Welcome to Data Science Online Bootcamp

Day 3

dφ

Democratizing Data Science Learning

Learning Objectives

Conditional Statement

Dictionary

Conditional Statement: if-else-elif

- In the real world, we commonly evaluate information around us and then choose one course of action or another based on what we observe:

  > If the weather is nice, then I'll go for a walk. (It's implied that if the weather isn't nice, then I won't go for a walk.)

- In a Python program, the if statement is how you perform this sort of decision-making. It allows for **conditional execution of a statement** or group of statements **based on the value of an expression**.

Tutorial on Conditional Statements

If

- Syntax (how to write If statement in python?)-

```
if test expression/condition:
    statement(s)
```

- Here, the program evaluates the test expression and will execute statement(s) only if the text expression is True.
- If the text expression is False, the statement(s) is not executed.
- In Python, the **body of the if statement is indicated by the indentation.** Body starts with an indentation and the **first unindented line marks the end.**
- Python interprets **non-zero values as True (even negative values).** None and 0 are interpreted as False.

If

- Flowchart:

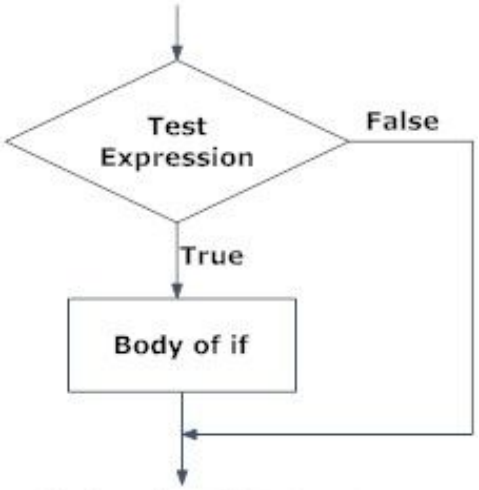


Fig: Operation of if statement

- Example: Python program to detect if a number is even. (sign % tells us the remainder of an expression. Any number that has a remainder 0 after dividing by 2 must be even.)

```
z = 4
if z%2 ==0:
    print("z is even") #indentation is important
```

```
z is even
```

If - else

- Syntax-

```
if test expression:
    Body of if
else:
    Body of else
```

- The if..else statement evaluates test expression and will execute the body of if only when the test condition is True.

- If the condition is False, the body of else is executed. Indentation is used to separate the blocks.

If-else

- Flowchart:

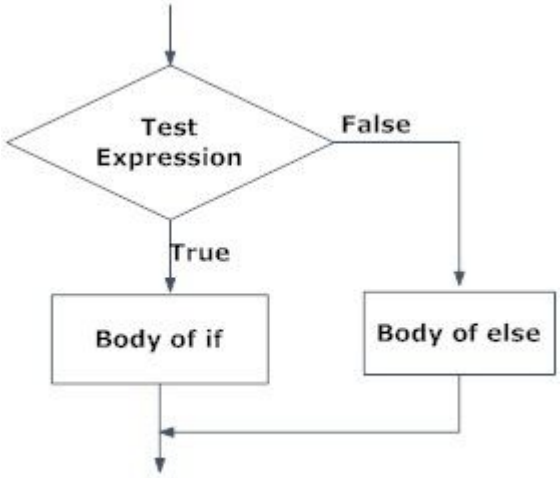


Fig: Operation of if...else statement

- Example:

```
z = 5
if z%2 ==0:
    print("z is even")
else:                          #don't need to specify condition
    print("z is odd")
```

```
z is odd
```

If - elif - else

- Syntax-

```
if test expression:
    Body of if
elif test expression:
    Body of elif
else:
    Body of else
```

- The elif is short for else if. It allows us to check for multiple expressions.
- If the condition for if is False, it checks the condition of the next elif block and so on.
- If all the conditions are False, the body of else is executed.
- Only one block among the several if...elif...else blocks is executed according to the condition.
- The if block can have only one else block. But it can have multiple elif blocks.

