Dictionaries

- Dictionaries are ordered collection of data items.
- They store multiple items in a single variable.
- Dictionaries items are key-value pairs that are separated by commas and enclosed within curly brackets {}.

In [1]:
```
d = {'key1':'item1','key2':'item2'}
d
```

Out[1]:
```
{'key1': 'item1', 'key2': 'item2'}
```

In [2]:
```
d=dict()
d
```

Out[2]:
```
{}
```

In [3]:
```
info = {'name':'Karan', 'age':19, 'eligible':True}
print(info)
```

```
{'name': 'Karan', 'age': 19, 'eligible': True}
```

In [4]:
```
#Accessing single values

#Values in a dictionary can be accessed using keys.
#We can access dictionary values by mentioning keys either in square brackets
#or by using get method.

info = {'name':'Heena', 'age':25, 'eligible':True}
print(info['name'])
print(info.get('age'))
```

```
Heena
25
```

In [5]:
```
# print all the values in the dictionary using values() method.

info = {'name':'Heena', 'age':25, 'eligible':True}
print(info.values())
```

```
dict_values(['Heena', 25, True])
```

In [6]:
```
# print all the keys in the dictionary using keys() method.

print(info.keys())
```

```
dict_keys(['name', 'age', 'eligible'])
```

In [7]:
```
# print all the key-value pairs in the dictionary using items() method

print(info.items())
```

```
dict_items([('name', 'Heena'), ('age', 25), ('eligible', True)])
```

In [10]:
```
# create a new key and assign a value to it

info = {'name':'Heena', 'age':19, 'eligible':True}

print(info)
```

```
{'name': 'Heena', 'age': 19, 'eligible': True}
```

In [12]:
```
# update() method updates the value of the key provided to it
#if the item already exists in the dictionary,
#else it creates a new key-value pair.

info.update({'age':20})
info.update({'DOB':1995})
print(info)
```

```
{'name': 'Heena', 'age': 20, 'eligible': True, 'DOB': 1995}
```

In [13]:
```
# use the del keyword to remove a dictionary item.

info = {'name':'Karan', 'age':19, 'eligible':True, 'DOB':2003}
del info['age']
print(info)
```

```
{'name': 'Karan', 'eligible': True, 'DOB': 2003}
```

In [14]:
```
# use the copy() method to copy the contents of one dictionary
# into another dictionary.

newDictionary = info.copy()
print(newDictionary)
```

```
{'name': 'Karan', 'eligible': True, 'DOB': 2003}
```

In [15]:
```
# use the dict() function to make a new dictionary with the items of
# original dictionary.

newDictionary = dict(info)
print(newDictionary)
```

```
{'name': 'Karan', 'eligible': True, 'DOB': 2003}
```

In [16]:
```
# Create dictionary with duplicate keys
d1 = {"1": 1, "1": 2}
print(d1)

# It will only print one key, although no error was thrown
# If we  try to access this key, then it'll return 2,
# so the value of the second key
print(d1["1"])
```

```
{'1': 2}
2
```

In [17]:
```
# Create a two-element dictionary using curly brackets
d2 = {"Rishabh": {"Age": 27, "Hometown": "Shimla"},
      "Taarun": {"Age": 25, "Hometown": "Kullu"}}
```

In [18]:
```
# Access the value associated with the key 'John'
print("Rishabh's personal data is:")
print(d2["Rishabh"])
```

```
Rishabh's personal data is:
{'Age': 27, 'Hometown': 'Shimla'}
```

Booleans

In [19]:
```
True
```

Out[19]:
```
True
```

In [20]:
```
False
```

Out[20]:
```
False
```

Comparison Operators

In [21]:
```
# check one is greater than two it return boolean result

1>2
```

Out[21]: False

In [22]:
```
# check two is greater than one it return boolean result

2>1
```

Out[22]: True

In [23]:
```
# check one is greater than equal to one it return boolean result

1>=1
```

