Python Conditional Statements

Python uses conditional statements to make decisions based on conditions. The main types are:

- `if`
- `if-else`
- `if-elif-else`
- nested `if`

---

1. `if` Statement

Description:

Runs a block of code only if the condition is **true**.

Example 1

In [8]:
```
age = 20
if age >= 18:
    print("the condition is True")
    print("You are an adult.")

print("last line")
```

```
the condition is True
You are an adult.
last line
```

In [9]:
```
num = int(input("Enter a number: "))
if num > 0:
    print("the condition is True")
    print("The number is positive")

print("out of if block")
```

```
the condition is True
The number is positive
out of if block
```

In [10]:
```
print(" statements are used to execute a block of code if a specified condition
if 5<0:
    print("This will not be printed because the condition is false.")
```

```
statements are used to execute a block of code if a specified condition is true.
```

In [11]:
```
print(" statements are used to execute a block of code if a specified condition
if 10>100:
    print("This will  be printed because the condition is True.")
```

```
statements are used to execute a block of code if a specified condition is true.
```

In [12]:
```
a=8
if a>6:
    print(a,"is greater than 6.") # Indentation for the block-of-code
print("Independent statement") # Not indented
```

```
8 is greater than 6.
Independent statement
```

if-else

Python `if-else` Statement

Introduction

The `if-else` statement in Python is used to make decisions. It allows the program to execute one block of code if the condition is true and another block if it is false.

Syntax

```
if condition:
    # code to run if condition is True
else:
    # code to run if condition is False
```

In [13]:
```
num = 25
if num > 0:
    print("The number is Positive.")
else:
    print("The number is Negative or Zero.")
```

```
The number is Positive.
```

In [14]:
```
marks = 45

if marks >= 40:
    print("Pass")
else:
    print("Fail")
```

```
Pass
```

In [15]:
```
num = 24

if num % 2 == 0:
    print("Even")
else:
    print("Odd")
```

```
Even
```

In [16]:
```
age = int(input("Enter a age: "))

if age >= 18:
```

```
    print("You are eligible to vote.")
else:
    print("You are not eligible to vote.")
```

```
You are not eligible to vote.
```

if-elif-else

if: Starts a conditional block. Executes the code under it only if the condition is True.

elif (short for "else if"): Adds more conditions to check only if the previous if or elif was False.

else: A final fallback. Executes its code block only if none of the above conditions were True.

```
if condition1:
    # code block if condition1 is true
elif condition2:
    # code block if condition2 is true
elif condition3:
    # code block if condition3 is true
else:
    # code block if none of the above are true
```

In [4]:
```
score = int(input("Enter a number: "))
if score >= 90:
    print("Grade: A")
elif score >= 80:
    print("Grade: B")
elif score >= 70:
    print("Grade: C")
else:
    print("Grade: F")
```

```
Grade: C
```

In [7]:
```
# This code checks if a number is positive, negative, or zero
num = float(input("Enter a number: "))
if num > 0:
    print("The number is positive.")
elif num < 0:
    print("The number is negative.")
else:
    print("The number is zero.")
```

```
The number is positive.
```

In [8]:
```
x = int(input("Enter a number: "))

if x > 10:
    print("x is greater than 10")
elif x == 5:
    print("x is exactly 5")
else:
    print("x is something else")
```

```
x is greater than 10
```

In [13]:
```
# elif can be used to check alternate conditions
country=input("Name of the country:")
print("The country entered is:", country)
if country==country.upper():
    print("The font entered is ALL CAPITAL")
elif country==country.lower():
    print("The font is all lowercase.")
elif country==country.capitalize():
    print("The font received is appropriate.")
else:
    print("Mixed case received")
print("One of the 4 blocks of code was executed.")
```

```
The country entered is: 123
The font entered is ALL CAPITAL
One of the 4 blocks of code was executed.
```

Nested If

In [18]:
```
attendance=65
marks=45
if attendance>=70:
    print("eligible to write exam")
    if marks>=35:
        print("pass...!!!")
    else:
        print("Failed..!!!")

else:
    print("Shotage of attendance..!!")
    print("Not Eligible to write exam")
```

```
Shotage of attendance..!!
Not Eligible to write exam
```

In [2]:
```
attendance=int(input("enter your attentance"))
mark=int(input('enter your marks'))
if attendance>=70:
    print("eligible to write the exam")
    if mark>=35:
        print("pass...!")
    else:
        print("fail")
else:
    print("shortage of attendance")
    print("Not eligible to write exam")
```

```
eligible to write the exam
pass...!
```

In [ ]:
```
msg = "Welcome to the Python programming class."

if "Python" in msg:
    print("This is a Python class!")
else:
    print("This is not a Python class.")
```

While Loop

In [2]:
```
print("print numbers from 1 to 10 using while loop")
i= 1
while i <= 10:
    print(i)
    i += 1

print("loop ended")
```

```
print numbers from 1 to 10 using while loop
1
2
3
4
5
6
7
8
9
10
loop ended
```

In [11]:
```
print("print squares of numbers from 1 to 10 using while loop")
i = 1
while i <= 10:
    square = i * i
    print(square)
    i += 1
```

```
print squares of numbers from 1 to 10 using while loop
1
4
9
16
25
36
49
64
81
100
```

