- **USER INPUT**

```
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
print(num1 + num2)
```

```
# Integer example
num1 = int(input("Enter the first integer: "))
num2 = int(input("Enter the second integer: "))

# Performing addition
result = num1 + num2

# Displaying the result
print(f"The sum of {num1} and {num2} is {result}.")
```

```
# Complex number example
num1 = complex(input("Enter the first complex number (e.g., 3+4j): "))
num2 = complex(input("Enter the second complex number (e.g., 1+2j): "))

# Performing addition
result = num1 + num2

# Displaying the result
print(f"The sum of {num1} and {num2} is {result}.")
```

- **Boolean Values**

```
# Taking user input
num1 = int(input("Enter a number: "))

# Printing the result directly
print(f"Is the number positive? {num1 > 0}")
```

```
# Taking user input
num1 = int(input("Enter a number: "))

# Checking if the number is positive (True) or not (False)
is_positive = num1 > 0

# Displaying the result
print(f"Is the number positive? {is_positive}")
```

Example 1: Check if a number is even

```
num = int(input("Enter a number: "))
print(f"Is the number even? {num % 2 == 0}")
```

Example 2: Check if a number is divisible by 5

```
num = int(input("Enter a number: "))
print(f"Is the number divisible by 5? {num % 5 == 0}")
```

Example 3: Check if two numbers are equal

```
num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))
print(f"Are the two numbers equal? {num1 == num2}")
```

Example 4: Check if a string contains only alphabets

```
text = input("Enter a string: ")
print(f"Does the string contain only alphabets?
{text.isalpha()}")
```

Example 5: Check if a number is negative

```
num = int(input("Enter a number: "))
print(f"Is the number negative? {num < 0}")
```

Example 6: Check if a string is a palindrome

```
text = input("Enter a string: ")
print(f"Is the string a palindrome? {text == text[::-1]}")
```

Example 7: Check if a number is greater than another

```
num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))
print(f"Is the first number greater than the second? {num1 >
num2}")
```

Example 8: Check if a number is within a range

```
num = int(input("Enter a number: "))
print(f"Is the number between 1 and 10? {1 <= num <= 10}")
```

Example 9: Check if a string starts with a specific letter

```
text = input("Enter a string: ")
print(f"Does the string start with 'A'? {text.startswith('A')}")
```

- **Lists**

```
mylist = ["apple", "banana", "cherry"]
```

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.

Ordered

When we say that lists are ordered, the items have a defined order, and that order will not change.

If you add new items to a list, the new items will be placed at the end of the list.

Changeable

The list is changeable, meaning we can change, add, and remove items in a list after it has been created.

Allow Duplicates

Since lists are indexed, lists can have items with the same value:

Lists allow duplicate values:

```
thislist = ["apple", "banana", "cherry", "apple", "cherry"]

print(thislist)
```

List Length

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

```
thislist = ["apple", "banana", "cherry"]

print(len(thislist))
```

List Items - Data Types

List items can be of any data type:

```
list1 = ["apple", "banana", "cherry"]

list2 = [1, 5, 7, 9, 3]

list3 = [True, False, False]
```

A list can contain different data types:

```
list1 = ["abc", 34, True, 40, "male"]
```

type()

From Python's perspective, lists are defined as objects with the data type 'list':

```
mylist = ["apple", "banana", "cherry"]

print(type(mylist))
```

The list() Constructor

Using the `list()` constructor when creating a new list is also possible.

```
thislist = list(("apple", "banana", "cherry")) # note the
double round-brackets

print(thislist)
```

Python - Access List Items

Access Items

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

```
thislist = ["apple", "banana", "cherry"]

print(thislist[1])
```

Negative Indexing

Negative indexing means start from the end

`-1` refers to the last item, `-2` refers to the second last item etc.

```
thislist = ["apple", "banana", "cherry"]


print(thislist[-1])
```

Range of Indexes

You can specify a range of indexes by specifying where to start and where to end the range.

When specifying a range, the return value will be a new list with the specified items.

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi",
"melon", "mango"]

print(thislist[2:5])
```

This example returns the items from the beginning to, but NOT including, "kiwi":

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi",
"melon", "mango"]

print(thislist[:4])
```

By leaving out the end value, the range will go on to the end of the list:

This example returns the items from "cherry" to the end:

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi",
"melon", "mango"]

print(thislist[2:])
```

Range of Negative Indexes

Specify negative indexes if you want to start the search from the end of the list:

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi",
"melon", "mango"]

print(thislist[-4:-1])
```

Check if Item Exists

To determine if a specified item is present in a list use the `in` keyword:

```
thislist = ["apple", "banana", "cherry"]

if "apple" in thislist:

 print("Yes, 'apple' is in the fruits list")
```

- **range**

```
# Printing numbers from 1 to 10 using range

for num in range(1, 11):

    print(num)
```