

Experiment - 1.1

AIM: Write a program that asks the user for a weight in kilograms and converts it to pounds. There are 2.2 pounds in a kilogram.

Description: This program is a simple unit conversion tool designed to convert a weight value from kilograms to pounds. The conversion is based on the standard conversion factor where 1 kilogram is equal to 2.2 pounds. The program takes a numerical input from the user, performs the calculation, and then outputs the result in a clear, user-friendly sentence.

Algorithm

1. **Start** the program.
2. **Declare** a variable, kilograms, to store the weight provided by the user.
3. **Prompt** the user to enter a weight value in kilograms.
4. **Read** the user's input and **store** it as a floating-point number in the kilograms variable.
5. **Declare** a new variable, pounds.
6. **Calculate** the weight in pounds by multiplying the value in kilograms by the conversion factor, 2.2.
7. **Store** the result of the calculation in the pounds variable.
8. **Display** the output in a formatted string, showing both the original weight in kilograms and the converted weight in pounds.
9. **End** the program.

Program:

```
kilograms = float(input("Enter weight in kilograms: "))  
pounds = kilograms * 2.2  
print(f"{kilograms} kilograms is equal to {pounds} pounds.")
```

Output:

```
Python 3.13.7 (tags/v3.13.7:bcee1c3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (AMD64)] on win32  
Enter "help" below or click "Help" above for more information.  
>  
===== RESTART: C:/Users/krish/Desktop/python lab/1.1.py =====  
Enter weight in kilograms: 20  
20.0 kilograms is equal to 44.0 pounds.  
>|
```

Experiment - 1.2

AIM: Write a program that uses a for loop to print the numbers 8, 11, 14, 17, 20, . . . , 83, 86, 89.

Description: This program uses a **for loop** to generate and display a sequence of numbers. The sequence starts at 8, ends just before 90, and increases by a step of 3 for each number. This allows the program to efficiently print all the numbers in the pattern 8, 11, 14, 17, and so on, up to 89, all on a single line. The program also uses the `end=' '` argument within the `print` function to ensure the numbers are separated by a comma and a space, and the final `print()` function creates a new line to maintain a clean output.

Algorithm

1. **Start** the program.
2. **Initialize** a for loop that iterates through a sequence of numbers.
3. **Specify** the range for the loop:
 - The starting number is 8.
 - The ending number is 90 (the loop will stop just before reaching 90, so 89 will be the last number printed).
 - The step or increment for each iteration is 3.
4. **Iterate** through the specified range. For each number in the sequence:
 - **Print** the current number.
 - **Add** a comma and a space after each number instead of a new line.
5. **After** the loop has finished, **print** a new line to ensure any subsequent output appears on a new line.
6. **End** the program.

Program:

```
for num in range(8, 90, 3):  
    print(num, end=' ', )  
print()
```

Output:

```
>>> |===== RESTART: C:/Users/krish/Desktop/python lab/1.2.py =====  
8, 11, 14, 17, 20, 23, 26, 29, 32, 35, 38, 41, 44, 47, 50, 53, 56, 59, 62, 65, 68, 71, 74, 77, 80, 83, 86, 89,  
>>> |
```

Experiment - 1.3

AIM: Write a program that Split a string into array of characters in Python.

Description:

This program demonstrates a straightforward method for converting a given string into a list (or array) of its individual characters in Python. The program takes a string, such as "Python Programming is very very easy!", and uses the built-in list() function to perform the conversion. The list() function iterates through the string and treats each character, including spaces and punctuation, as a separate element, which is then stored in a new list. Finally, the program prints the resulting list of characters.

Algorithm

1. Start the program.
2. Declare a string variable, input_string, and assign it a value (e.g., "Python Programming is very very easy!").
3. Declare a new variable, char_array.
4. Convert the input_string into a list of its individual characters by passing it as an argument to the built-in list() function.
5. Assign the resulting list of characters to the char_array variable.
6. Print a descriptive message followed by the contents of the char_array to display the output.
7. End the program.