Out[23]: True

In [24]:
```
# check one is less than equal to four it return boolean result

1 <= 4
```

Out[24]: True

In [25]:
```
# check one is equal to one it return boolean result

1 == 1
```

Out[25]: True

In [26]:
```
# check string hi is equal to bye, it return boolean result

'hi' == 'bye'
```

Out[26]: False

In [27]:
```
# check string bye is equal to bye, it return boolean result


'bye' == 'bye'
```

Out[27]: True

In [28]:
```
'hi' == 'hi'
```

Out[28]: True

Logical Operators

In [29]:
```
# And operator used to check the two statements

(1 > 2) and (2 < 3)
```

Out[29]: False

In [30]:
```
#usage of OR operator

(1 > 2) or (2 < 3)
```

Out[30]: True

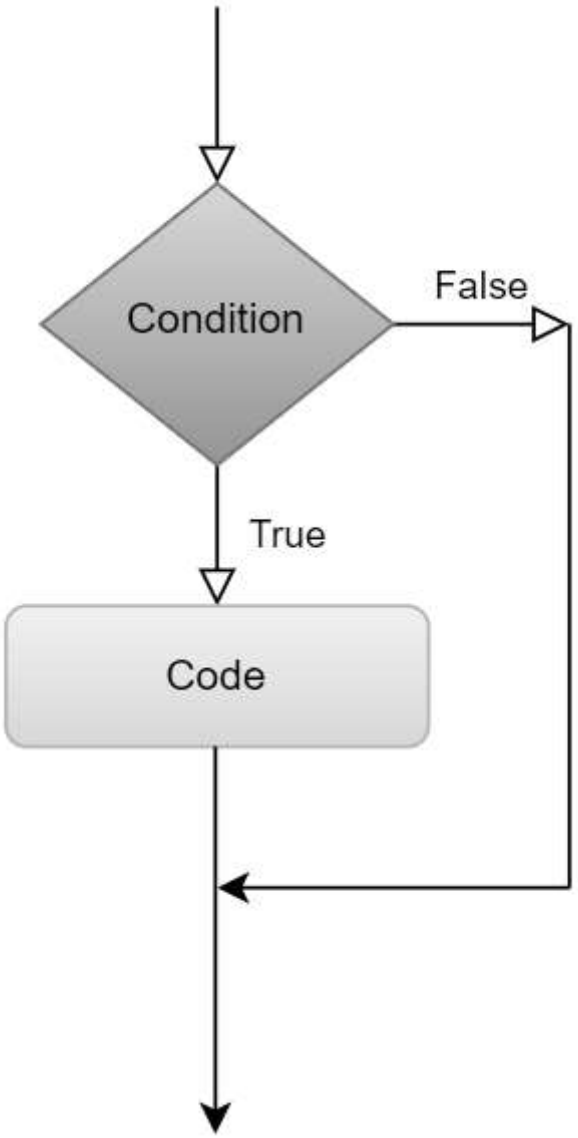
In [31]:
```
#usage of NOT operator

x = False

not x
```

Out[31]: True

if,elif, else Statements



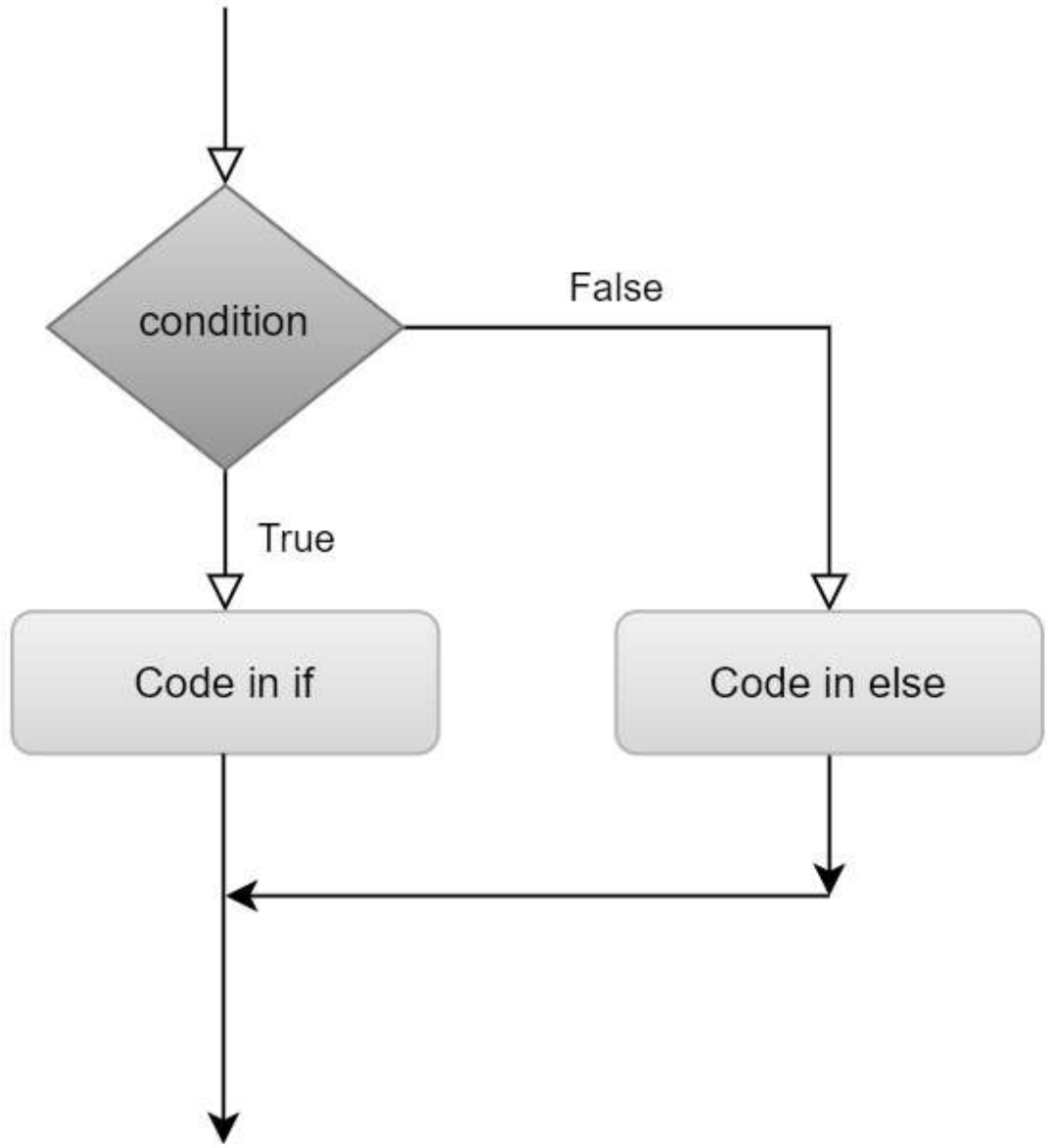
In [34]:
```
# execute the block of code inside if statement if the expression evaluates Tr

if 1 < 2:
    print('Yep!')
```

```
Yep!
```

In [35]:
```
# execute the block of code inside if statement if the expression evaluates Tr

if 1 > 2:
    print('Yep!')
```



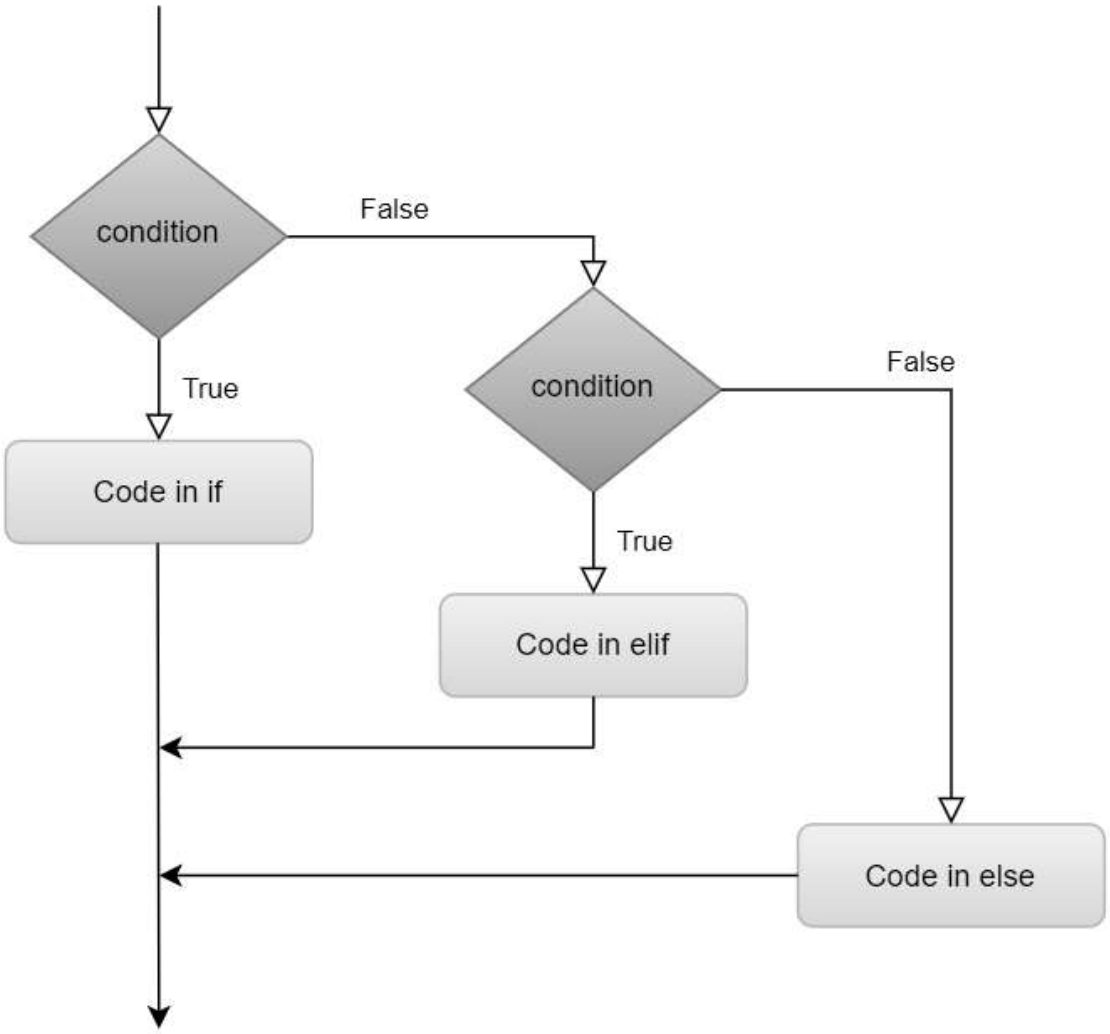
In [36]:
```
#execute the block of code inside if statement if the expression evaluates Tru
# otherwise else will execute

if 1 < 2:
    print('first')
else:
    print('last')
```

```
first
```

In [37]:
```
if 1 > 2:
    print('first')
else:
    print('last')
```

```
last
```



In [38]:
```
if 1 == 2:
    print('first')
elif 3 == 3:
    print('middle')
else:
    print('Last')
```

```
middle
```

Nested if Statement

In [39]:
```
num = 18
if (num < 0):
    print("Number is negative.")
elif (num > 0):
    if (num <= 10):
        print("Number is between 1-10")
    elif (num > 10 and num <= 20):
        print("Number is between 11-20")
    else:
        print("Number is greater than 20")
else:
    print("Number is zero")
```

```
Number is between 11-20
```

In [ ]: