

INDEX

S. No.	Title	Page No.
PYTHON BASICS		
1.	Input a welcome message and display it	1
2.	Input 2 numbers and display the largest & smallest number	2
3.	Input 3 numbers and display the largest & smallest number	3
4.	<ul style="list-style-type: none"> $1 + x^2 + x^3 \dots + x^n$ $x - \frac{x^2}{2!} + \frac{x^3}{3!} - \frac{x^4}{4!} \dots \pm \frac{x^n}{n!}$ 	4
5.	Check if number is Perfect, Armstrong or Palindrome	6
6.	Prime or composite	9
7.	Fibonacci Series	10
8.	Patterns	11
9.	Character type	13
10.	Convert marks to grade	14
11.	Table of 10	15
12.	Check if date is valid	16
13.	Factorial & Sum of the digits of a number	17
14.	Find sum & average of odd, even and prime numbers	19
15.	Sum of prime numbers in a range of 2 numbers	20
16.	Calculate roots of quadratic equation	21
STRING MANIPULATION		
17.	Count number of times 'a' appears in sentence	22
18.	Print pattern from strings a a abc cba a bb ab ab cb abab ccc abc a c abcabcabc	23
19.	Count number of words in a sentence	25
20.	Count number of vowels in word	26
21.	Check if word is palindrome	27
22.	Check if entered word is present in sentence	28
23.	Find largest name	29
24.	Find shortest name	30
25.	Count number of alphabets, digits and special characters	31
26.	Convert lowercase to uppercase and vice versa	32
27.	Capitalize first letter of each word	33
28.	Extract all numbers and find their sum	34

29.	Print each word in new line	35
30.	Count number of times a word appears in sentence	36
31.	Find and replace a word in given sentence	37
32.	Reverse word and replace in sentence	38
LISTS		
33.	Sum of all odd and even numbers in a list	39
34.	Linear search of number in list	40
35.	Reverse a list	41
36.	Swap 1 st and 2 nd position, 3 rd and 4 th and so on...	42
37.	Swap first and second half of a list	43
38.	Make all numbers divisible by 2 to 0 else 1	44
39.	Split list into odd and even numbers	45
40.	Print all prime numbers from a list	46
41.	Print all perfect numbers from a list	47
42.	Print all Armstrong numbers from a list	48
43.	Merge two lists one after the other	49
44.	Input two lists and combine and sort	50
45.	Input two lists and combine and sort in descending	51
46.	Zipper merge of two lists	52
47.	Take a list, add an element in L th index	53
48.	Delete a number from a list	54
49.	Print sum of each row of matrix	55
50.	Print sum of each column of matrix	56
51.	Print sum and diagonal of square matrix	57
52.	Print lower triangle of square matrix	58
53.	Print transpose of a matrix	59
54.	Search for name in list	60
55.	Input list of n names and print names starting with 'a'	61
56.	Input list of n names and print names containing 'b' or 'v'	62
57.	Input line of text and count words starting with 'a'	63
58.	Input line of text and print all palindrome words	64
59.	Find and replace a word in sentence	65
60.	Input train details and list trains going from Trivandrum to Mumbai	66
61.	Input employee details and change salaries accordingly	68
62.	Input student details and calculate total marks	70
63.	Sort previous list in alphabetical order	72
64.	Big menu driven program	73
65.	Reverse sentence w/o reversing individual words	77

66.	Shift all negative numbers to the right side without changing order	78
67.	Input list and print largest and second largest numbers	79
TUPLES		
68.	Split tuple into odd and even numbers	80
69.	Print max and min elements form a tuple	81
70.	Input n names into a tuple and	82
71.	Print max values from nested tuple	83
72.	Input nested tuples and create new tuple with lengths of sub tuples	84
73.	Input student details and calculate total marks	85
74.	Input student details and print highest scoring student	86
75.	Print number of times 'kerala' appears in nested tuple	87
76.	Input student details, name and marks and add marks to that student	88
77.	Input tuple of numbers and print new tuple with reverse numbers (123 -> 321)	89
78.	Input tuple of numbers and print sum of digits of numbers	90
79.	Input email id and make a tuple with usernames and one with domains	91
80.	Create tuple with n digits of Fibonacci series	92
81.	Input nested tuple of pairs and print out only if the pair is even	93
82.	Print mode of tuple	94
83.	Input nested tuple and print sum of alternate digits	95
84.	Check if max element is repeated	96
85.	Check if min element is in the middle of the tuple	97
86.	Check if inputted tuple is sorted	98

Program 1

Aim: Write a python program to input a welcome message and display it.