In [ ]:
```
print("print squares of numbers from 1 to 5. with break statement when number is
i = 1
while i <= 5:
    square = i * i
    print(square)
    if square == 25:
        break  # Exit the loop when i is 3

    i += 1
print("end of loop")
```

```
print squares of numbers from 1 to 5. with break statement when number is 5
1
4
9
16
25
end of loop
```

In [13]:
```python
print("program on continue statement")
print("This program will print squares of numbers from 1 to 10, skip the values
i = 1
while i <= 10:
    square = i * i
    if square == 25:
        i += 1  # Skip the value 5
        print("Skipping square of 5")
        continue  # Skip the rest of the loop for this iteration
    print(square)
    i += 1
print("end of loop  ")
```

```
program on continue statement
This program will print squares of numbers from 1 to 10, skip the values at 25.
1
4
9
16
Skipping square of 5
36
49
64
81
100
end of loop
```

In [18]:
```python
count = 0
while count < 5:
    print("Count is:", count)
    count = count+1

print("Loop has ended.")
```

```
Count is: 0
Count is: 1
Count is: 2
Count is: 3
Count is: 4
Loop has ended.
```

In [ ]:
```python
i=1
while i < 5:
    print(f"The {i}st iteration")
    print("The value of i:", i)
    i += 1
```

```
The 1st iteration
The value of i: 1
The 2st iteration
The value of i: 2
The 3st iteration
The value of i: 3
The 4st iteration
The value of i: 4
```

In [8]:
```python
# This code demonstrates a simple while loop
i= 0
while i < 5:
    print("Ashwin")
    i += 1
print("end of loop")
```

```
Ashwin
Ashwin
Ashwin
Ashwin
Ashwin
end of loop
```

In [10]:
```python
i=0
while i < 10:
    if i % 2 == 0:
        print(f"{i} is even")

    i += 1
print("End of loop")
```

```
0 is even
2 is even
4 is even
6 is even
8 is even
End of loop
```

Break Statement

In [11]:
```python
i= 1
while i < 10:
    print("The value of i:", i)
    if i == 5:
        print("Breaking the loop at i =", i)
        break
    i += 1
```

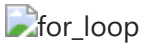
```
The value of i: 1
The value of i: 2
The value of i: 3
The value of i: 4
The value of i: 5
Breaking the loop at i = 5
```

In [1]:
```python
i= 0
while i < 5:
    print("The value of i:", i)
```

```
    i += 1
    if i == 3:
        print("Skipping the rest of the loop when i is 3")
        continue
    print("This line will not be printed when i is 3")
print("End of loop")
```

```
The value of i: 0
This line will not be printed when i is 3
The value of i: 1
This line will not be printed when i is 3
The value of i: 2
Skipping the rest of the loop when i is 3
The value of i: 3
This line will not be printed when i is 3
The value of i: 4
This line will not be printed when i is 3
End of loop
```

For Loop



In [2]:
```
str_var = "Hello  World!"
for char in str_var:
    print(char)
```

```
H
e
l
l
o


W
o
r
l
d
!
```

In [3]:
```
name_list = ["Anu", "Bhanu", "Charlie"]
for name in name_list:
    print(name)
    if name == "Bhanu":
        print("Found Bhanu, breaking the loop.")
        break
print("Loop has ended.")
```

```
Anu
Bhanu
Found Bhanu, breaking the loop.
Loop has ended.
```

In [ ]:
```
name_list = ["ashwin", "divya","john", "charlie"]
for name in name_list:

    if name == "john":
```

```
        print("Found john, skipping to the next iteration.")
        continue
    print("The name is:", name)
print("Loop has ended.")
```

```
The name is: ashwin
The name is: divya
Found Bob, skipping to the next iteration.
The name is: charlie
Loop has ended.
```

In [42]:
```
for i in range(5):
    print("The value of i is:", i)
```

```
The value of i is: 0
The value of i is: 1
The value of i is: 2
The value of i is: 3
The value of i is: 4
```

In [43]:
```
for i in range(2,10):
    print("The value of i is:", i)
```

```
The value of i is: 2
The value of i is: 3
The value of i is: 4
The value of i is: 5
The value of i is: 6
The value of i is: 7
The value of i is: 8
The value of i is: 9
```

In [44]:
```
for i in range(0,10,2):
    print("The value of i is:", i)
```

```
The value of i is: 0
The value of i is: 2
The value of i is: 4
The value of i is: 6
The value of i is: 8
```

Nested for loop

In [23]:
```
num_list = [1, 2, 3]
alpha_list = ["a", "b", "c", "d"]
for num in num_list:
    for alpha in alpha_list:
        print(f"Number: {num}, Letter: {alpha}")
```

```
Number: 1, Letter: a
Number: 1, Letter: b
Number: 1, Letter: c
Number: 1, Letter: d
Number: 2, Letter: a
Number: 2, Letter: b
Number: 2, Letter: c
Number: 2, Letter: d
Number: 3, Letter: a
Number: 3, Letter: b
Number: 3, Letter: c
Number: 3, Letter: d
```

