Java Data Types

This repository explains and demonstrates various **data types in Java** with examples. Java supports two main types of data: **Primitive Data Types** and **Reference Data Types**.

Table of Contents

- Primitive Data Types
  - byte
  - short
  - int
  - long
  - float
  - double
  - char
  - boolean
- Reference Data Types
  - String
  - Array
  - Object
  - Null

Primitive Data Types

Primitive data types are the most basic data types in Java. They represent simple values such as integers, floating-point numbers, characters, and boolean values.

byte (8-bit integer)

The `byte` type represents an 8-bit signed integer. The range is from **-128 to 127**.

```
byte b = 100;
System.out.println(b);  // Output: 100
```

---

short (16-bit integer)

The `short` type represents a 16-bit signed integer. The range is from **-32,768 to 32,767**.

```
short s = 5000;
System.out.println(s);  // Output: 5000
```

---

int (32-bit integer)

The `int` type is the most commonly used integer type. The range is from **-2,147,483,648 to 2,147,483,647**.

```
int i = 100000;
System.out.println(i);  // Output: 100000
```

---

long (64-bit integer)

The `long` type is used for larger integer values. The range is from **-2^63 to 2^63-1**. You must append an `L` to indicate a `long` literal.

```
long l = 10000000000L;
System.out.println(l);  // Output: 10000000000
```

---

float (32-bit floating-point number)

The `float` type represents a 32-bit floating-point number. To define a `float`, you must append an `f` to the number.

```
float f = 3.14f;
System.out.println(f);  // Output: 3.14
```

---

double (64-bit floating-point number)

The `double` type is used for decimal numbers, providing more precision than `float`. It is the default for floating-point numbers.

```java
double d = 3.14159;
System.out.println(d);  // Output: 3.14159
```

---

char (16-bit Unicode character)

The `char` type stores a single 16-bit Unicode character.

```java
char c = 'A';
System.out.println(c);  // Output: A
```

---

boolean (true or false)

The `boolean` type represents one of two values: `true` or `false`.

```java
boolean isJavaFun = true;
System.out.println(isJavaFun);  // Output: true
```

Reference Data Types

Reference data types refer to objects and arrays in Java. They point to a memory location where the actual data is stored.

String (Sequence of characters)

A `String` represents a sequence of characters. In Java, it is an object, and it is used to store text.

```java
String str = "Hello, Java!";
System.out.println(str);  // Output: Hello, Java!
```

Array (A collection of elements of the same type)

An array in Java is a collection of variables of the same type, referenced by a single variable.

```java
int[] numbers = {1, 2, 3, 4, 5};
System.out.println(numbers[0]);  // Output: 1
```

Object (Instances of a class)

An object is an instance of a class. You can create custom types using classes in Java.

```
class Person {
    String name;
    int age;

    Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
}

Person person1 = new Person("Alice", 30);
System.out.println(person1.name + " is " + person1.age + " years old.");
// Output: Alice is 30 years old.
```

Null (Represents no object reference)

The `null` value is a reference that points to no object. It indicates that a reference variable is not pointing to any memory location.

```
String name = null;
System.out.println(name);  // Output: null
```