Modules used: N/A

Data types used: String

Script:

```
name = input("Enter your name: ")
print(f"Hello {name.capitalize()}!")
```

Output:

```
-----
Enter your name: abyaz
Hello Abyaz!
```

Program 2

Aim: Write a python program to input 2 numbers and display the largest & smallest number.

Modules used: N/A

Data types used: String, float

Script:

```
a, b = input("Enter numbers seperated by comma: ").strip().split(",")
a, b = float(a), float(b)
if a > b:
    print(f"Largest number: {a}\nSmallest number: {b}")
elif a < b:
    print(f"Largest number: {b}\nSmallest number: {a}")
else:
    print("They are equal")
```

Output:

```
>>> |
===== RESTART: D:\School Coding\11TH\I
Enter two numbers seperated by a comma: 1, 2
Largest number: 2.0
Smallest number: 1.0
>>> |
===== RESTART: D:\School Coding\11TH\I
Enter two numbers seperated by a comma: 2, 1
Largest number: 2.0
Smallest number: 1.0
>>> |
===== RESTART: D:\School Coding\11TH\I
Enter two numbers seperated by a comma: 6, 6.0
They are equal
>>> |
```

Program 3

Aim: Write a python program to input 3 numbers and display the largest & smallest number.

Modules used: N/A

Data types used: String, float

Script:

```
a, b, c = input("Enter numbers separated by comma: ").strip().split(",")
a, b, c = float(a), float(b), float(c)

if a == b == c:
    print("They are equal")
else:
    if a >= b and a >= c:
        largest = a
    elif b >= a and b >= c:
        largest = b
    else:
        largest = c

    if a <= b and a <= c:
        smallest = a
    elif b <= a and b <= c:
        smallest = b
    else:
        smallest = c

    print(f"Largest number: {largest}\nSmallest number: {smallest}")
```

Output:

```
>>> Enter numbers separated by comma: 3, 4, 5
      Largest number: 5.0
      Smallest number: 3.0
```

Program 4

Aim: Find the sum of the series: $1 + x^2 + x^3 \dots + x^n$

Modules used: N/A

Data types used: Integer

Script:

```
x = int(input("Enter the value of x: "))
n = int(input("Enter the value of n: "))
ans = 1
for i in range(2, n+1):
    ans += x**i
print(f"Final answer: {ans}")
```

Output:

```
Enter the value of x: 2
Enter the value of n: 5
Final answer: 61
>>> |
```

Program 4

Aim: Find the sum of the series: $x - \frac{x^2}{2!} + \frac{x^3}{3!} - \frac{x^4}{4!} \dots \pm \frac{x^n}{n!}$

Modules used: N/A

Data types used: Integer

Script:

```
def fac(n):
    a = 1
    for i in range(n, 1, -1):
        a * i
    return a

x = int(input("Enter the value of x: "))
n = int(input("Enter the value of n: "))
ans = 0
for i in range(1, n+1):
    if i % 2 == 0:
        ans -= (x**i)/fac(i)
    else:
        ans += (x**i)/fac(i)

print(f"Final answer: {ans}")
```

Output:

```
RESTART:
Enter the value of x: 2
Enter the value of n: 3
Final answer: 6.0
>>> |
```


Program 5

Aim: A menu driven program that checks if the given number is perfect / Armstrong / Palindrome

Modules used: N/A

Data types used: Integer

Script:

```
while True:
    print("\t#-----rEeee-----#")
    print("\t|Check if number is: |")
    print("\t| 1. Perfect |")
    print("\t| 2. Armstrong |")
    print("\t| 3. Palindrome |")
    print("\t#-----#")
    ree = int(input("\t>>> "))
    if ree == 1:
        n = int(input("\n\tEnter number: "))
        l = 1
        for i in range(2,n):
            if n%i == 0:
                l+=i
        if l == n:
            print(f"\t{n} is perfect")
        else:
            print(f"\t{n} is not perfect")
        break
    elif ree == 2:
        n = input("\n\tEnter number: ")
        pow_ = len(n)
        l = 0
        for i in n:
            l += int(i)**pow_
        if l == int(n):
            print(f"\t{n} is an armstrong number")
        else:
            print(f"\t{n} is not an armstrong number")
        break
    elif ree == 3:
        n = input("\n\tEnter number: ")
        if len(n) == 1:
            print(f"\t{n} is a palindrome")
            break
        t = int(n)
        l = 0
        n = int(n)
        for i in range(len(str(n)), 0, -1):
            a = n%10
            n //= 10
            l *= 10
            l += a
        if l == t:
            print(f"\t{t} is a palindrome")
        else:
            print(f"\t{t} is not a palindrome")
        break
    else:
        print("INVALID INPUT\nPlease try again.")
        print("\n\n_____ \n\n")
```

Output:

```

>>> #-----rEeee-----#
    |Check if number is: |
    | 1. Perfect         |
    | 2. Armstrong      |
    | 3. Palindrome     |
    #-----#
    >>> 1

    Enter number: 6
    6 is perfect

>>> #-----rEeee-----#
    |Check if number is: |
    | 1. Perfect         |
    | 2. Armstrong      |
    | 3. Palindrome     |
    #-----#
    >>> 2

    Enter number: 153
    153 is an armstrong number

>>> #-----rEeee-----#
    |Check if number is: |
    | 1. Perfect         |
    | 2. Armstrong      |
    | 3. Palindrome     |
    #-----#
    >>> 3

    Enter number: 12345678987654321
    12345678987654321 is a palindrome

```

```
#-----rEeee-----#
|Check if number is: |
| 1. Perfect         |
| 2. Armstrong       |
| 3. Palindrome      |
#-----#
>>> 5
INVALID INPUT
Please try again.
```

```
#-----rEeee-----#
|Check if number is: |
| 1. Perfect         |
| 2. Armstrong       |
| 3. Palindrome      |
#-----#
>>>
```

Program 6

Aim: Write a program to input a number and check if the number is a prime or composite number.

Modules used: `math`

Data types used: Integer

Script:

```
import math
n = int(input("Enter number: "))
isPrime = True
if n == 1:
    print("1 is neither prime nor composite")
elif n == 2:
    print("2 is prime")
else:
    for i in range(2, math.ceil(math.sqrt(n))+1):
        if n % i == 0:
            isPrime = False
            break
    print(f"{n} is prime") if isPrime else print(f"{n} is not prime")
```

Output:

```
>>> Enter number: 2
      2 is prime
>>>
      =====
>>> Enter number: 10
      10 is not prime
>>>
```

Program 7

Aim: Write a program to display the n terms of a Fibonacci series.

Modules used: N/A

Data types used: Integer

Script:

```
n = int(input("Enter the number of digits: "))
a, b = 0, 1
for i in range(n):
    print(a, end=' ')
    a, b = b, a + b
```

Output:

```
>>> Enter the number of digits: 5
0 1 1 2 3
>>>
```

Program 8

Aim: Generate the following patterns using for loop

Pattern-1	Pattern-2	Pattern-3
*	1 2 3 4 5	A
**	1 2 3 4	AB
***	1 2 3	ABC
****	1 2	ABCD
*****	1	ABCDE

Modules used: N/A

Data types used: Integer / String

Script:

```
print("\tPATTERN 1")
n = int(input("Enter the number of rows: "))
for i in range(1, n+1):
    print("*" * i)

print("\tPATTERN 2")
n = int(input("Enter the number of rows: "))
for i in range(n, 0, -1):
    for j in range(1, n+1):
        print(j, end=" ")
    print()
    n -= 1

print("\tPATTERN 2")
n = int(input("Enter the number of rows: "))
for i in range(1, n+1):
    r = 65+n
    for j in range(65, r):
        print(chr(j), end=" ")
    print()
    n -= 1
```

Output:

```

PATTERN 1
Enter the number of rows: 5
*
**
***
****
*****

PATTERN 2
Enter the number of rows: 5
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1

PATTERN 2
Enter the number of rows: 5
A B C D E
A B C D
A B C
A B
A
>>>

```

Program 9

Aim: Write a program to input a character and print whether it is an upper-case alphabet, lower-case alphabet, a digit, or special character

Modules used: N/A

Data types used: String

Script:

```
n = input("Enter character: ")
c = n[0]
if ord(c) in range(48, 58):
    print(f"{c} is a digit")
elif ord(c) in range(65, 91):
    print(f"{c} is a uppercase character")
elif ord(c) in range(97, 123):
    print(f"{c} is a lowercase character")
else:
    print(f"{c} is a special digit")
```

Output:

```
Enter character: ;
; is a special digit
>>>
===== RESTART
Enter character: C
C is a uppercase character
>>>
===== RESTART
Enter character: l
l is a lowercase character
>>>
===== RESTART
Enter character: 9
9 is a digit
>>>
```


Program 10

Aim: To write a program to input percentage marks of a student and find the grade as per mark.

Modules used: N/A

Data types used: Integer

Script:

```
g = float(input("Enter marks out of 100: "))
o = "F"
if g >= 90:
    o = "A"
elif g >= 80:
    o = "B"
elif g >= 70:
    o = "C"
elif g >= 60:
    o = "D"
elif g >= 50:
    o = "E"
print(f"Grade is {o}")
```

Output:

```
Enter marks out of 100: 87.5
Grade is B
>>>
```

Program 11

Aim: Write a program to print the table of ten

Modules used: N/A

Data types used: Integer, String

Script:

```
n = int(input("Enter the number of rows: "))
for i in range(1, n+1):
    print(f"10 * {i} = {10*i}")
```

Output:

```
Enter the number of rows: 10
10 * 1 = 10
10 * 2 = 20
10 * 3 = 30
10 * 4 = 40
10 * 5 = 50
10 * 6 = 60
10 * 7 = 70
10 * 8 = 80
10 * 9 = 90
10 * 10 = 100
>>> |
```

Program 12

Aim: Write a program to check validity of date

Modules used: N/A

Data types used: Integer

Script:

```
year = int(input("Enter year: "))
month = int(input("Enter month: "))
day = int(input("Enter day: "))

leap_year = (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0)

if month == 2:
    max_days = 29 if leap_year else 28
elif month == 4 or month == 6 or month == 9 or month == 11:
    max_days = 30
else:
    max_days = 31

if day <= max_days:
    print("The date is valid.")
else:
    print("The date is invalid.")
```

Output:

```
Enter year: 2012
Enter month: 2
Enter day: 29
The date is valid.
>>> |
```

Program 13

Aim: Write a menu driven program to find a) factorial of a number b)
Sum of digits of a number

Modules used: N/A

Data types used: Integer / String

Script:

```
while True:
    print("\t#-----rEeee-----#")
    print("\t|Find :           |")
    print("\t|  1. Factorial       |")
    print("\t|  2. Sum of digits    |")
    print("\t|  3. Quit            |")
    print("\t#-----#")
    ree = int(input("\t>>> "))
    if ree == 1:
        n = int(input("\n\tEnter number: "))
        ans = 1
        for i in range(n, 1, -1):
            ans *= i
        print(f"\t{n}! = {ans}")
        break
    elif ree == 2:
        n = input("\n\tEnter number: ")
        ans = 0
        for i in n:
            ans += int(i)
        print(f"\tSum of all digits is: {ans}")
        break
    elif ree == 3:
        print("Quitting")
        break
    else:
        print("\tINVALID INPUT\t\nPlease try again.")
        print("\t\n\n_____ \n\n")|
```

Output:

```

#-----rEeee-----#
|Find :                |
|  1. Factorial        |
|  2. Sum of digits    |
|  3. Quit             |
#-----#
>>> 1

Enter number: 5
5! = 120

>>>
===== RESTART: D:
#-----rEeee-----#
|Find :                |
|  1. Factorial        |
|  2. Sum of digits    |
|  3. Quit             |
#-----#
>>> 2

Enter number: 123
Sum of all digits is: 6

>>> |

```

Program 14

Aim: Write a program to calculate sum and average of odd, even and prime no.

