ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

DAY – 8 **2 July 2025**

Strings

Strings in python are surrounded by either single quotation marks, or double quotation marks.

'hello' is the same as "hello".

You can display a string literal with the print() function:

```
a = "Hello"
print(a)
```

Slicing

You can return a range of characters by using the slice syntax.

Specify the start index and the end index, separated by a colon, to return a part of the string.

Get the characters from position 2 to position 5 (not included):

```
b = "Hello, World!"
print(b[2:5])
```

Python has a set of built-in methods that you can use on strings.

The upper() method returns the string in upper case:

```
a = "Hello, World!"
print(a.upper())
```

The lower() method returns the string in lower case:

```
a = "Hello, World!"
print(a.lower())
```

The strip() method removes any whitespace from the beginning or the end:

```
a = " Hello, World! "
print(a.strip()) # returns "Hello, World!"
```

By : Aditi Tangri URN : 2302460 CRN : 2315004

The split() method splits the string into substrings if it finds instances of the separator:

```
a = "Hello, World!"
print(a.split(",")) # returns ['Hello', ' World!']
```

String Concatenation

To concatenate, or combine, two strings you can use the + operator.

Merge variable `a` with variable `b` into variable `c`:

```
a = "Hello"
b = "World"
c = a + b
print(c)
```

Boolean Values

Booleans represent one of two values: True or False.

```
print(10 > 9)
print(10 == 9)
print(10 < 9)
```

```
True
False
False
```

Python Operators

Operators are used to perform operations on variables and values.

Python divides the operators in the following groups:

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Identity operators
- Membership operators
- Bitwise operators

Operator Precedence

Operator precedence describes the order in which operations are performed.

Parentheses has the highest precedence, meaning that expressions inside parentheses must be evaluated first:

```
print((6 + 3) - (6 + 3))
```

List

Lists are used to store multiple items in a single variable.

Lists are one of 4 built-in data types in Python used to store collections of data, the other 3 are Tuple, Set and Dictionary all with different qualities and usage.

Lists are created using square brackets:

```
thislist = ["apple", "banana", "cherry"]
print(thislist)
```

List Items

List items are ordered, changeable, and allow duplicate values.

List items are indexed, the first item has index [0], the second item has index [1] etc.

List Length

To determine how many items a list has, use the len() function:

```
thislist = ["apple", "banana", "cherry"]
print(len(thislist))
```

Access Items

List items are indexed and you can access them by referring to the index number:

```
thislist = ["apple", "banana", "cherry"]
print(thislist[1])
```

Change Item Value

To change the value of a specific item, refer to the index number:

```
thislist = ["apple", "banana", "cherry"]
thislist[1] = "blackcurrant"
print(thislist)
```

Append Items

To add an item to the end of the list, use the append() method:

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

Remove Specified Item

The remove() method removes the specified item.

```
thislist = ["apple", "banana", "cherry"]
thislist.remove("banana")
print(thislist)
```

Tuple

Tuples are used to store multiple items in a single variable.

A tuple is a collection which is ordered and unchangeable.

Tuples are written with round brackets.

```
thistuple = ("apple", "banana", "cherry")
print(thistuple)
```

Tuple Items

Tuple items are ordered, unchangeable, and allow duplicate values.

Tuple items are indexed, the first item has index [0], the second item has index [1] etc**.**

Set

Sets are used to store multiple items in a single variable.

A set is a collection which is *unordered*, *unchangeable*, and *unindexed*.

```
thisset = {"apple", "banana", "cherry"}
print(thisset)
```

Dictionary

Dictionaries are used to store data values in key:value pairs.

A dictionary is a collection which is ordered*, changeable and do not allow duplicates.

Dictionaries are written with curly brackets, and have keys and values:

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang"
  "year": 1964
}
print(thisdict)
```

Python Conditions and If statements

Python supports the usual logical conditions from mathematics:

- Equals: a == b
- Not Equals: a != b
- Less than: a < b
- Less than or equal to: a <= b
- Greater than: a > b
- Greater than or equal to: a >= b

These conditions can be used in several ways, most commonly in "if statements" and loops.

An "if statement" is written by using the if keyword.