Program:

```
input_string = "This is pythin program"
char_array = list(input_string)
print("Array of characters:", char_array)
```

OUTPUT:

```
>>> |===== RESTART: C:/Users/krish/Desktop/python lab/1.3.py =====
    |Array of characters: ['T', 'h', 'i', 's', ' ', 'i', 's', ' ', 'p', 'y', 't', 'h', 'i', 'n', ' ', 'p', 'r', 'o', 'g', 'r', 'a', 'm']
>>> |
```

Experiment - 1.4

AIM: write a Python program to get the largest number from a list.

Description:

This program is designed to find the largest number within a list provided by the user. It prompts the user to enter a list of numbers as a string, which it then converts into a Python list object using the `eval()` function. The program's core functionality relies on Python's built-in `max()` function, a highly efficient tool for identifying the maximum value within a list or any iterable. After finding the largest number, the program stores it and displays a clear message showing the final result.

Algorithm

1. Start the program.
2. Prompt the user to enter a list of numbers.
3. Read the user's input string.
4. Convert the input string into a list of numbers using the `eval()` function and store it in a variable called `numbers`.
5. Use the `max()` function on the `numbers` list to find the largest number.
6. Store the largest number found in a new variable, `largest`.
7. Print a message that includes the value of the largest variable.
8. End the program.

PROGRAM:

```
numbers = eval(input("Enter list:"))  
  
largest = max(numbers)  
  
print("The largest number in the list is:", largest)
```

OUTPUT:

```
>>> |  
===== RESTART: C:/Users/krish/Desktop/python lab/1.4.py =====  
Enter list:[12,53,670,76,879,45,79]  
The largest number in the list is: 879
```

Experiment - 1.5

AIM: write a Python program to get the largest number from a list.

Description:

This program is designed to calculate the nth Fibonacci number, where n is a position in the sequence provided by the user. The Fibonacci sequence is a series of numbers where each number is the sum of the two preceding ones, starting from 0 and 1. The program defines a function called fibonacci(n) that handles the calculation. It first checks for invalid input, then returns the correct value for the first two positions (1 and 2). For any position greater than 2, it uses an iterative for loop to compute the number efficiently by updating two variables in each step until the desired position is reached. Finally, it prompts the user for a position and prints the corresponding Fibonacci number.

Algorithm

1. Start the program.
2. Define a function named fibonacci that takes one argument, n.
3. Inside the fibonacci function:
 - Check if n is less than or equal to 0. If it is, return the string "Invalid input".
 - Otherwise, check if n is equal to 1 or 2. If it is, return n-1.
 - Otherwise (for n>2):
 - Initialize two variables, a and b, with the values 0 and 1 respectively.
 - Start a for loop that iterates n-2 times.
 - In each iteration, update the values of a and b by simultaneously assigning the current value of b to a, and the sum of the current a and b to b.
 - After the loop, return the value of b.
4. Prompt the user to enter a positive integer for the position.
5. Read the user's input and store it as an integer in the variable n.
6. Call the fibonacci function with n as the argument.
7. Print a descriptive message along with the value returned by the fibonacci function.
8. End the program.

Program:

```
def fibonacci(n):  
    if n <= 0:  
        return "Invalid input"  
    elif n == 1 or n == 2:  
        return n-1  
    else:  
        a, b = 0, 1  
        for _ in range(n - 2):  
            a, b = b, a + b  
        return b  
  
n = int(input("Enter the position (n) to find nth Fibonacci number: "))  
print(f"The {n}th Fibonacci number is:", fibonacci(n))
```

Output:

```
===== RESTART: C:/Users/krish/Desktop/python lab/1.5.py =====  
Enter the position (n) to find nth Fibonacci number: 5  
The 5th Fibonacci number is: 3  
|
```