Functions

In [ ]:
```python
def greet() : # Function definition
    print("Hello,good morning!")


greet()  # Calling the function to execute its code
```

```
Hello,good morning!
```

In [16]:
```python
def greet(name):
    print("Hello, good morning" ,name )

greet("Ashwin")
```

```
Hello, good morning Ashwin
```

In [ ]:
```python
def add_numbers(num1, num2):  # Function to add two numbers,not returning the va
    sum= num1 + num2
    print("The sum is:", sum)

add_numbers(5, 10) # calling the function
```

```
The sum is: 15
```

In [ ]:
```python
def add_numbers(num1, num2): # Function to add two numbers,not returning the val
    sum= num1 + num2
    print("The sum is:", sum)

result=add_numbers(5, 10) # calling the function and storing the result
print("The result of the addition is:", result)
```

```
The sum is: 15
The result of the addition is: None
```

In [ ]:
```python
def add_numbers(num1, num2): # Function to add two numbers, returning the value
    """ this function adds two numbers and returns the sum.this is a docstring."
    sum= num1 + num2
    print("The sum is:", sum)
    return sum  # Returning the sum value

result=add_numbers(5, 10) # calling the function and storing the result
print("The result of the addition is:", result)
```

```
The sum is: 15
The result of the addition is: 15
```

In [ ]:
```
add_numbers.__doc__ # Accessing the docstring of the function
```

In [ ]:
```
def area_of_circle(radius):
    """This function calculates the area of a circle given its radius."""
    pi = 3.14
    area = pi * radius * radius
    return area

radius = float(input("Enter the radius of the circle: "))
area = area_of_circle(radius)
```

Lambda Function

In [21]:
```
print("add two numbers using lambda function")
add_two_numbers = lambda x, y: x + y  # Lambda function to add two numbers
result = add_two_numbers(5, 10)  # Calling the lambda function
print(result)  # Printing the result
```

```
add two numbers using lambda function
15
```

Filter Function

In [23]:
```
print("filter even numbers from a list using filter function")
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
filter_function = lambda x: x % 2 == 0

even_numbers = filter(filter_function, numbers)
list(even_numbers)  # Convert the filter object to a list to see the results
```

```
filter even numbers from a list using filter function
```

Out[23]:
```
[2, 4, 6, 8, 10]
```

Map Function

In [24]:
```
print("squares of numbers from 1 to 10 using map function")
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

squares = map(lambda x: x * x, numbers)  # Using map to calculate squares
list(squares)  # Convert the map object to a list to see
```

```
squares of numbers from 1 to 10 using map function
```

Out[24]:
```
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

In [ ]:
```
fname_list = ["Ashwin ", "Divya ", "John "]
lname_list = ["sharma", "shetty", "thomas"]

concat_function = lambda fname, lname: fname + lname  # Lambda function to conca
full_names = map(concat_function, fname_list, lname_list)  # Using map to concat
list(full_names)  # Convert the map object to a list to see the results
```

Out[ ]:
```
['Ashwin sharma', 'Divya shetty', 'Johnthomas']
```

In [26]:
```
name_list = ["Ashwin", "Divya", "John"]
nested_list=map(list, name_list)  # Using map to convert each name to a list
list(nested_list)  # Convert the map object to a list to see the results
```

Out[26]:
```
[['A', 's', 'h', 'w', 'i', 'n'],
 ['D', 'i', 'v', 'y', 'a'],
 ['J', 'o', 'h', 'n']]
```

Module

Functools Module

Reduce Function

In [27]:
```
import functools
print("sum of numbers from 1 to 10 using reduce function")
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
sum_of_numbers = functools.reduce(lambda x, y: x + y, numbers)  # Using reduce t
print("The sum of numbers from 1 to 10 is:", sum_of_numbers)
```

```
sum of numbers from 1 to 10 using reduce function
The sum of numbers from 1 to 10 is: 55
```

Datetime Module

In [2]:
```
import datetime as dt
```

In [3]:
```
date_var=dt.date(2005,12,3)
```

In [4]:
```
date_var
```

Out[4]:
```
datetime.date(2005, 12, 3)
```

In [5]:
```
date_var.month
```

Out[5]:
```
12
```

In [6]:
```
date_var.year
```

Out[6]:
```
2005
```

In [7]:
```
date_var.day
```

Out[7]:
```
3
```

In [10]:
```
from datetime import date # from datetime module import date class
```

In [13]:
```
todays_date=date.today()
todays_date
```

Out[13]:
```
datetime.date(2025, 8, 13)
```

In [14]:
```
from datetime import datetime
```

In [18]:
```
date_time=datetime.today()
date_time
```

Out[18]:
```
datetime.datetime(2025, 8, 13, 15, 27, 48, 990798)
```

In [19]:
```
from datetime import timedelta
```

In [20]:
```
todays_date
```

Out[20]:
```
datetime.date(2025, 8, 13)
```

In [26]:
```
after_2yrs_current_date=todays_date+timedelta(730)
after_2yrs_current_date
```

Out[26]:
```
datetime.date(2027, 8, 13)
```

In [ ]: