Variables and Data Types in Python

In this document, we will explore the fundamental concepts of variables and data types in Python, a versatile and widely-used programming language. Understanding how to effectively use variables and data types is crucial for writing efficient and effective code. This guide will provide an overview of what variables are, how to declare them, and the various data types available in Python.

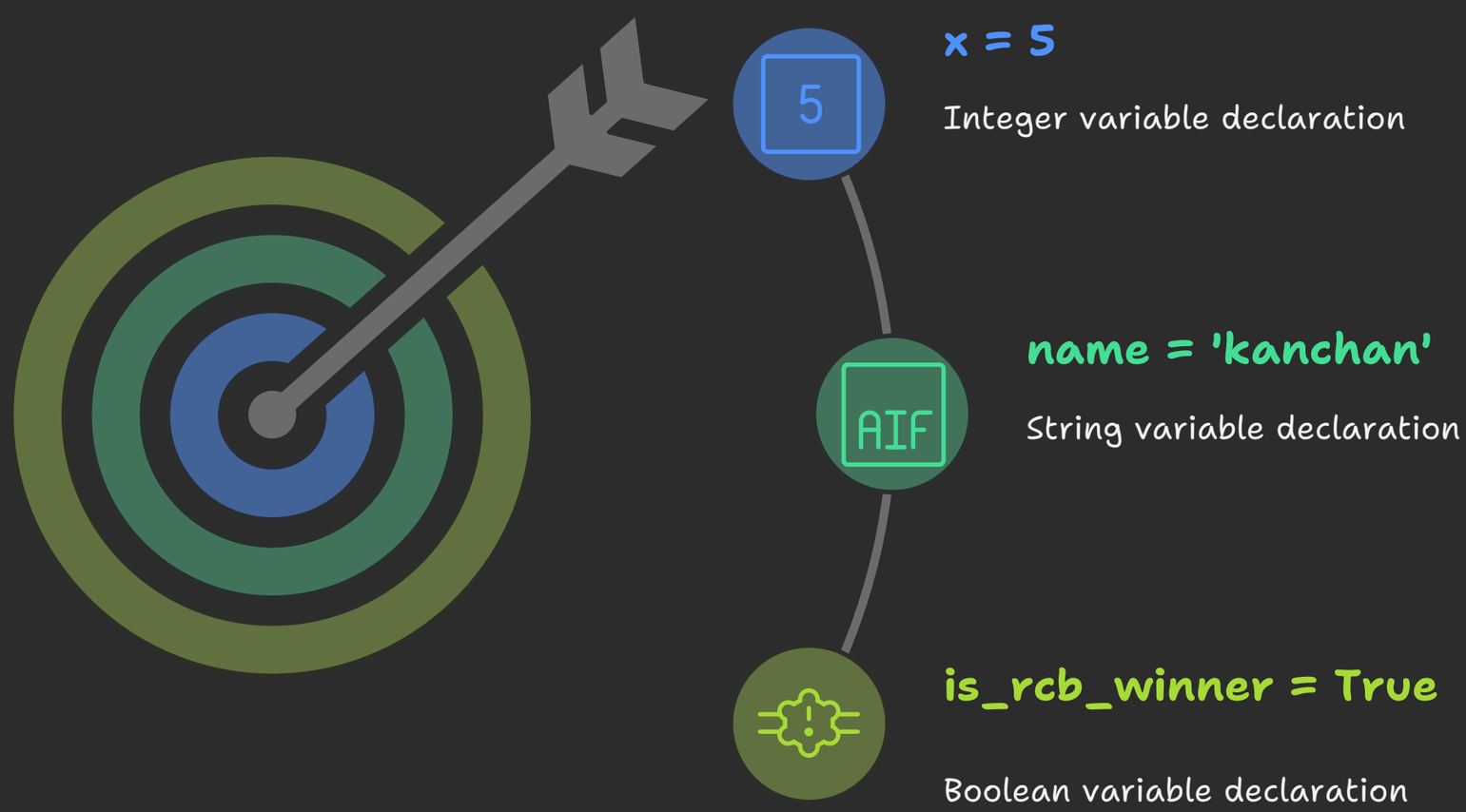
Variables in Python

A variable in Python is a symbolic name that is a reference or pointer to an object. Variables are used to store information that can be referenced and manipulated in a program. In Python, you do not need to declare a variable before using it; you can simply assign a value to it.

Declaring Variables

To declare a variable, you simply assign a value to a name using the equals sign (**=**). For example:

```
x = 5
name = "Alice"
is_active = True
```



In the above example, **x** is a variable that holds an integer value, **name** holds a string, and **is_rcb_winner** holds a boolean value.

Variable Naming Rules

When naming variables in Python, there are a few rules to follow:

- Variable names must start with a letter (a-z, A-Z) or an underscore (_).
- The rest of the variable name can contain letters, numbers (0-9), and underscores.
- Variable names are case-sensitive (e.g., **myVar** and **myvar** are different variables).
- Avoid using Python reserved keywords as variable names.(eg- True, False)

Data Type conversion

Data Types in Python

Python has several built-in data types that are used to define the type of data a variable can hold. The most common data types include:

1. Numeric Types

- **int**: Represents integer values (e.g., **5**, **-3**, **42**).
- **float**: Represents floating-point numbers (e.g., **3.14**, **-0.001**, **2.0**).
- **complex**: Represents complex numbers (e.g., **3 + 4j**).

2. Sequence Types

- **list**: Represents ordered, mutable collections of items (e.g., **[1, 2, 3]**, **["apple", "banana", "cherry"]**).
- **tuple**: Represents ordered, immutable collections of items (e.g., **(1, 2, 3)**, **("apple", "banana", "cherry")**).

3. Mapping Type

- **dict**: Represents key-value pairs (e.g., **{"winner": "RCB", "edition": 18}**).

4. Set Types

- **set**: Represents an unordered collection of unique items (e.g., **{1, 2, 3}**).
- **frozenset**: Represents an immutable version of a set.

5. Boolean Type

- **bool**: Represents boolean values, either **True** or **False**.

Type Conversion

Python allows for type conversion, which means you can convert one data type to another. This can be done using built-in functions such as **int()**, **float()**, **str()**, and **bool()**. For example:

```
# Converts string to integer
num_str = "123"
num_int = int(num_str)
```