Java Reserved Words:

In Java, reserved words, also known as keywords, are predefined words that have specific meanings and functionalities within the language. These words cannot be used as identifiers (e.g., variable names, class names, method names) because they are already reserved for specific purposes. Here is a list of Java reserved words along with a sample example for each:

1. `abstract`:
   - Used to declare an abstract class or abstract method.
   ```java
   abstract class Shape {
       abstract void draw();


   }
   ```

2. `assert`:
   - Used to perform runtime checks, usually for debugging purposes.
   ```java
   int age = 18;
   assert age >= 18 : "Not old enough to vote";
   ```

3. `boolean`:
   - Represents a boolean data type, which can have the values `true` or `false`.
   ```java
   boolean isJavaFun = true;
   ```

4. `break`:
   - Used to terminate a loop or switch statement.
   ```java
   for (int i = 0; i < 5; i++) {
       if (i == 4) {
           break;
       }
       System.out.println(i);
   }

0 1 2 3
   ```

5. `byte`:
   - Represents a byte data type, which is a signed 8-bit integer.
   ```java
   byte age = 25;
   ```

6. `case`:
   - Used in a switch statement to define different cases.
   ```java
   int day = 2;
   switch (day) {
       case 1:
           System.out.println("Monday");
           break;
       case 2:
           System.out.println("Tuesday");
           break;
       // Other cases...
   }
   ```

7. `catch`:
   - Used in exception handling to catch and handle exceptions.
   ```java
   try {
       // Code that may throw an exception
   } catch (Exception e) {
       // Exception handling code
   }
   ```

8. `char`:
   - Represents a character data type, enclosed in single quotes.
   ```java
   char grade = 'A';

String Name=”Deepak’
Char grade =”A’
   ```

9. `class`:
   - Used to define a class.
   ```java
   public class MyClass {
   ```

```
    // Class members and methods
}
```

10. `const` (not used in Java anymore):
    - Previously used to define a constant variable. Note that `final` is now preferred for constant variables.
    ```java
    final int MAX_VALUE = 100;


#100
Final MAX_VALUE =100


    ```

11. `continue`:
    - Used to skip the rest of the loop and move to the next iteration.
    ```java
    for (int i = 0; i < 10; i++) {
        if (i % 2 == 0) {
            continue;
        }
        System.out.println(i);
    }
    ```

12. `default`:
    - Used in a switch statement as a default case when no other case matches.
    ```java
    int day = 7;
    switch (day) {
        case 1:
            System.out.println("Monday");
            break;
        // Other cases...
        default:
            System.out.println("Invalid day");
    }
    ```

13. `do`:
    - Used to start a do-while loop.

```java
int i = 0;
do {
    System.out.println(i);
    i++;
} while (i < 5);
```

14. `double`:
    - Represents a double-precision 64-bit floating-point data type.
    ```java
    double pi = 3.14159;
    ```

15. `else`:
    - Used in an if-else statement to define the alternative code block.
    ```java
    int age = 25;
    if (age >= 18) {
        System.out.println("You are an adult.");
    } else {
        System.out.println("You are a minor.");
    }
    ```

16. `enum`:
    - Used to define an enumeration, a special data type for defining constant values.
    ```java
    enum Day {
        MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, SUNDAY
    }
    ```

17. `exports` (used in Java modules):
    - Used to export a package from a module in Java modules.
    ```java
    // module-info.java
    module com.example.mymodule {
        exports com.example.mypackage;
    }
    ```

18. `extends`:
    - Used to create a subclass by extending a superclass (inheritance).

```java

Class ParentClass{

        ///methods


class ChildClass extends ParentClass {
    // Class members and methods
}
```

19. `final`:
    - Used to declare a constant variable or to make a class, method, or variable unchangeable (immutable).
    ```java
    final int MAX_VALUE = 100;
    final class ImmutableClass {
        // Class members and methods
    }
    ```

20. `finally`:
    - Used in exception handling to define a block of code that will be executed regardless of whether an exception is thrown or not.
    ```java
    try {
        // Code that may throw an exception
    } catch (Exception e) {
        // Exception handling code
    } finally {
        // Code that will be executed regardless of exceptions
    }
    ```

These are the reserved words in Java, each with its specific purpose in the language. As a programmer, it's essential to be familiar with these keywords and use them appropriately when writing Java code.