Printing Messages in Python

In [ ]:
```
# Output the message

print("Hello, World!")
print("Python is fun!")

#Multiple Values
print("My name is", "Alice", "and I am", 21, "years old.")


#using end =" "
print("Hello", end=" ")
print("World!")

#Special Characters
print("Hello\nWorld!")  # Newline character
print("Hello\tWorld!")  # Tab character
print("Hello \"World\"!")  # Double quotes
print("Hello \\ World!")  # Backslash

#formatted Strings
name = "Alice"
age = 21
print(f"My name is {name} and I am {age} years old.")

#String Concatenation
print("Hello " + "World!")

#Multiline print statements
print("""This is a multiline
print statement.
It spans multiple lines.""")
```

Data types and Variables

In [ ]:
```
#variables and data types

age = 21
price = 19.99
name = "Alice"
is_student = True
print(age, price, name, is_student)


#Boolean Operations

is_sunny = True
is_warm = False
print(is_sunny and is_warm)  # Logical AND
print(is_sunny or is_warm)   # Logical OR
print(not is_sunny)          # Logical NOT
```

Taking User Input

In [ ]:
```
#Taking User Input

name = input("Enter your name: ")
age = int(input("Enter your age: "))
print("Hello", name, "you are", age, "years old.")


#Basic Arithmetic operations

a = 10
b = 3
print("Addition:", a + b)
print("Subtraction:", a - b)
print("Multiplication:", a * b)
print("Division:", a / b)
print("Modulus:", a % b)
print("Exponentiation:", a ** b)
```

Conditional Statements

In [ ]:

```python
#Conditional Statements


age = int(input("Enter your age: "))
if age < 18:
    print("You are a minor.")
elif age < 65:
    print("You are an adult.")
else:
    print("You are a senior.")






#Nested Ifs

num = 5
if num > 0:
    if num % 2 == 0:
        print("Positive even number")
    else:
        print("Positive odd number")
else:
    print("Non-positive number")


#Comparing Values

x = 10
y = 20
if x < y:
    print("x is less than y")
elif x == y:
    print("x is equal to y")
else:
    print("x is greater than y")
```

In [10]:
```
#Grading System

marks = int(input("Enter your Marks"))

if marks > 100:
    print("Invalid Marks ")
elif marks >= 90 and marks <= 100 :
    print("Your Grade is A+ ")
elif marks >= 80 and marks < 90:
    print("Your Grade is A ")
elif marks >=70  and marks < 80:
    print("Your Grade is B ")
elif marks >=60  and marks < 70:
    print("Your Grade is C ")
else:
    print("You Have Failed")
```

```
Enter your Marks101
Invalid Marks
```

In [ ]:
```
```

For and while loops

In [ ]:

```python
#For loops

for i in range(5):
    print(i)



for i in range(2, 6):
    print(i)


for i in range(0, 10, 2):
    print(i)


for char in "Hello":
    print(char)


names = ["Alice", "Bob", "Charlie"]
for name in names:
    print(name)

for i in range(3):
    for j in range(2):
        print(f"i={i}, j={j}")


fruits = ["apple", "banana", "cherry"]
for index, fruit in enumerate(fruits):
    print(f"Index {index}: {fruit}")

#2 x 1 = 2
```

In [17]:

```python
var = int(input("Enter table number "))
for i in range(1,11):
    print(var,' x ',i, " = ", i*var)
```

```
Enter table number 5
5  x  1  =  5
5  x  2  =  10
5  x  3  =  15
5  x  4  =  20
5  x  5  =  25
5  x  6  =  30
5  x  7  =  35
5  x  8  =  40
5  x  9  =  45
5  x  10  =  50
```

In [ ]:

In [ ]:
```
#While loops

count = 0
while count < 5:
    print(count)
    count += 1


num = 10
while num > 0:
    print(num)
    num -= 2


while True:
    response = input("Type 'exit' to stop: ")
    if response == 'exit':
        break
    print("You typed:", response)


count = 0
while count < 5:
    print(count)
    count += 1
else:
print("Loop finished")
```

