Java DataTypes:

In Java, data types define the type and size of data that can be stored in variables. Java supports two types of data: primitive data types and reference data types. Here's a list of some commonly used data types in Java, along with sample examples for each:

1. **Primitive Data Types**:
   - These data types represent basic values and are directly supported by the language.

   a. `int`: Used to store whole numbers (32-bit signed integer).
      ```java
      int age = 30000000;
1 byte =32 bit
      ```

   b. `double`: Used to store floating-point numbers with double precision (64-bit).
      ```java
      double pi = 3.14159265359;
      ```

   c. `char`: Used to store a single character (16-bit Unicode value).
      ```java
      char grade = 'A';
      ```

   d. `boolean`: Used to store true or false values.
      ```java
      boolean isJavaFun = true;false
      ```

   e. `byte`: Used to store small integers (8-bit signed integer).
      ```java
      byte flags = 4;
      ```

   f. `short`: Used to store short integers (16-bit signed integer).
      ```java
      short temperature = -10;
      ```

   g. `long`: Used to store large whole numbers (64-bit signed integer).
      ```java
      long population = 789_000_000L; // Note the 'L' suffix to denote a long literal.
      ```

h. `float`: Used to store floating-point numbers with single precision (32-bit).
```java
float weight = 68.5f; // Note the 'f' suffix to denote a float literal.
```

2. **Reference Data Types**:
- These data types refer to objects created using classes or interfaces.

a. `String`: Used to store sequences of characters (text).
```java
String greeting = "Hello, World!";
```

b. `Arrays`: Used to store multiple values of the same type in a contiguous memory block.
```java
int[] numbers = {1, 2, 3, 4, 5};
```

c. `Classes`: Used to create user-defined data types (objects).
```java
class Person {
    String name;
    int age;
}

Person person1 = new Person();
person1.name = "Alice";
person1.age = 25;
```

d. `Interfaces`: Used to define a contract for classes that implement them.
```java
interface Shape {
    void draw();
     String name =’deepak
}

class Circle implements Shape {
    @Override
    public void draw() {
        System.out.println("Drawing a circle.");
                System.out.println(name);
```

```
    }
}
```

Java's data types provide flexibility and precision for storing different types of data, allowing developers to build robust and efficient applications.