```
a = 33
b = 200
if b > a:
  print("b is greater than a")
```

The Python Match Statement

Instead of writing many if..else statements, you can use the match statement.

The match statement selects one of many code blocks to be executed.

```
match expression:
  case x:
    code block
  case y:
    code block
  case z:
    code block
```

The while Loop

With the while loop we can execute a set of statements as long as a condition is true.

```
i = 1
while i < 6:
  print(i)
  i += 1
```

Python For Loops

A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).This is less like the for keyword in other programming languages, and works more like an iterator method as found in other object-orientated programming languages.

With the for loop we can execute a set of statements, once for each item in a list, tuple, set etc.

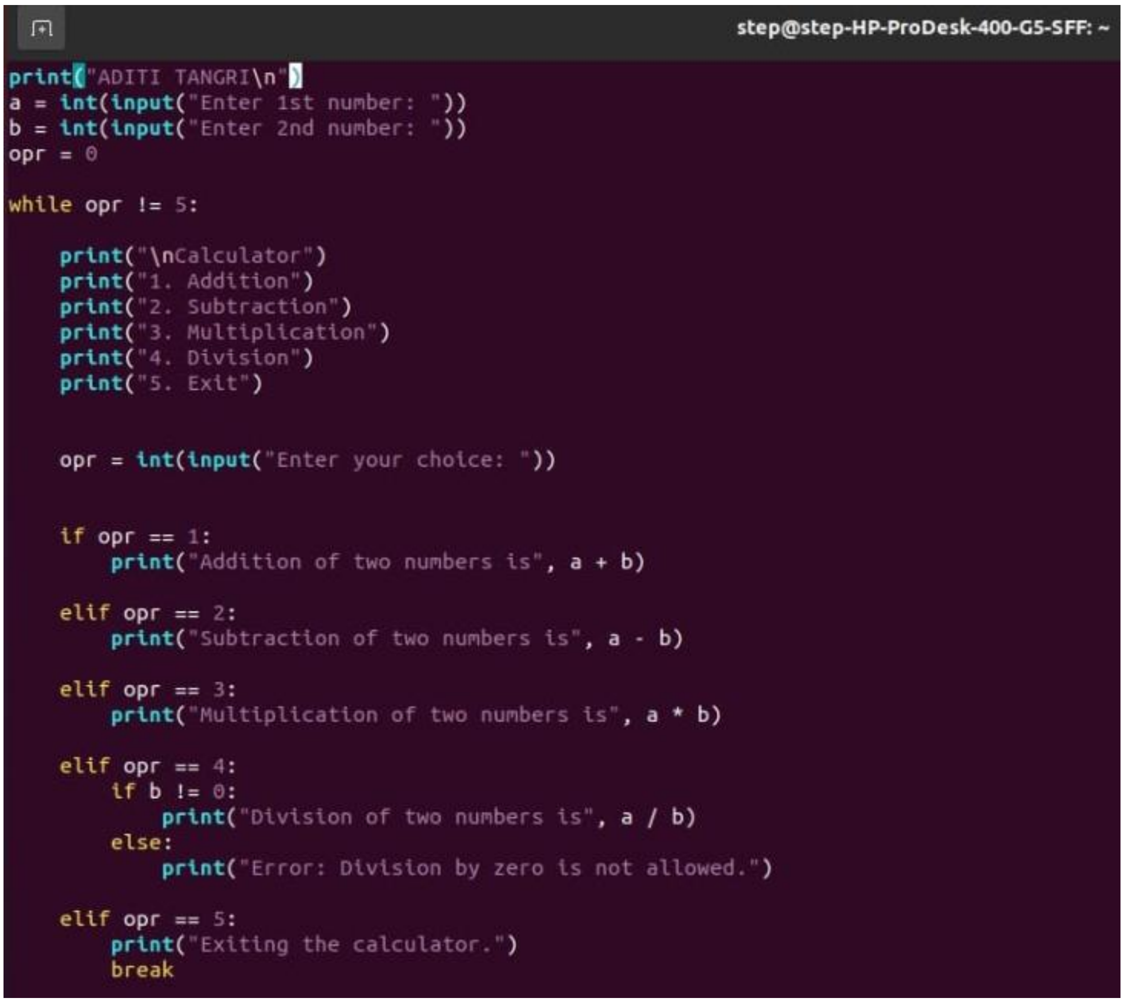
```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
```

Python Functions

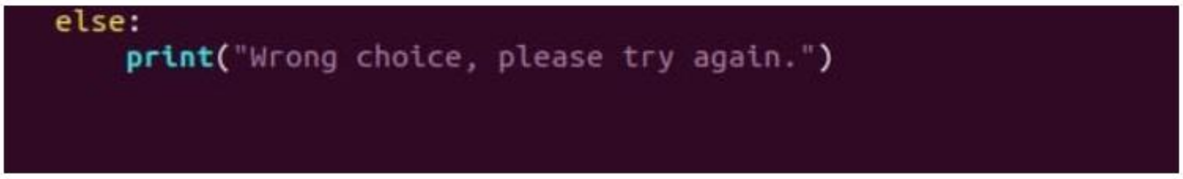
A function is a block of code which only runs when it is called.You can pass data, known as parameters, into a function.A function can return data as a result.

