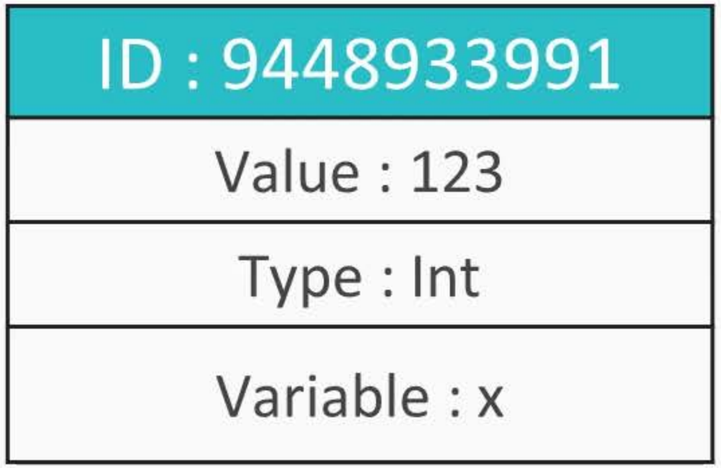
In this Video, we are going to take a look at these topics...

- How does Python store data in memory?
- What are mutability and immutability?
- How exactly are they different in terms of behavior?

What Happens When You Create a Data Object?

```
>>> x = 123
```



How Immutable Objects Store Data

```
>>> x = 123
>>> y = 123
```

ID : 9448933991
Value : 123
Type : Int
Variable : x, y

```
>>> y = y + 1
```

ID : 9448934156
Value : 124
Type : Int
Variable : y

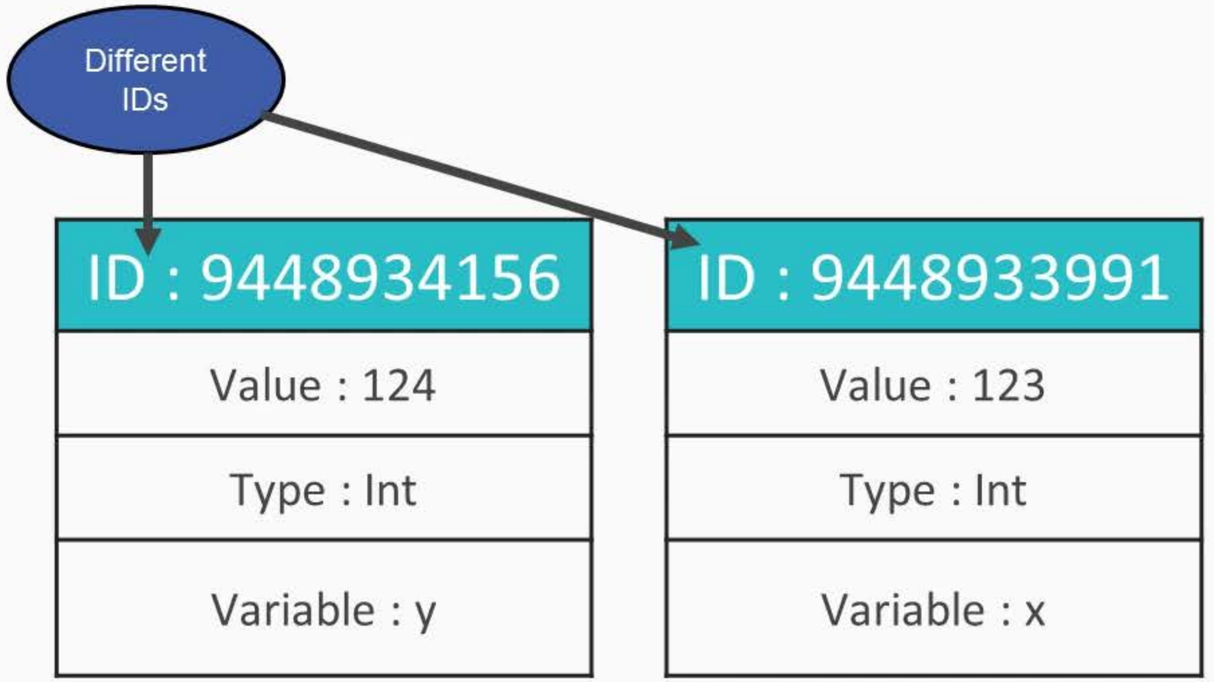
ID : 9448933991
Value : 123
Type : Int
Variable : x

How Immutable Objects Store Data

```
>>> x = 123
>>> y = 123
```

ID : 9448933991
Value : 123
Type : Int
Variable : x, y

```
>>> y = y + 1
```



How Mutable Objects Store Data

```
>>> my_list = [1, 2, 3]
```

Same IDs

```
>>> my_list = [1, 2, 3]
>>> my_list.append(4)
>>> my_list.append(5)
```

List of Immutable Data Types

- Integer
- Float
- Complex
- Bool
- Frozenset

List of Immutable Data Types (Continued)

- Byte
- String
- Tuple
- Range

List of Mutable Data Types

- Lists
- Dictionary
- Set
- Byte array
- User-defined classes

List of Mutable Data Types

- Lists
- Dictionary
- Set
- Byte array
- User-defined classes

Summary

- Python represents all its data as objects
- Python assigns a unique ID when creating a new object
- This ID can be accessed using the `id()` function
- Mutable objects allow modification of variables while keeping the same ID
- When we try to modify a variable, Python creates a new object with a different ID and assigns it to the variable