

# Understanding Data Datatypes and Information

This presentation provides an introduction to data and information in Java. We will explore data types and their differences, learn about RAM and hard disk, and discover primitive and non-primitive data types.

```
<title>HTML Tutorial</title>  
<meta name="description" content="HTML tutorial">  
<meta name="author" content="Andrew">  
<meta name="copyright" content="2000-2011 and beyond...">  
<meta name="robots" content="all">  
<meta name="viewport" content="width=780">  
<base target="_top">  
<style type="text/css" media="all">@import "/us.css";</style>  
<link rel="stylesheet" type="text/css" href="/print.css" media="print">  
<link rel="shortcut icon" type="image/ico" href="/favicon.ico">  
<link rel="search" type="application/opensearch" title="HTML So  
htmlsource-search.xml">  
<script>  
</script>  
<script src="/scripts.js" type="text/javascript"></script>  
<style type="text/css">  
<!--
```

# Overall

**01**

**What is data & information**

**02**

**Types of data**

**03**

**Data types in  
java**

**04**

**Practical exercise**

# Introduction

**Review last  
lesson**

**In Java, to print text or variables to the console, you typically use the `System.out.println()` method. This method prints the text or the value of the variable and then inserts a newline**



Saifullah "Haidari"

# How can you use ?

Review last  
lesson

```
System.out.println("Hello World!");
```

**You can add as many println() methods as you want. Note that it will add a new line for each method:**

```
System.out.println("Hello World!");  
System.out.println("I am learning Java.");  
System.out.println("It is awesome!");
```

Saifullah  
"Haidari"



by Saifullah Haidari

# Using Quotes?

Review last  
lesson

## Double Quotes

When you are working with text, it must be wrapped inside double quotations marks "".

If you forget the double quotes, an error occurs:

```
System.out.println("This sentence will work!");  
System.out.println(This sentence will produce an error);
```



# Differences

**Review last  
lesson**

There is also a `print()` method, which is similar to `println()`.

The only difference is that it does not insert a new line at the end of the output

```
System.out.print("Hello World! ");  
System.out.print("I will print on the same  
line.");
```

# Print Numbers

**Review last  
lesson**

## numbers

You can also use the `println()` method to print numbers.  
However, unlike text, we don't put numbers inside double quotes:

## Examples

```
System.out.println(3);  
System.out.println(358);  
System.out.println(50000);
```







# Hello World in Java

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

The "Hello World" program demonstrates the basic structure of a Java program, including the "main" method, which is the entry point for execution.



by Saifullah Haidari

## Review last lesson

```
var perc = 99.0, wmin = 1920, hmin = 1080, w, h, w1, h1, ratio;  
var FromDoc = open ( File ("D:\FromMacro.psd"));  
var IntoDoc = open ( File ("D:\IntoMacro.psd"));  
  
app.preferences.rulerUnits = Units.PIXELS;  
w = FromDoc.width.value;  
h = FromDoc.height.value;  
ratio = h/w;  
app.activeDocument = FromDoc;  
activeDocument.activeLayer = activeDocument.layers[0];  
var shapeRef =  
[ [ Math.floor ((w-1920)/2), Math.floor ((h-1080)/2) ],  
  [ Math.floor ((w-1920)/2)+1920, Math.floor ((h-1080)/2) ],  
  [ Math.floor ((w-1920)/2)+1920, Math.floor ((h-1080)/2)+1080 ],  
  [ Math.floor ((w-1920)/2), Math.floor ((h-1080)/2)+1080 ] ];  
app.activeDocument.selection.select ( shapeRef, SelectionType.REPLACE );  
app.activeDocument.selection.copy ();  
app.activeDocument = IntoDoc;  
activeDocument.activeLayer = activeDocument.layers[0];  
IntoDoc.paste ();  
  
while (1) {  
    if ( (w < wmin) || (h < hmin) ) break;  
    app.activeDocument = FromDoc;  
    activeDocument.activeLayer = activeDocument.layers[0];  
  
    app.activeDocument.activeLayer.copy ();  
    app.activeDocument = betweenDoc;  
    betweenDoc.paste ();  
    w1 = w;  
    h1 = h;  
    w = w * perc / 100;  
    h = w * ratio;  
}
```

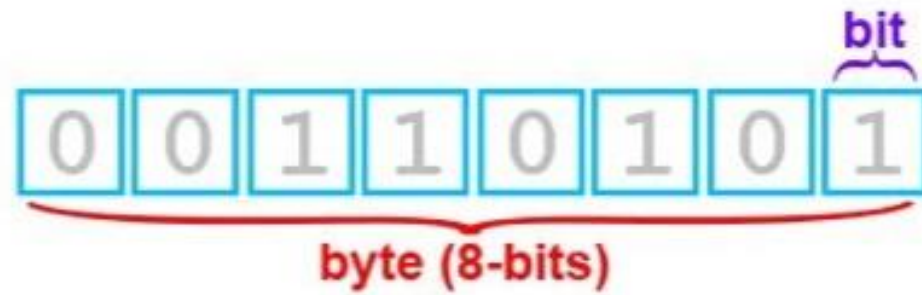
Image ID: 2BGAXKE  
www.alamy.com



# Common data storage measurements

Common Data Storage Measurements	
UNIT	VALUE
bit	1 bit
byte	8 bits
kilobyte	1,024 bytes
megabyte	1,024 kilobytes
gigabyte	1,024 megabytes
terabyte	1,024 gigabytes
petabyte	1,024 terabytes

# Bit and byte



# RAM vs. Hard Disk

## RAM

**RAM is temporary, volatile memory. It's like a workspace where the computer actively stores data and instructions while you're working.**

## Hard Disk

**The hard disk is permanent, non-volatile memory. It's like a library where the computer stores files and data for long-term access.**

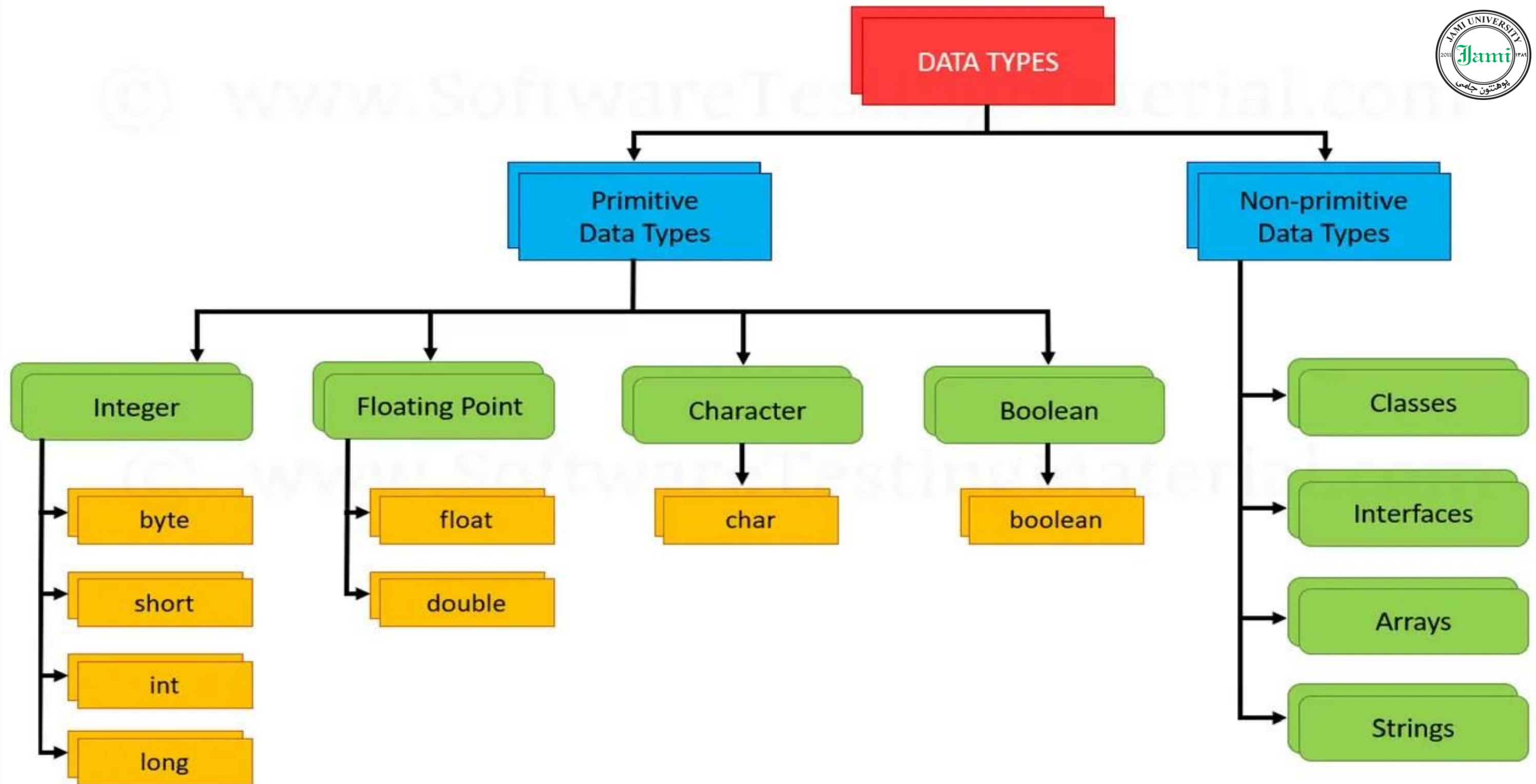
# What is Data? What is Information?

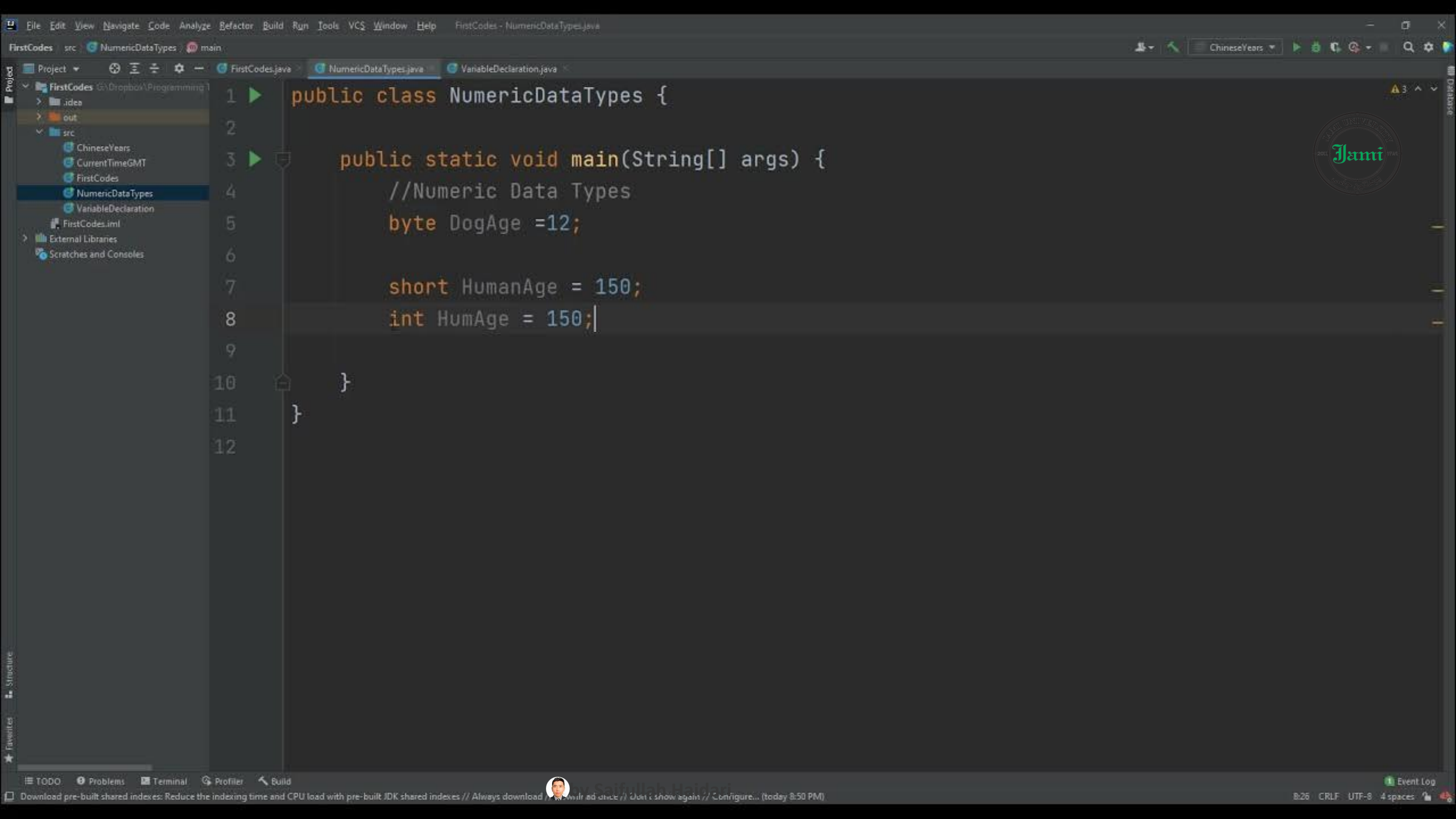
## Data

**Data is raw, unprocessed facts and figures, lacking context or meaning. It's like a collection of ingredients before they are assembled into a dish.**

## Information

**Information is processed data that provides context and meaning. It's like a recipe that guides you to create a meal from raw ingredients.**







Type	Size (in bits)	Range
byte	8	-128 to 127
short	16	-32,768 to 32,767
int	32	$-2^{31}$ to $2^{31}-1$
long	64	$-2^{63}$ to $2^{63}-1$
float	32	1.4e-045 to 3.4e+038
double	64	4.9e-324 to 1.8e+308
char	16	0 to 65,535
boolean	1	true or false



# Data Types

Data Type	Size	Description
byte	1 byte	Stores whole numbers from -128 to 127
short	2 bytes	Stores whole numbers from -32,768 to 32,767
int	4 bytes	Stores whole numbers from -2,147,483,648 to 2,147,483,647
long	8 bytes	Stores whole numbers from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
float	4 bytes	Stores fractional numbers. Sufficient for storing 6 to 7 decimal digits
double	8 bytes	Stores fractional numbers. Sufficient for storing 15 decimal digits
boolean	1 bit	Stores true or false values
char	2 bytes	Stores a single character/letter or ASCII values

## Example

```

int myNum = 5;           // Integer (whole number)
float myFloatNum = 5.99f; // Floating point number
char myLetter = 'D';     // Character
boolean myBool = true;   // Boolean
String myText = "Hello"; // String

```




# Data Types

Data Type	Size (bytes)	Usage
int	4	Whole numbers
long	8	Larger whole numbers
float	4	Single-precision floating point numbers
double	8	Double-precision floating point numbers
char	2	Single 16-bit Unicode character
String	Variable	Sequence of characters
boolean	1	True or false values



by Saifullah Haidari



CORE JAVA CHEATSHEET

Learn JAVA from experts at [www.edureka.co](http://www.edureka.co)

**Java Programming**

Java is a high level, general purpose programming language that produces software for multiple platforms. It was developed by James Gosling in 1991 and released by Sun Microsystems in 1996 and is currently owned by Oracle.

**Primitive Data Types**

Type	Size	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	1.4e-38 to 3.4e38
double	8	1.7e-308 to 1.7e308
char	2	Unicode Character Set
Boolean	1	True, False

**Java Operators**

Type	Operators
Arithmetic	+, -, *, /, %, ++, --
Assignment	=, +=, -=, *=, /=, %+, %=, &+, &-=
Relational	<, >, <=, >=, ==, !=
Logical	&,   , !
Bitwise	&,  , ^, ~, <<, >>
Unary	+, -, ++, --, ~, !