In [20]:
```
pip install pandoc
```

```
Could not fetch URL https://pypi.org/simple/pandoc/: (https://pypi.org/simpl
e/pandoc/:) There was a problem confirming the ssl certificate: HTTPSConnecti
onPool(host='pypi.org', port=443): Max retries exceeded with url: /simple/pan
doc/ (Caused by SSLError(SSLCertVerificationError(1, '[SSL: CERTIFICATE_VERIF
Y_FAILED] certificate verify failed: unable to get local issuer certificate
(_ssl.c:1129)'))) - skippingNote: you may need to restart the kernel to use u
pdated packages.

WARNING: Retrying (Retry(total=4, connect=None, read=None, redirect=None, sta
tus=None)) after connection broken by 'SSLError(SSLCertVerificationError(1,
'[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: unable to get lo
cal issuer certificate (_ssl.c:1129)'))': /simple/pandoc/


Could not fetch URL https://pypi.org/simple/pip/: (https://pypi.org/simple/pi
p/:) There was a problem confirming the ssl certificate: HTTPSConnectionPool
(host='pypi.org', port=443): Max retries exceeded with url: /simple/pip/ (Cau
sed by SSLError(SSLCertVerificationError(1, '[SSL: CERTIFICATE_VERIFY_FAILED]
certificate verify failed: unable to get local issuer certificate (_ssl.c:112
9)'))) - skipping

WARNING: Retrying (Retry(total=3, connect=None, read=None, redirect=None, sta
tus=None)) after connection broken by 'SSLError(SSLCertVerificationError(1,
'[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: unable to get lo
cal issuer certificate (_ssl.c:1129)'))': /simple/pandoc/
WARNING: Retrying (Retry(total=2, connect=None, read=None, redirect=None, sta
tus=None)) after connection broken by 'SSLError(SSLCertVerificationError(1,
'[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: unable to get lo
cal issuer certificate (_ssl.c:1129)'))': /simple/pandoc/
WARNING: Retrying (Retry(total=1, connect=None, read=None, redirect=None, sta
tus=None)) after connection broken by 'SSLError(SSLCertVerificationError(1,
'[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: unable to get lo
cal issuer certificate (_ssl.c:1129)'))': /simple/pandoc/
WARNING: Retrying (Retry(total=0, connect=None, read=None, redirect=None, sta
tus=None)) after connection broken by 'SSLError(SSLCertVerificationError(1,
'[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: unable to get lo
cal issuer certificate (_ssl.c:1129)'))': /simple/pandoc/
ERROR: Could not find a version that satisfies the requirement pandoc (from v
ersions: none)
ERROR: No matching distribution found for pandoc
```

In [ ]:
```python
#Loop Control Statements

for i in range(10):
    if i == 5:
        break
    print(i)

for i in range(10):
    if i % 2 == 0:
        continue
    print(i)


for i in range(5):
    if i == 3:
        pass
    print(i)
```

In [ ]:
```python
#Creating Lists

fruits = ["apple", "banana", "cherry"]
print(fruits[0])
fruits.append("orange")
print(fruits)
```

Data Structures: List

In [ ]:
```
# Creating a List
fruits = ["apple", "banana", "cherry"]
print(fruits)

# Accessing List Elements
print(fruits[0])  # First element
print(fruits[-1]) # Last element

# Modifying List Elements
fruits[1] = "blueberry"
print(fruits)

# Adding Elements to a List
fruits.append("orange")
print(fruits)

# Inserting Elements into a List
fruits.insert(1, "blueberry")
print(fruits)

# Removing Elements from a List
fruits.remove("banana")
print(fruits)

# Popping Elements from a List
last_fruit = fruits.pop()
print(fruits)
print(last_fruit)

# Slicing a List
fruits = ["apple", "banana", "cherry", "date", "elderberry"]
print(fruits[1:3])  # ['banana', 'cherry']
print(fruits[:2])   # ['apple', 'banana']
print(fruits[2:])   # ['cherry', 'date', 'elderberry']

# Looping Through a List
fruits = ["apple", "banana", "cherry"]
for fruit in fruits:
    print(fruit)

# List Comprehension
numbers = [1, 2, 3, 4, 5]
squares = [num ** 2 for num in numbers]
print(squares)
```

Data Structures: Tuples

In [ ]:
```
# Creating a Tuple
fruits = ("apple", "banana", "cherry")
print(fruits)

# Accessing Tuple Elements
print(fruits[0])  # First element
print(fruits[-1]) # Last element

# Immutable Nature of Tuples
# fruits[1] = "blueberry"  # This will raise an error

# Tuple with One Element
single_element_tuple = ("apple",)
print(single_element_tuple)

# Unpacking Tuples
fruits = ("apple", "banana", "cherry")
(first, second, third) = fruits
print(first, second, third)

# Using Tuples as Keys in Dictionaries
coordinates = {(0, 0): "Origin", (1, 1): "Point A"}
print(coordinates)
print(coordinates[(1, 1)])
```

Data Structures: Dictionary

In [ ]:
```
# Creating a Dictionary
student = {"name": "Alice", "age": 21, "major": "CS"}
print(student)

# Accessing Dictionary Elements
print(student["name"])

# Modifying Dictionary Elements
student["age"] = 22
print(student)

# Adding Elements to a Dictionary
student["grade"] = "A"
print(student)

# Removing Elements from a Dictionary
del student["major"]
print(student)

# Dictionary Methods
print(student.keys())
print(student.values())
print(student.items())

# Looping Through a Dictionary
for key, value in student.items():
    print(f"{key}: {value}")

# Checking if Key Exists in Dictionary
if "age" in student:
    print("Age is a key in the student dictionary.")
```

Data Structures: Sets

In [ ]:

```
# Creating a Set
fruits = {"apple", "banana", "cherry"}
print(fruits)

# Adding Elements to a Set
fruits.add("orange")
print(fruits)

# Removing Elements from a Set
fruits.remove("banana")
print(fruits)

# Set Methods
tropical_fruits = {"banana", "mango", "papaya"}

# Union
all_fruits = fruits.union(tropical_fruits)
print(all_fruits)

# Intersection
common_fruits = fruits.intersection(tropical_fruits)
print(common_fruits)

# Difference
unique_fruits = fruits.difference(tropical_fruits)
print(unique_fruits)

# Checking Membership in a Set
print("banana" in fruits)
print("grape" in fruits)

# Looping Through a Set
for fruit in fruits:
    print(fruit)
```