Modules used: N/A

Data types used: Integer / Float

Script:

```
n = int(input("Enter number: "))

SO, SE, SP, CO, CE, CP = 0, 0, 0, 0, 0, 0

for num in range(1, n + 1):
    if num % 2 == 0:
        SE += num
        CE += 1
    else:
        SO += num
        CO += 1

    if num > 1:
        is_prime = True
        for i in range(2, int(num**0.5) + 1):
            if num % i == 0:
                is_prime = False
                break
        if is_prime:
            SP += num
            CP += 1

AO = SO / CO if CO > 0 else 0
AE = SE / CE if CE > 0 else 0
AP = SP / CP if CP > 0 else 0

print(f"Sum of even numbers until {n} = {SE}\nAverage of even numbers until {n} = {AE}\n")
print(f"Sum of odd numbers until {n} = {SO}\nAverage of odd numbers until {n} = {AO}\n")
print(f"Sum of prime numbers until {n} = {SP}\nAverage of prime numbers until {n} = {AP}\n")
```

Output:

```
Enter number: 10
Sum of even numbers until 10 = 30
Average of even numbers until 10 = 6.0

Sum of odd numbers until 10 = 25
Average of odd numbers until 10 = 5.0

Sum of prime numbers until 10 = 17
Average of prime numbers until 10 = 4.25
```

Program 15

Aim: Write a program to find sum of prime no. between 2 ranges

Modules used: N/A

Data types used: Integer / Float

Script:

```
a = int(input("Start of range: "))
b = int(input("End of range: "))
a, b = (a, b) if a > b else (b, a)
ans = 0
for num in range(b, a + 1):
    if num > 1:
        is_prime = True
        for i in range(2, int(num**0.5) + 1):
            if num % i == 0:
                is_prime = False
                break
        if is_prime:
            ans += num
print(f"Sum of prime numbers between {b} and {a} is {ans}")
```

Output:

```
>>> | Start of range: 0
      | End of range: 10
      | Sum of prime numbers between 0 and 10 is 17
```

Program 16

Aim: Write a program to calculate the roots of a quadratic equation

Modules used: `math`

Data types used: Integer / Float

Script:

```
import math

a = float(input("Enter coefficient a: "))
b = float(input("Enter coefficient b: "))
c = float(input("Enter coefficient c: "))

D = b**2 - 4*a*c

if D >= 0:
    if D > 0:
        print(f"The roots are real and distinct, they are: {(-b + math.sqrt(D)) / (2*a)}, {(-b - math.sqrt(D)) / (2*a)}")
    else:
        print(f"The roots are real and equal, it is {(-b - math.sqrt(D)) / (2*a)}")
else:
    print("No real roots")
```

Output:

```
>>> Enter coefficient a: 1
      Enter coefficient b: 0
      Enter coefficient c: -1
      The roots are real and distinct, they are: 1.0, -1.0
>>> |
```


Program 17

Aim: Write a program to input a sentence and count the number of times 'a' appears

Modules used: N/A

Data types used: String

Script:

```
1 s = input(">>> ")
2 a = 0
3 for i in s:
4     if i == 'a':
5         a += 1
6 print(f"number of times 'a' appears is: {a}")
```

Output:

```
>>> hiiii how are you doing my boy
number of times 'a' appears is: 1
```

Program 18

Aim: Write a program to take in a string and print out the following patterns

a	a	abc	cba	a
bb	ab	ab	cb	abab
ccc	abc	a	c	abcabcabc

Modules used: N/A

Data types used:

Script:

```
1 s = input(">>> ")
2
3 # pattern 1
4 for i in range(len(s)):
5     print(s[i] * (i+1))
6
7 print()
8
9 # pattern 2
10 for i in range(len(s)):
11     print(s[:i+1])
12
13 print()
14
15 # pattern 3
16 for i in range(len(s), 0, -1):
17     print(s[:i])
18
19 print()
20
21 # pattern 4
22 for i in range(len(s), 0, -1):
23     print(s[::-1][:i])
24
25 print()
26
27 # pattern 5
28 for i in range(1, len(s) + 1):
29     print(s[:i] * i)
30
```