If-elif -else

- Flowchart:

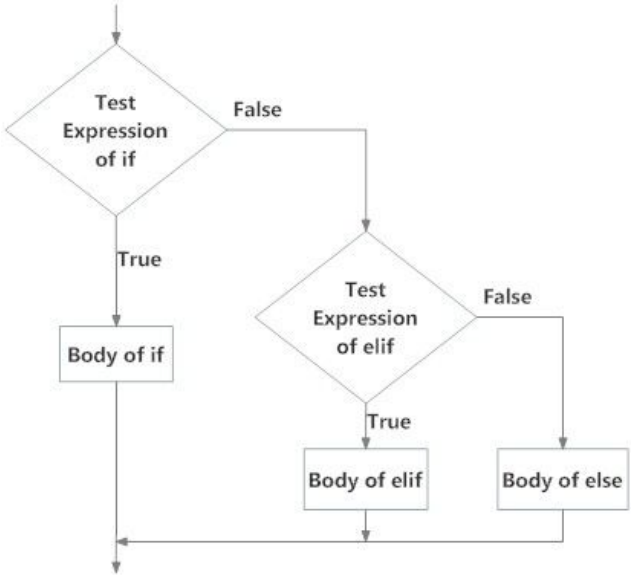


Fig: Operation of if...elif...else statement

- Example:

```
z = 5
if z%2 == 0:
    print("z is divisible by 2")
elif z%3 == 0:
    print("z is divisible by 3")
else:                       #don't need to specify condition
    print("z is neither divisible by 2 nor by 3")
```

```
z is neither divisible by 2 nor by 3
```

If-elif -else

- Now let's try to understand the given example.
  - We first assigned the value 5 to x
  - The control shifts to the next line where x%2 is checked. 5 is not divisible by 5 and so the control doesn't shift to the body of if.
  - Then, the elif statement: x%3 is executed. Since 5 is not divisible by 3, the body of elif is not executed.
  - Finally, the else statement is executed and the control shifts to the body of the control statement. The print statement ("z is neither divisible by 2 nor by 3" is executed.

- Point to be noted: The conditions are checked in a top to bottom order. If any of the above if or elif condition is True, it'll be executed and no further conditions will be checked.

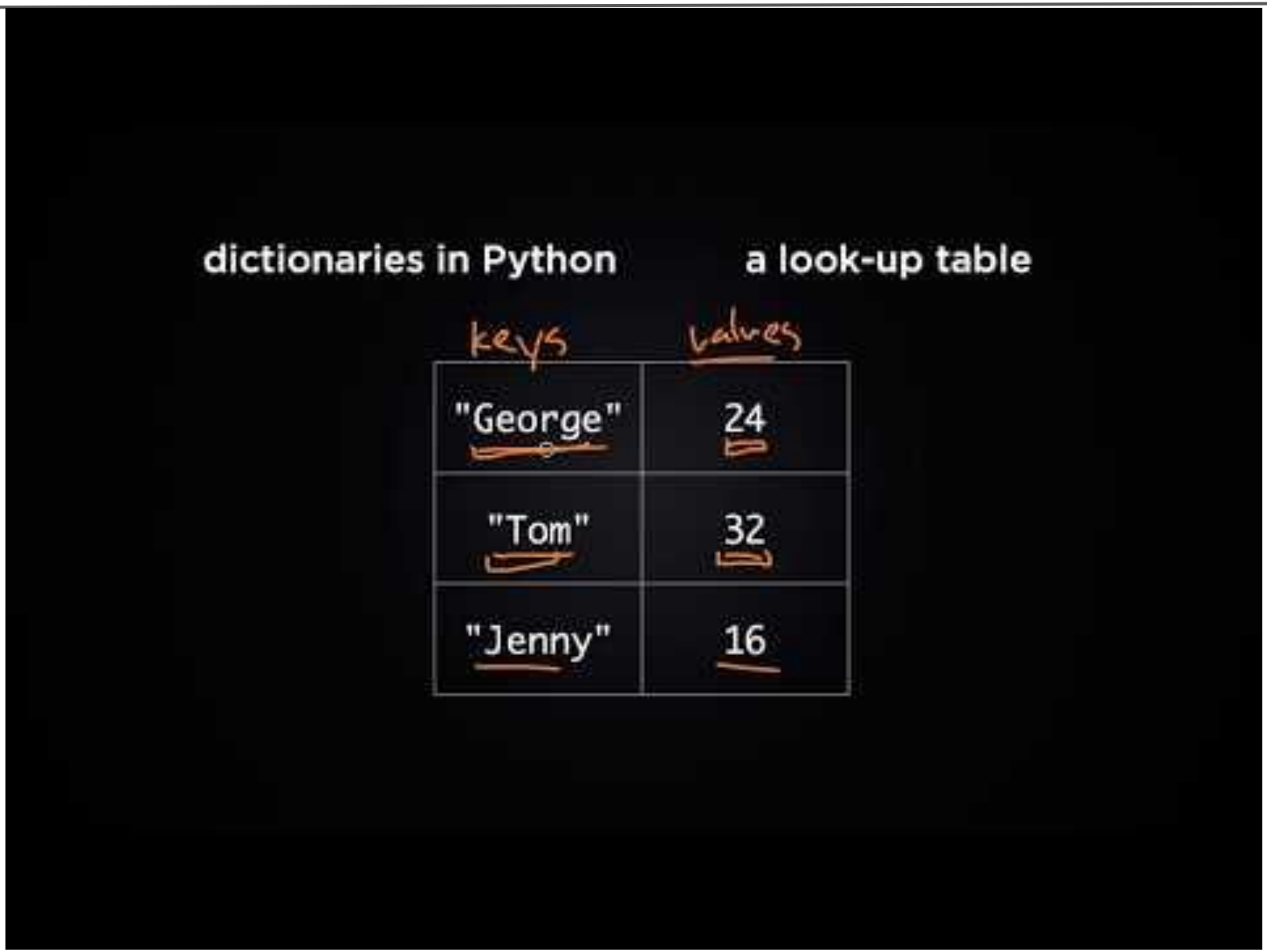
- Can you figure out what will z=6 print in the given example?