**Java Variables**

```
(public|private|static) type name [= expression][value];
```

**Java Methods**

```
(public|private|static) (type | void) name(arg1, ..., argN) {statements}
```

**Data Type Conversion**

```
// Widening (byte to short to int to long to float to double)
int i = 10; //int to long
long l = i; //Automatic type conversion
// Narrowing
double d = 10.0;
long j = (long)d; //Explicit type casting
// Numeric values to String
String str = String.valueOf(value);
// String to Numeric values
int i = Integer.parseInt(str);
double d = Double.parseDouble(str);
```

**User Input**

```
// Using BufferedReader
BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
String name = reader.readLine();
// Using Scanner
Scanner in = new Scanner(System.in);
String s = in.nextLine();
int a = in.nextInt();
// Using Console
String name = System.console().readLine();
```

**Iterative Statements**

```
// for loop
for (condition) (expression)
// do-while loop
for (int i = 0; i < arr.length; i++) {
    // do something
}
// while loop
while (condition) {expression}
// do-while loop
do {expression} while (condition);
```

**Fibonacci series**

```
for (i = 1; i <= n; i++) {
    System.out.print(i + " ");
    int sum = t1 + t2; t1 = t2; t2 = sum;
}
```

**Pyramid Pattern**

```
n = 2^n - 1;
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n-i; j++) {
        System.out.print(" ");
    }
    for (int k = 0; k < i; k++) {
        System.out.print("1");
    }
    System.out.println();
}
```

**Decisive Statements**

```
//if statement
if (condition) (expression)
//if-else statement
if (condition) (expression) else (expression)
//switch statement
switch (var) { case 1: expression; break; default: expression; break; }
```

**Prime Number**

```
if (n < 2)
    return false;
for (int i = 2; i <= n/i; i++)
    if (n % i == 0) return false;
return true;
```

**Factorial of a Number**

```
int factorial(int n) {
    if (n == 0)
        return 1;
    else
        return n * factorial(n-1);
}
```

**Arrays in Java**

**1 - Dimensional**

```
// Initializing
type[] varName = new type[size];
// Declaring
type[] varName; varName = new type[size];
```

**Array with Random Variables**

```
double[] arr = new double[n];
for (int i = 0; i < n; i++) {
    arr[i] = Math.random();
}
```

**Maximum value in an Array**

```
double max = 0;
for (int i = 0; i < arr.length; i++) {
    if (arr[i] > max) max = arr[i];
}
```

**Reversing an Array**

```
for (int i = 0; i < arr.length/2; i++) {
    double temp = arr[i];
    arr[i] = arr[arr.length-1-i];
    arr[arr.length-1-i] = temp;
}
```

**Multi - Dimensional Arrays**

```
// Initializing
datatype[][] varName = new datatype[row][col];
// Declaring
datatype[][] varName = {{value1, value2,...},{value1, value2,...}};
```

**Transposing A Matrix**

```
for (i = 0; i < row; i++) {
    for (j = 0; j < col; j++) {
        System.out.print(arr[j][i] + " ");
    }
    System.out.println();
}
```

**Multiplying Two Matrices**

```
for (i = 0; i < row; i++) {
    for (j = 0; j < col; j++) {
        sum = 0;
        for (k = 0; k < row2; k++) {
            sum += arr[i][k] * arr2[k][j];
        }
        arr[i][j] = sum;
    }
}
```

**Java Strings**

```
// Creating String using literal
String str1 = "Welcome";
// Creating String using new keyword
String str2 = new String("Welcome");
```

**String Methods**

```
str1+str2 //concatenates the strings
String indexOf() //returns the index of the first occurrence of the specified character
String length() //returns the length of the string
String charAt() //returns the character at the specified index
String toUpperCase() //returns the string in all uppercase
String toLowerCase() //returns the string in all lowercase
String replace() //replaces the specified character with the specified character
String trim() //removes the leading and trailing spaces
String contains() //checks for the existence of the specified character in the string
String startsWith() //checks for the existence of the specified prefix in the string
String endsWith() //checks for the existence of the specified suffix in the string
```