Output:

```
>>> abc
a
bb
ccc

a
ab
abc

abc
ab
a

cba
cb
c

a
abab
abcabcabc
>>>
```

Program 19

Aim: Write a program to input a sentence and count the number of words

Modules used: N/A

Data types used: String

Script:

```
1 w = input(">>> ").split()
2 print(f"number of words in sentence: {len(w)}")
```

Output:

```
>>> how exasperated      i feel right now
number of words in sentence: 6
```

Program 20

Aim: Write a program to input a word and count the number of vowels in the word

Modules used: N/A

Data types used: String

Script:

```
1 s = input(">>> ")
2 v = 0
3 for i in s:
4     if i in "aeiouAEIOU":
5         v += 1
6 print(f"number of vowels in given input is {v}")
```

Output:

```
>>> i am very swagger
number of vowels in given input is 5
>>>
```

Program 21

Aim: Write a program to input a word and check if it is a palindrome

Modules used: N/A

Data types used: String

Script:

```
1 s = input(">>> ")
2 if s == s[::-1]:
3     print(f"'{s}' is a palindrome")
4 else:
5     print(f"'{s}' is not a palindrome")
```

Output:

```
>>> mom
'mom' is a palindrome
>>>
===== RESTART: D:\Sch
>>> abbas
'abbas' is not a palindrome
>>>
```

Program 22

Aim: Write a program to input a word and a sentence and check whether the word is present in sentence

Modules used: N/A

Data types used: String

Script:

```
1 w = input("Enter word: ")
2 s = input("Enter sentence: ")
3 if w in s:
4     print(f"yes, word is in sentence")
5 else:
6     print(f"no, word is not in sentence")
```

Output:

```
Enter word: existentialism
Enter sentence: i am having an existential crisis
no, word is not in sentence
>>>
===== RESTART: D:\School Coding\CS Periods\
Enter word: apple
Enter sentence: i like apple
yes, word is in sentence
>>>
```

Program 23

Aim: Write a program to input n names and print the largest name

Modules used: N/A

Data types used: String

Script:

```
1 n = int(input("Enter n: "))
2 l = ""
3
4 for i in range(n):
5     c = input(f"{i+1}. ")
6     if len(c) > len(l):
7         l = c
8
9 print(f"The largest string is: {l}")
```

Output:

```
Enter n: 5
1. elephant
2. shark
3. antidisestablishmentarianism
4. hi
5. hehe
The largest string is: antidisestablishmentarianism
>>>
```


Program 24

Aim: Write a program to input n names and print the shortest name

Modules used: N/A

Data types used: String

Script:

```
1 n = int(input("Enter the number of strings: "))
2 s = None
3
4 for i in range(n):
5     c = input(f"{i+1}. ")
6     if s is None or len(c) < len(s):
7         s = c
8
9 print(f"The shortest string is: {s}")
```

Output:

```
Enter the number of strings: 5
1. i
2. really
3. hope
4. this
5. works
The shortest string is: i
>>>
```

Program 25

Aim: Write a program to input a line of text and count the number of alphabets, numbers and special characters in the text

Modules used: N/A

Data types used: String

Script:

```
1 s = input(">>> ")
2 a, d, ob = 0, 0, 0
3 for i in s:
4     if i in "0123456789":
5         d += 1
6     elif i in "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ":
7         a += 1
8     else:
9         ob += 1
10
11 print(f"alphabets - {a}\ndigits - {d}\nspecial characters - {ob}")
```

Output:

```
>>> this is a "test" string $$      :D
alphabets - 18
digits - 0
special characters - 15
>>> |
```

Program 26

Aim: Write a program to input a line of text and convert to uppercase if in lowercase and vice versa

Modules used: N/A

Data types used: String

Script:

```
1 s = input(">>> ")
2 o = ""
3 for i in s:
4     if "a" <= i <= "z":
5         o += chr(ord(i) - 32)
6     elif "A" <= i <= "Z":
7         o += chr(ord(i) + 32)
8     else:
9         o += i
10 print(f"output string: {o}")
```

Output:

```
>>> tHis Is A TeSt
output string: ThIS is a tEst
>>>
```

Program 27

Aim: Write a program to input a line of text and capitalize first character of each word

Modules used: N/A

Data types used: String

Script:

```
1 s = input(">>> ")
2 o = chr(ord(s[0]) - 32)
3
4 for i in range(1, len(s)):
5     if s[i] != " " and s[i-1] == " ":
6         o += chr(ord(s[i]) - 32)
7     else:
8         o += s[i]
9
10 print(f"output string: {o}")
```

Output:

```
>>> i am very swagger guys
output string: I Am Very Swagger Guys
>>>
```

Program 28

Aim: Write a program to input a line of text and extract all numbers and find their sum

Modules used: N/A

Data types used: String

Script:

```
1 s = input(">>> ")
2 d = ""
3 s_ = 0
4 for i in s:
5     if i in "0123456789":
6         d += f"{i} + "
7         s_ += int(i)
8 print(f"Extracted digits : {d[:-3]}\nSum : {s_}")
```

Output:

```
Python 3.7.4 Shell: CPython 3.7.4 Shell [Python 3.7.4]
>>> the saree is actually really cheap and is only 45 aed, the shirt is like only 10 bucks bro
Extracted digits : 4 + 5 + 1 + 0
Sum : 10
>>>
```

Program 29

Aim: Write a program to input a line of text and print each word in a new line

Modules used: N/A

Data types used: String

Script:

```
1 s = input(">>> ")
2 for i in s.split():
3     print(i)
```

Output:

```
>>> hello, how are you doing today      john
hello,
how
are
you
doing
today
john
>>>
```

Program 30

Aim: Write a program to input a line of text and a word and count the number of times the word appears in the text

Modules used: N/A

Data types used: String

Script:

```
1 t = input("Enter text: ")
2 w = input("Enter word: ")
3 c = 0
4
5 for i in t.split():
6     if i == w:
7         c += 1
8
9 print(f"number of times '{w}' appears is {c}")
```

Output:

```
Enter text: i love apples man, like i love the sweetness of apples
Enter word: apples
number of times 'apples' appears is 2
>>>
```

Program 31

Aim: Write a program to input a line of text, and two words and replace the first word with the second word

Modules used: N/A

Data types used: String

Script:

```
1 s = input(">>> ").split()
2 o = input("Pick word to replace: ")
3 n = input(f"Pick word to replace '{o}' with: ")
4
5 o_ = ""
6 for i in s:
7     if i == o:
8         o_ += f"{n} "
9     else:
10        o_ += f"{i} "
11
12 print(o_[:-1])
```

Output:

```
>>> hello my name is asdf
Pick word to replace: asdf
Pick word to replace 'asdf' with: abyaz
hello my name is abyaz
>>>
```


Program 32

Aim: Write a program to input a line of text and a word, reverse the word and replace it in the text

Modules used: N/A

Data types used: String

Script:

```
1 s = input(">>> ").split()
2 w = input("Pick word to reverse: ")
3
4 o_ = ""
5 for i in s:
6     if i == w:
7         o_ += f"{w[::-1]} "
8     else:
9         o_ += f"{i} "
10
11 print(o_[:-1])
```

Output:

```
>>> hello my name is zayba
Pick word to reverse: zayba
hello my name is abyaz
>>>
```

Program 33

Aim: Input a list of numbers and find sum of odd and even numbers separately

Modules used: N/A

Data types used: List, Int

Script:

```
l = list(eval(input(">>> ")))
so, se = 0, 0
for i in l:
    if i % 2 == 0:
        se += i
    else:
        so += i
print(f"Sum of even: {se}\nSum of odd : {so}")
```

Output:

```
>>> 1, 2, 3, 4, 5, 6, 7, 8, 9
Sum of even: 20
Sum of odd : 25
>>> |
```

Program 34

Aim: To input a list of numbers and search for a number using linear search

Modules used: N/A

Data types used: List

Script:

```
l = list(eval(input(">>> ")))
c = input("enter number to search for: ")
for j in l:
    if c == str(j):
        print(f"number '{c}' found in list")
        break
    else:
        print(f"number '{c}' not found in list")
```

Output:

```
>>> 1, 2, 3, 4, 5, 6
enter number to search for: 3
number '3' found in list
>>>
=====
>>> 1, 2, 3, 4, 5, 6
enter number to search for: 8888
number '8888' not found in list
>>>
```