Dictionary

- Dictionary is unordered collection of key-value pairs.
- Real word dictionaries are a good analogy to understand them: they contain a list of items(words), each item has a key(the word) and a value(the word's meaning).
- It generally is used when we have a huge amount of data.
- It is defined within braces with each item being in the form of key: value pair. Syntax –

```
my_dict = {
"key1":"value1",
"key2":"value2",
}
```

Tutorial on Dictionaries



Dictionary

- The keys in a dictionary must always be unique and immutable. This is the reason dictionary keys can be String but not List.
- On the other hand, Values in a dictionary can be of any datatype and can be duplicated
- Dictionary keys are case sensitive, same name but different cases of Key will be treated distinctly.
- Example:

```
my_dict = {1: 'Blue', 2: 'Yellow', 3: 'Red'}
```

```
my_dict
```

```
{1: 'Blue', 2: 'Yellow', 3: 'Red'}
```

Looping over Dictionary

- Let's say we have a dictionary containing countries as keys and their populations as values.

- For looping through a dictionary, we use a method called items( ). Similar to enumerate, it gives us both the keys and values of a dictionary.

```
world = { "afghanistan":30.55,
          "albania":2.77,
          "algeria":39.21 }

for key, value in world.items() :
    print(key + " -- " + str(value))
```

```
afghanistan -- 30.55
albania -- 2.77
algeria -- 39.21
```

Interested to learn more about Dictionaries?

For additional practice on dictionaries, visit: https://www.w3schools.com/python/python_dictionaries.asp

Let's Practice!

1. Take values of length and breadth of a rectangle from user and check if it is square or not.

2. Take two int values from user and print greatest among them.

3. Write a program to read the age of a candidate and determine whether it is eligible for casting his/her own vote.

4. Write a Python program to add a key to a dictionary.
   Sample Dictionary : {0: 10, 1: 20}
   Expected Result : {0: 10, 1: 20, 2: 30}

5. Below are the two lists, convert it into a dictionary.
   keys = ['Ten', 'Twenty', 'Thirty']
   values = [10, 20, 30]
   Expected output:
   {'Ten': 10, 'Twenty': 20, 'Thirty': 30}

Let's Practice!

6. Access the value of key 'history'

```
sampleDict = {
"class":{
    "student":{
      "name":"Mike",
      "marks":{
        "physics":70,
        "history":80
      }
    }
  }
}
```

Expected output:

80

Let's Practice!

7. Given the following dictionary:
   inventory = {
   'gold' : 500,
   'pouch' : ['flint', 'twine', 'gemstone'],
   'backpack' : ['xylophone','dagger', 'bedroll','bread loaf']
   }

   Try to do the following:

- Add a key to inventory called 'pocket'.
- Set the value of 'pocket' to be a list consisting of the strings 'seashell', 'strange berry', and 'lint'.
- .sort()the items in the list stored under the 'backpack' key.
- Then .remove('dagger') from the list of items stored under the 'backpack' key.
- Add 50 to the number stored under the 'gold' key.

Quiz

Learners who enrolled for the bootcamp can access DPhi Learning Platform to submit their day-wise module quizzes.

- **Link to learning platform:** https://learn.dphi.tech/
- **FAQ's by learners:** https://bit.ly/DPhiBootcampFAQ

That’s it for the day. Thank you!

Feel free to post any queries in the #help channel on Slack