the problem

- Used boolean variable
- Used ++ operator
- Learnt IDE feature of converting foreach to normal for loop
- If condition without else and without braces

What is a String exactly?

- A String is a non-primitive datatype in Java
- It is used to store text data
- Internally, it is an *object* (more on it later)
- A string literal has to be enclosed in double quotes
- String fruit = “Apple” (Declaration and initialization)
- Java has several inbuilt methods for strings

How to use special characters in strings?

- Since string literals must be enclosed in quotes, compiler would have difficulties in dealing with certain situations.
- We use **escape** character **** to deal with it
- System.out.println(" "Java" is a language") (error)
- System.out.println(" \"Java\" is a language") (no error)

Useful string operations

- Strings can be joined using **+** operator or **concat()**
- length() gives us the number of characters in the string
- toUpperCase() and toLowerCase()
- equals() and isEmpty()

Note: An empty string is also a blank string, but not vice versa.

Problem

Given the name of a country as “India” find if it contains the letter ‘d’ in it

Learnings from the problem

- using char literal
- use of charAt() method of a string
- How IntelliJ Idea prompts us with method details
- Use of 'break' keyword inside for-loop

Problem – Part A

- Given a person's email account, display a welcome message to him
  - Let us use '**hard-coded**' email id for now
  - Let us use simple concatenation for displaying the welcome message

Learnings from the problem

- What is the meaning of Hardcoded
- Use of indexOf() method of a string
- Use of substring() method of a string

Problem – Part B

- When a user enters his email account, display a welcome message to him
  - Let us show the **date** and time the user logged in
  - Lets use **format** the displaying message in better way

Learnings from the problem

- Packages can be imported into a Java application
- IDEA will prompt us to import packages
- How to display today’s date
- How to format strings using MessageFormat.format()

What are methods?

- A method is a named block of code which can be **reused** any number of times in an application
- In software development, there is a golden rule called **DRY** (Don’t Repeat Yourself) Repeated code is a major source of bugs and maintenance nightmare!
- Methods are also used to split big chunk of codes to smaller and manageable slices of functionality, hence called functions

Defining a simple method

- The only required elements of a method declaration are the method's return type, name, a pair of parentheses, (), and a body between braces, {}

```
void printSeparator(){
    System.out.println(".........................");
}
```

Six components of method declaration

1. Modifiers—such as public, private, and others you will learn about later.
2. The return type—the data type of the value returned by the method, or void
3. The **method name**—the rules for field names apply to method names as well
4. The **parameter list** in parenthesis—a comma-delimited list of input parameters, preceded by their data types, enclosed by parentheses, (). If there are no parameters, you must use empty parentheses.
5. An exception list—to be discussed later.
6. The method body, enclosed between braces—the method's code

Method Signature

- Two of the components of a method declaration comprise the *method signature*—the method's **name** and the parameter **types**
- Example:
  - calculateAnswer(double, int, double, double)

Problem

- Create a login functionality as per the requirements given below:
  - Display a separator at the beginning and end of the screen
  - Ask the user to enter his username and then password
  - In case user’s inputs are not valid, ask for inputs again
  - In case the entries are valid, show the result of the login
  - Expected username is *sam* and password is *1234*

Learnings from the problem

- Defining and using methods
- Reading user input from console
- Use of do-while loop
- Use of logical operators !, && , ||
- Use of equals() method to compare two strings

Problem

- Given a month as a number, output the corresponding month in words.
  - If input is 3, output must be *March*
  - If input is 4, output must be *April*
  - If input is 13, output must be *Invalid*

Learnings from the problem

- Converting String to int
- Using switch statement
- Extracting method for making the code readable
- Using newer and more readable format of switch statement