Introduction to Control Flow in Python

Control flow is essential in Python (and any programming language) because it allows your program to make decisions and execute different code paths depending on conditions.

Without control flow, every program would just run line by line from top to bottom without any logic or flexibility.

✅ Types of Control Flow in Python:

Conditional Statements: `if`, `elif`, `else`

Loops: `for`, `while`

Loop Control: `break`, `continue`, `pass`

1 Conditional Statements: `if-elif-else`

The first type of control flow satement is `if-elif-else`.

if-elif-else helps your program:

- Make decisions
- Handle multiple conditions
- Provide a fallback using else if nothing matches

This statement helps with conditional execution of code, i.e., the piece of code to be executed is selected based on certain condition(s).

conditions includes

1.1 Comparison operators

- For testing if conditions are true or false, first we need to learn the operators that can be used for comparison.
- Some common python comparison operators are shown in below example

1.2 Logical operators

- Sometimes we may need to check multiple conditions simultaneously.
- The logical operator and is used to check if all the conditions are true, while the logical operator or is used to check if either of the conditons is true.

```
"""
    **************************** Comparison Operators
```

```
******************************
"""

# x == y        Produce True if … x is equal to y
# x != y        … x is not equal to y
# x > y … x is greater than y
# x < y … x is less than y
# x >= y        … x is greater than or equal to y
# x <= y        … x is less than or equal to y

"""
   ****************************** Logical operators
******************************
"""
#Checking if both the conditions are true using 'and'
#Checking if either condition is true using 'or'

if condition1:
    # Code runs if condition1 is True
elif condition2:
    # Code runs if condition1 is False and condition2 is True
elif condition3:
    # (Optional) Code runs if above conditions are False and this one is
True
else:
    # Code runs if none of the above conditions are True
```

In [ ]:
```
## Exercise Problem 1: write a code in python which will take age as input and tell

age = int(input("Enter your age: "))

if age >= 18:
    print("You can vote!")
elif age > 0:
    print("You are too young to vote.")
else:
    print("Invalid age entered.")
```

In [ ]:
```
## Exercise Problem 2: Write a program in Python which takes a year as input from u

"""
# Leap year conditions:

->Divisible by 4
    -> divisible by 100
        -> Divisible by 400 (Leap Year)
        -> Not divisible by 400 (Not a leap year)
    -> Not divisible by 100 (Leap Year)
->Not Divisible by 4 (Not a leap year)

"""
year = int(input("Enter a year: "))
```

```
if (year % 4 == 0):
    if (year % 100 == 0):
        if (year % 400 == 0):
            print(f"{year} is a leap year.")
        else:
            print(f"{year} is not a leap year.")
    else:
        print(f"{year} is a leap year.")
else:
    print(f"{year} is not a leap year.")
```

2. Loops: `for`, `while`

In [ ]:
```
## Exercise Problem 1: Write a program in Python which takes a number as input from

# using for loop -------------
num = int(input("Enter a number: "))

if num <= 1:
    print(f"{num} is not a prime number.")
else:
    for i in range(2, num):
        if num % i == 0:
            print(f"{num} is not a prime number.")
            break
    else:
        print(f"{num} is a prime number.")


# using while loop -------------

num = int(input("Enter a number: "))

if num <= 1:
    print(f"{num} is not a prime number.")
else:
    i = 2
    is_prime = True

    while i < num:
        if num % i == 0:
            is_prime = False
            break
        i += 1

    if is_prime:
        print(f"{num} is a prime number.")
    else:
        print(f"{num} is not a prime number.")
```

```
Enter a number: 5
5 is a prime number.
```

In [ ]:
```
# Exercise Problem 2: Given a list of unique numbers and a target sum, print all pa

arr = [1,2,3,4,5,6,7,8]
target = 5

n = len(arr)
is_found = False

for i in range(n):
  for j in range(i+1,n):
    if arr[i]+arr[j] ==  target:
      is_found = True
      print(arr[i],arr[j])


if is_found == False:
  print("No pair found")
```

```
1 4
2 3
```

3. Loop Control: `break`, `continue`, `pass`

In [ ]:
```
# Break Example 1: Given a list of numbers and a target, find the first occurence o

arr = [5,3,6,5,3,2,1]
target = 3

n = len(arr)
idx = -1

for i in range(n):
  if arr[i] ==  target:
      is_found = True
      idx = i
      break


if is_found == True:
  print(i)
else:
  print("No pair found")
```

```
1
```

In [ ]:
```
# Continue Example 1: given n, print all natural numbers less than equal to n which

n = 25

for i in range(1,n+1):
  if i%5 != 0:
    continue

  print(i)
```

```
5
10
15
20
25
```

Pass: When you're writing code but haven't implemented certain parts yet, you can use pass as a placeholder to avoid syntax errors.

In [ ]:
```python
# Pass Example:
def process_data():
    pass  # You haven't written the logic yet, but want the code to run

# Without pass, Python will raise an IndentationError because the function body is
```

Hands-On Examples

In [ ]:
```python
# if-elif-else statement- control flow

# taken age as input from user and tell whether the user can vote ?

age = int(input("Enter your age"))

if age>= 18:
    print("Congratulations !! you are eligible to vote")
elif age<0:
    print("Age can't be negative. please enter a valid age")
else:
    print("Ohhh !! you are not eligible as per your age")
```

In [10]:
```python
# for loop
# Ex: check whether there is any voter with age tage
tage = 23
age_list = [51,23,35,23,70]

found = 0
for age in age_list:
    if age == tage:
        found = 1
        break

if found == 1:
    print("There is a voter with target age")
else:
    print("no voter found")
```

```
There is a voter with target age
```

In [11]:
```python
# while loop
# Ex: check whether there is any voter with age tage
tage = 23
age_list = [51,23,35,23,70]
```

```
found = 0
n = len(age_list)
for i in range(n):
    age = age_list[i]
    if age == tage:
        found = 1
        break

if found == 1:
    print("There is a voter with target age")
else:
    print("no voter found")
```

```
There is a voter with target age
```

In [ ]:
```
# break
# Ex: check whether there is any voter with age tage
tage = 10
age_list = [51,23,35,23,70]

found = 0
for age in age_list:
    if age == tage:
        found = 1
        break

if found == 1:
    print("There is a voter with target age")
else:
    print("no voter found")
```

In [1]:
```
# continue
# Ex: find the average age of all the eligible voters from list of applicants
tage = 10
age_list = [15,23,35,23,70,15]

avgAge = 0
cnt = 0
for age in age_list:
    if age < 18:
        continue
    avgAge += age
    cnt += 1

if cnt:
    avgAge = avgAge/cnt

print(f'avg age is {avgAge} years')
```

```
avg age is 37.75 years
```

In [8]:
```
# pass

def calcSumAgeHelper():
    pass

def calcAvgAge():
```

```
    avgAge = 0
    calcSumAgeHelper()
    # some coding logic written
    print(avgAge)


calcAvgAge()
```

```
0
```

In [ ]: