Swap Two Variables

Aim : To write python program to swap two variables

Algorithm

1. **Start**.
2. **Input**:
    - Prompt the user to enter a value for variable x and store it.
    - Prompt the user to enter a value for variable y and store it.
3. **Swap Variables**:
    - Assign the value of x to temp
    - Assign the value of y to x
    - Assign the value of temp to y
4. **Output**:
    - Print the value of x after swapping.
    - Print the value of y after swapping.
5. **End**.

Program

```
# Python program to swap two variables

x = input('Enter value of x: ')
y = input('Enter value of y: ')

temp = x
x = y
y = temp

print('The value of x after swapping: {}'.format(x))
print('The value of y after swapping: {}'.format(y))
```

Output

```
Enter value of x: 5
Enter value of y: 6
The value of x after swapping: 6
The value of y after swapping: 5
```

Count Vowels in a String

Aim : To write python program to count number of vowels in a given string

Algorithm

1. **Start**.
2. **Input**:
    - Prompt the user to enter a string and store it in a variable st.
3. **Initialize**:
    - Create a variable vowels and assign string "aeiouAEIOU" (all vowels, both lowercase and uppercase).
    - Initialize a counter variable vc to 0. This will count the number of vowels.
4. **Loop**:
    - For each character ch in the string st:
        1. Check if ch is present in the vowels string:
            - If **true**, increment the counter vc by 1.
5. **Output**:
    - Print the total count of vowels using the value of vc.
6. **End**.

Program

```
st = input("Enter a string: ")

vowels = "aeiouAEIOU"

vc = 0

for ch in st:
   if ch in vowels:
      vc =vc + 1

print(f"The number of vowels in the string is: {vc}")
```

Output

```
Enter a string: computer science
The number of vowels in the string is: 6
```

Find Minimum values among N values

Aim : To write python program to find minimum of vales amount n values

Algorithm:

1. **Input the number of variables**:
   - Prompt the user to enter the number of variables, `n`.
   - Store the input in a variable `n`.
2. **Initialize an empty list**:
   - Create an empty list `a` to store the variables.
3. **Input the variables**:
   - Loop from `i = 0` to `n - 1`:
     - In each iteration, prompt the user to input a number (`num`).
     - Convert the input into a floating-point number and append it to the list `a`.
4. **Find the minimum value**:
   - Use the `min()` function to find the smallest value in the list `a`.
5. **Output the result**:
   - Print the minimum value `x` using a formatted string.

Program

```
# Main program
n = int(input("Enter the number of variables: "))

# Initialize an empty list to store the variables
a = []

# Input: variables
for i in range(n):
    num = float(input(f"Enter variable {i+1}: "))
    a.append(num)

# Find and display the minimum value
x = min(a)
print(f"The minimum value is: {x}")
```

Output

```
Enter the number of values: 5
Enter variable 1: 525
Enter variable 2: 256
Enter variable 3: 350
Enter variable 4: 240
Enter variable 5: 580
The minimum value is: 240.0
```

Simple Sorting

Aim : To write python program to sort the given set of N numbers

Algorithm:

1. **Input the number of variables**:
   - Prompt the user to input the number of variables, "N".
   - Store this input in a variable "N".
2. **Initialize an empty list**:
   - Create an empty list "A" to store the input values.
3. **Input the variables**:
   - Loop from i = 0 to N - 1:
     - In each iteration, prompt the user to input a number (NUM).
     - Convert the input into a floating-point number and append it to the list "A".
4. **Sort the list**:
   - Use Python's **sorted()** function to sort the list a in ascending order.
   - Store the result in variable **x**.
5. **Display the sorted list**:
   - Print the sorted list **x** using a formatted string, displaying the numbers in ascending order.

Program

```
# Program for simple sorting of given numbers
n = int(input("Enter the number of variables: "))
# Initialize an empty list
a = []
# Input: variables
for i in range(n):
    num = float(input(f"Enter variable {i+1}: "))
    a.append(num)
# Find and display the minimum value
x=sorted(a)
print(f"The ascending order of given values are: {x}")
```

Output

```
Enter the number of values: 5
Enter variable 1: 45
Enter variable 2: 25
Enter variable 3: 83
Enter variable 4: 32
Enter variable 5: 56
The ascending order of given values are: [25.0, 32.0, 45.0, 56.0, 83.0]
```

Student’s marks statement

Aim : To write python program to generate grade, total and average

Algorithm:

1. **Start**
2. **Define a function grade(m)** that accepts marks m as input and returns a grade:
    - If m >= 80, return 'O'.
    - Else if m >= 71, return 'A+'.
    - Else if m >= 66, return 'A'.
    - Else if m >= 61, return 'B+'.
    - Else if m >= 56, return 'B'.
    - Else if m >= 50, return 'C'.
    - Else if m >= 40, return 'P'.
    - Else, return 'F'.
3. **Prompt the user to input the student's name** and store it in variable name.
4. **Create an empty dictionary marks** to store the marks for 5 subjects.
5. **For each subject (5 subjects)**:
    - Ask the user to input marks for the subject.
    - Store the marks in the marks dictionary.
6. **Calculate the total marks** by summing all the values in the marks dictionary.
7. **Calculate the average marks** by dividing the total marks by 5.
8. **Display the student's marks statement**:
    - Print the student's name.
    - For each subject:
        - Get the corresponding grade by calling the grade(m) function with the marks.
        - Print the subject's marks and the corresponding grade.
9. **Print the total and average marks**.
10. **End**

Program

```
# Function to calculate grade based on average marks
def grade(m):
  if m >= 80:
    return 'O'
  elif m >= 71:
    return 'A+'
  elif m >= 66:
    return 'A'
  elif m >= 61:
    return 'B+'
  elif m >= 56:
    return 'b'
  elif m >= 50:
```

```
        return 'C'
    elif m >= 40:
        return 'P'
    else:
        return 'F'
name = input("Enter student's name: ")
marks = {}
print("Enter marks ")
for i in range(5):
    x = float(input(f"Enter the marks for Subject {i}: "))
    marks[i] = x

total = sum(marks.values())
avg = total / 5
# Display the student's marks statement
print("\n----- Student's Marks Statement -----")
print(f"Student Name: {name}")
for i in range(5):
    gd=grade(marks[i])
    print(f"\nMark {marks[i]} \t Grade {gd} ")
print(f"\nTotal Marks: {total}")
print(f"Average Marks: {avg:.2f}")
```

Output

```
Enter student's name: Guna
Enter marks
Enter the marks for Subject 0: 96
Enter the marks for Subject 1: 67
Enter the marks for Subject 2: 89
Enter the marks for Subject 3: 78
Enter the marks for Subject 4: 55

----- Student's Marks Statement -----
Student Name: Guna
Mark 96.0       Grade O
Mark 67.0       Grade A
Mark 89.0       Grade O
Mark 78.0       Grade A+
Mark 55.0       Grade C
Total Marks: 385.0
Average Marks: 77.00
```

Square Root, GCD and Exponentiation

Aim : To write python program to find square root, GCD and exponentiation of given numbers

Algorithm:

1. **Start**
2. **Define a function find_square_root(n)** that:
   - Takes a number n as input.
   - Uses the math.sqrt() function to calculate the square root of n.
   - Returns the result.
3. **Define a function find_gcd(a, b)** that:
   - Takes two integers a and b as input.
   - Uses the math.gcd() function to find the GCD of a and b.
   - Returns the GCD.
4. **Define a function find_exponentiation(base, exp)** that:
   - Takes two numbers base and exp as input.
   - Uses the math.pow() function to calculate base raised to the power exp (i.e., baseexpbase^{exp}baseexp).
   - Returns the result.
5. **Prompt the user to input a number n for square root calculation**:
   - Read the input and convert it to a floating-point number.
   - Call the find_square_root(n) function with this input.
   - Print the result.
6. **Prompt the user to input two integers a and b to find the GCD**:
   - Read the input and convert them to integers.
   - Call the find_gcd(a, b) function.
   - Print the result.
7. **Prompt the user to input two numbers base and exp for exponentiation**:
   - Read the inputs for base and exp, and convert them to floating-point numbers.
   - Call the find_exponentiation(base, exp) function.
   - Print the result.
8. **End**

Program

```
import math

def find_square_root(n):
  return math.sqrt(n)

def find_gcd(a, b):
  return math.gcd(a, b)
```

```
def find_exponentiation(a, b):
    return math.pow(a, b)

n = float(input("Enter a number to find its square root: "))
print(f"The square root of {n} is: {find_square_root(n):.2f}")

a = int(input("Enter the first number for GCD: "))
b = int(input("Enter the second number for GCD: "))
print(f"The GCD of {a} and {b} is: {find_gcd(a, b)}")

base = float(input("Enter the base for exponentiation: "))
exp = float(input("Enter the exponent: "))
print(f"{base} raised to the power {exp} is: {find_exponentiation(base, exp):.2f}")
```

Output

```
Enter a number to find its square root: 9
The square root of 9.0 is: 3.00

Enter the first number for GCD: 5
Enter the second number for GCD: 20
The GCD of 5 and 20 is: 5

Enter the base for exponentiation: 2
Enter the exponent: 4
2.0 raised to the power 4.0 is: 16.00
```

Sum the array of numbers

Aim : To write python program to sum of given numbers

Algorithm:

1. **Start**
2. **Prompt the user to input n**, the number of variables:
    - Read the input and store it as an integer n.
3. **Initialize an empty list a** to store the input numbers.
4. **For each i from 0 to n-1** (using a loop):
    - Prompt the user to input a number num (as a floating-point value).
    - Append num to the list a.
5. **Calculate the sum of the numbers**:
    - Use the built-in sum() function to calculate the sum of the elements in the list a and store it in x.
6. **Print the sum**:
    - Display the result as the sum of the given values.
7. **End**

Program

```
n = int(input("Enter the number of variables: "))

a = []

for i in range(n):
   num = float(input(f"Enter variable {i+1}: "))
   a.append(num)

x=sum(a)
print(f"The sum of given values is: {x}")
```

Output

```
Enter the number of variables: 5
Enter variable 1: 5
Enter variable 2: 4
Enter variable 3: 3
Enter variable 4: 2
Enter variable 5: 1
The sum of given values is: 15.0
```

Linear Search

Aim : To write python program to search a number given array of numbers

Algorithm:

1. **Start**
2. **Define a function linear_search(arr, target)** that:
    - Takes a list arr and a value target as input.
    - Iterate through the list using a loop from index 0 to len(arr) - 1:
        - For each index i, check if arr[i] == target.
        - If a match is found, return the index i.
    - If the loop finishes without finding the target, return -1 (indicating that the target is not found).
3. **Prompt the user to input the number of elements num**:
    - Read the input and store it as an integer.
4. **Initialize an empty list arr** to store the input elements.
5. **For each i from 0 to num-1** (using a loop):
    - Prompt the user to input the value of the element.
    - Convert the input to an integer and append it to the list arr.
6. **Prompt the user to input the target value** to search for in the list.
7. **Call the linear_search(arr, target) function**:
    - Pass the list arr and the target value as arguments.
    - Store the result (either the index of the target or -1) in the variable result.
8. **Check the value of result**:
    - If result is not equal to -1, print that the target element is present at index result.
    - If result is equal to -1, print that the target element is not found in the array.
9. **End**

Program

```
def linear_search(arr, target):
    for i in range(len(arr)):
        if arr[i] == target:
            return i
    return -1

num = int(input("Enter N value "))
arr = []
for i in range(num):
    x = int(input(f"Enter {i+1} value : "))
    arr.append(x)
```

```
target = int(input("Enter target "))
result = linear_search(arr, target)

if result != -1:
    print(f"Element {target} is present at index {result}.")
else:
    print(f"Element {target} is not found in the array.")
```

Output

```
Enter N value 6
Enter 1 value : 11
Enter 2 value : 34
Enter 3 value : 56
Enter 4 value : 67
Enter 5 value : 23
Enter 6 value : 15
Enter target 67
Element 67 is present at index 3.
```