**Basic Java Program**

```
public class Demo {
    public static void main(String[] args) {
        System.out.println("Hello from edureka!");
    }
}
```

**Run** **Compile** **Execute**

classDemo.java  
javac classDemo.java  
java classDemo

INTERER

STRING

FLANN

# Primitive Data Types

## Definition

Primitive data types are fundamental building blocks in Java. They are predefined and provide basic storage for different types of data.

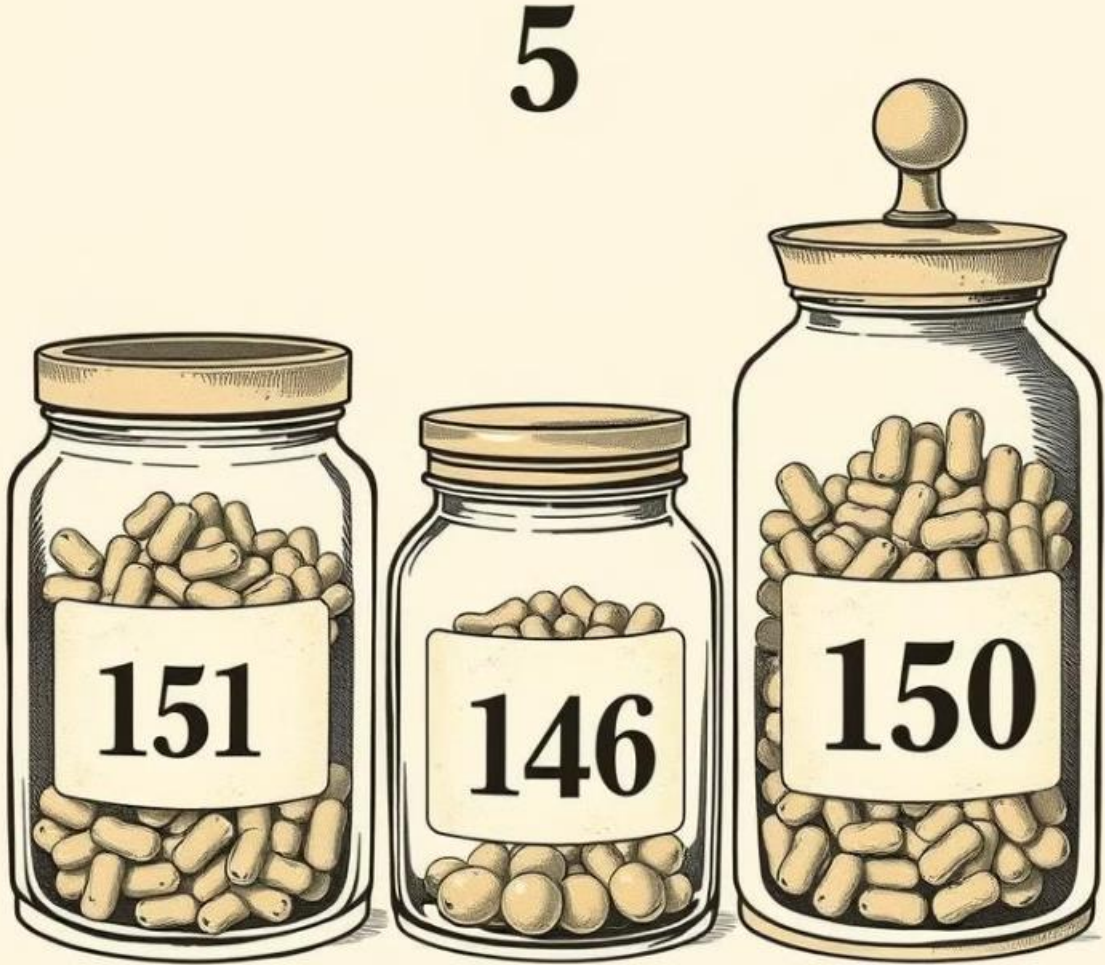
1



by Saifullah Haidari

# Numeric Data Types

Type	Size	Example
Byte	1 Byte	byte b = 100;
Short	2 Bytes	short s = 10000;
Int	4 Bytes	int i = 100000;
Long	8 Bytes	long l = 100000000000L;



# Example of Data Types

## Example

```
// Create variables of different data types
int items = 50;
float costPerItem = 9.99f;
float totalCost = items * costPerItem;
char currency = '$';

// Print variables
System.out.println("Number of items: " + items);
System.out.println("Cost per item: " + costPerItem + currency);
System.out.println("Total cost = " + totalCost + currency);
```





# Floating Point Data Types

## Float

The float data type stores single-precision floating-point numbers, allowing for decimal values.

## Double

The double data type stores double-precision floating-point numbers, providing higher precision for decimal values.

# Other Primitive Data Types

## Character

The char data type stores a single character, like a letter, number, or symbol.

## Boolean

The boolean data type stores a truth value, either true or false. It's used for representing conditions and logic in your programs.

### Example

```
boolean isJavaFun = true;
boolean isFishTasty = false;
System.out.println(isJavaFun);    // Outputs true
System.out.println(isFishTasty);  // Outputs false
```

### Example

```
char myVar1 = 65, myVar2 = 66, myVar3 = 67;
System.out.println(myVar1);
System.out.println(myVar2);
System.out.println(myVar3);
```





# Non-Primitive Data Types

## Definition

**Non-primitive data types are constructed using primitive data types. They provide more complex ways to represent data and manage information.**

## Examples

**The String data type represents sequences of characters, while arrays store collections of elements of the same data type.**

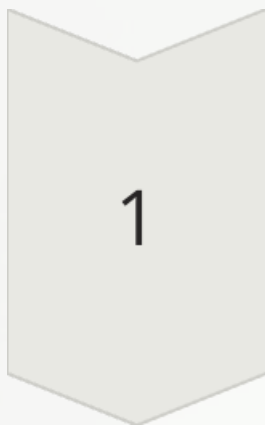
# Date Data Type

## Date

The Date data type represents dates and times, allowing you to work with specific points in time within your programs.

## LocalDate Example

The LocalDate class provides functionality for working with dates, allowing you to create, manipulate, and compare dates.



# Import from sccanner

```
java
import java.util.Scanner;

public class UserInfo {
    public static void main(String[] args) {
        // ایجاد یک شیء Scanner برای خواندن ورودی از کنسول
        Scanner scanner = new Scanner(System.in);

        // دریافت نام کاربر
        System.out.print("لطفاً نام خود را وارد کنید: ");
        String name = scanner.nextLine();

        // دریافت سن کاربر
        System.out.print("لطفاً سن خود را وارد کنید: ");
        int age = scanner.nextInt();

        // نمایش پیام خوش آمدگویی
        System.out.println("سال است " + age + " سن شما " + name + " سلام");

        // بستن شیء Scanner
        scanner.close();
    }
}
```