```
def my_function():
  print("Hello from a function")
```

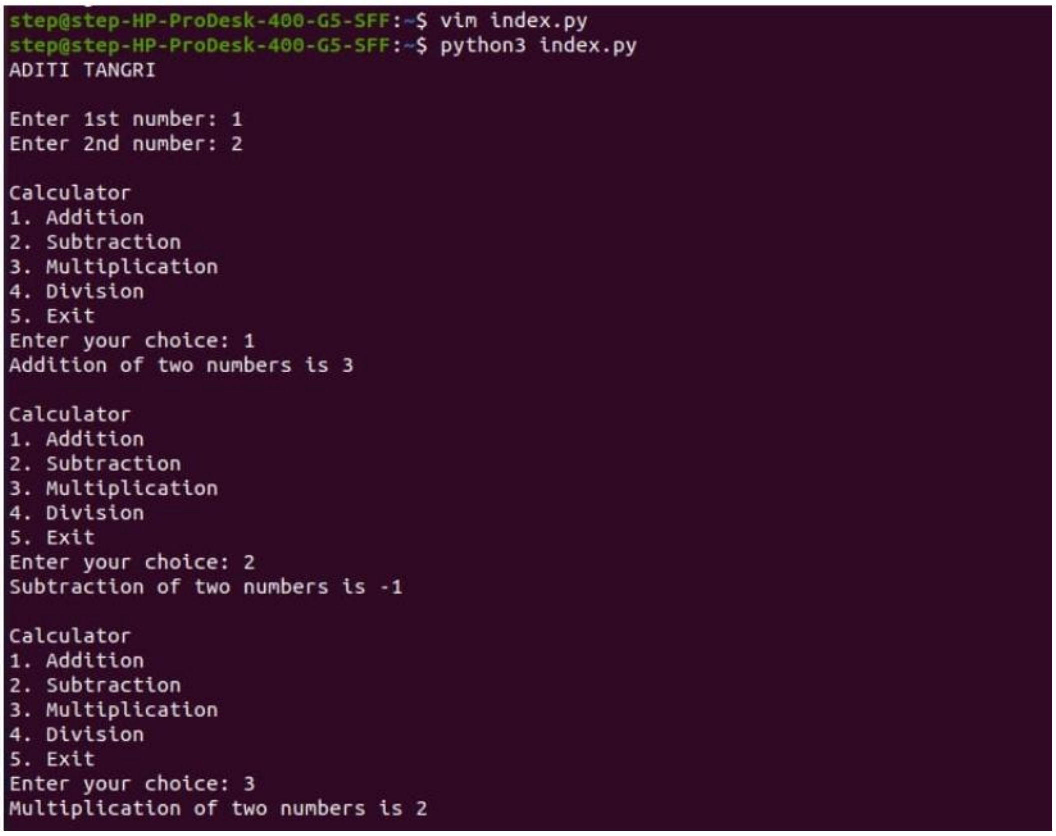
Simple Calculator Program in Python that performs basic arithmetic operations: addition, subtraction, multiplication, and division.



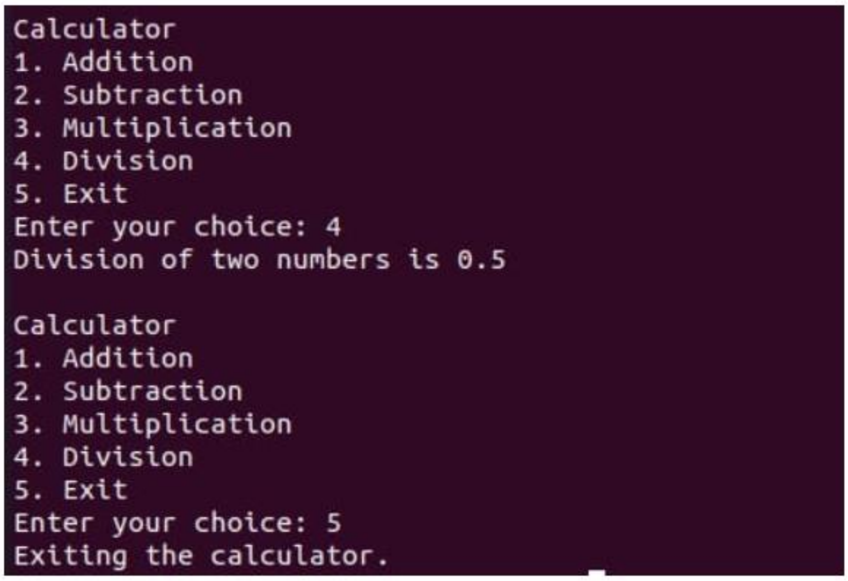
```
step@step-HP-ProDesk-400-G5-SFF: ~
print("ADITI TANGRI\n")
a = int(input("Enter 1st number: "))
b = int(input("Enter 2nd number: "))
opr = 0

while opr != 5:

    print("\nCalculator")
    print("1. Addition")
    print("2. Subtraction")
    print("3. Multiplication")
    print("4. Division")
    print("5. Exit")


    opr = int(input("Enter your choice: "))


    if opr == 1:
        print("Addition of two numbers is", a + b)

    elif opr == 2:
        print("Subtraction of two numbers is", a - b)

    elif opr == 3:
        print("Multiplication of two numbers is", a * b)

    elif opr == 4:
        if b != 0:
            print("Division of two numbers is", a / b)
        else:
            print("Error: Division by zero is not allowed.")

    elif opr == 5:
        print("Exiting the calculator.")
        break
```



```
    else:
        print("Wrong choice, please try again.")
```



```
step@step-HP-ProDesk-400-G5-SFF:~$ vim index.py
step@step-HP-ProDesk-400-G5-SFF:~$ python3 index.py
ADITI TANGRI

Enter 1st number: 1
Enter 2nd number: 2

Calculator
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit
Enter your choice: 1
Addition of two numbers is 3

Calculator
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit
Enter your choice: 2
Subtraction of two numbers is -1

Calculator
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit
Enter your choice: 3
Multiplication of two numbers is 2
```



```
Calculator
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit
Enter your choice: 4
Division of two numbers is 0.5

Calculator
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit
Enter your choice: 5
Exiting the calculator.
```