# Import from scanner

```
java
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in); // ایجاد یک شیء Scanner

        System.out.print("لطفاً نام خود را وارد کنید");
        String name = scanner.nextLine(); // خواندن یک رشته از ورودی

        System.out.print("لطفاً سن خود را وارد کنید");
        int age = scanner.nextInt(); // خواندن یک عدد صحیح از ورودی

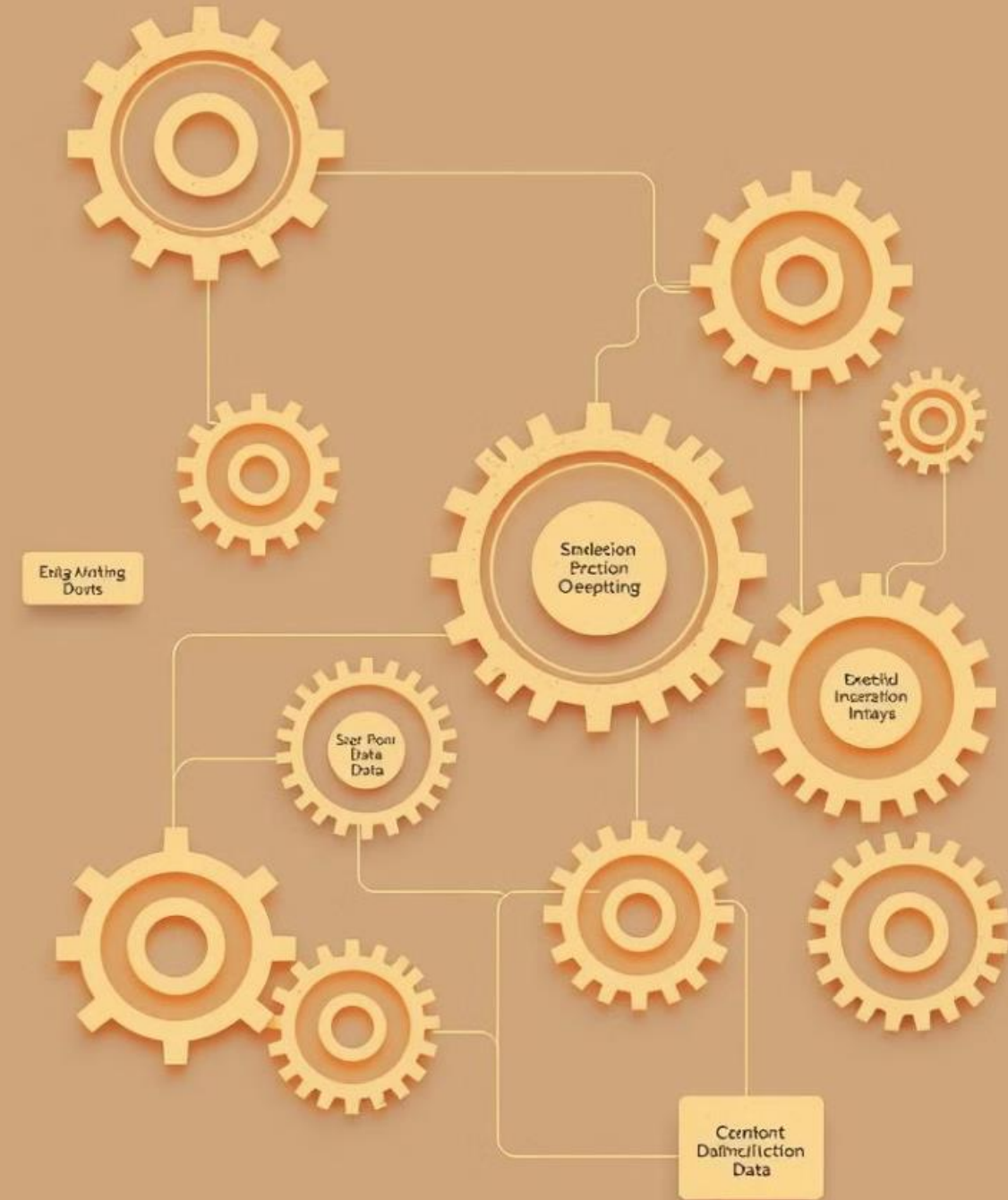
        System.out.println("سال است " + age + " سن شما " + name + " سلام");

        scanner.close(); // بستن شیء Scanner
    }
}
```



# Summary

Understanding data types is essential for any Java programmer. This presentation has provided a fundamental overview of data types, including primitive types like byte, short, int, long, float, double, char, and boolean, as well as non-primitive types like String and arrays. This knowledge lays the groundwork for understanding complex data structures and algorithms in Java.



# Homework

## Exercise:

Add the correct data type for the following variables:

```
 myNum = 9;  
 myFloatNum = 8.99f;  
 myLetter = 'A';  
 myBool = false;  
 myText = "Hello World";
```