Program 35

Aim: To input a list of numbers and reverse the list in place (without creating a new list)

Modules used: N/A

Data types used: List

Script:

```
l = list(eval(input(">>> ")))

left = 0
right = len(l) - 1

while left < right:
    l[left], l[right] = l[right], l[left]
    left += 1
    right -= 1

print(l)
```

Output:

```
>>> 1, 2, 3, 4, 5, 6
[6, 5, 4, 3, 2, 1]
>>>
```

Program 36

Aim: Swap 1st element to 2nd and so on...

Modules used: N/A

Data types used: List

Script:

```
l = list(eval(input(">>> ")))

if len(l) % 2 != 0:
    print("list has odd number of elements")
    l.append(eval(input("Please enter one more element: ")))

for i in range(0, len(l) - 1, 2):
    l[i], l[i + 1] = l[i + 1], l[i]

print(l)
```

Output:

```
>>> 1, 2, 3, 4
[2, 1, 4, 3]
>>>
=====
>>> 1, 2, 3
list has odd number of elements
Please enter one more element: 4
[2, 1, 4, 3]
>>> |
```

Program 37

Aim: Swap first and second half

Modules used: N/A

Data types used: List

Script:

```
l = list(eval(input(">>> ")))

mid = len(l) // 2

if len(l) % 2 == 0:
    temp = l[:mid]
    l[:mid] = l[mid:]
    l[mid:] = temp
else:
    temp = l[:mid]
    l[:mid] = l[mid + 1:]
    l[mid + 1:] = temp

print(l)
```

Output:

```
>>> 1,2,3,4,5,6
[4, 5, 6, 1, 2, 3]
>>>
=====
>>> 1,2,3,4,5
[4, 5, 3, 1, 2]
>>>
```

Program 38

Aim: Change all numbers divisible by 2 to 0 and others to 1

Modules used: N/A

Data types used: List

Script:

```
l = list(eval(input(">>> ")))
o = []
for i in l:
    o.append(i % 2)
print(o)
```

Output:

```
>>> 1, 2, 3, 4, 5, 6
[1, 0, 1, 0, 1, 0]
>>>
```

Program 39

Aim: Split one list into two containing odd and even numbers

Modules used: N/A

Data types used: List

Script:

```
l = list(eval(input(">>> ")))
lo, le = [], []
for i in l:
    lo.append(i) if i % 2 != 0 else le.append(i)
print(f"odd numbers : {lo}\neven numbers: {le}")
```

Output:

```
>>> 1, 2, 3, 4, 5, 6, 7, 8, 9
odd numbers : [1, 3, 5, 7, 9]
even numbers: [2, 4, 6, 8]
>>> |
```


Program 40

Aim: To input a list and print all prime numbers from it

Modules used: math

Data types used: List

Script:

```
import math
l = list(eval(input(">>> ")))
p = []
for n in l:
    isPrime = True
    if n == 1:
        continue
    elif n == 2:
        p.append(2)
    else:
        for i in range(2, math.ceil(math.sqrt(n))+1):
            if n % i == 0:
                isPrime = False
                break
        p.append(n) if isPrime else ...
print(f"list of primes: {p}")
```

Output:

```
>>> 2, 63, 21, 5, 88
list of primes: [2, 5]
>>>
```

Program 41

Aim: To input a list and print all perfect numbers from it

Modules used: N/A

Data types used: List

Script:

```
l_ = list(eval(input(">>> ")))
p = []

for n in l_:
    l = 1
    for i in range(2,n):
        if n%i == 0:
            l+=i
    if l == n:
        p.append(n)

print(f"list of perfect numbers: {p}")
```

Output:

```
>>> 1, 3, 6
list of perfect numbers: [1, 6]
>>>
```

Program 42

Aim: To input a list and print all Armstrong numbers from it

Modules used: N/A

Data types used: List

Script:

```
l_ = list(eval(input(">>> ")))
a = []

for n in l_:
    n = str(n)
    pow_ = len(n)
    l = 0
    for i in n:
        l += int(i)**pow_
    if l == int(n):
        a.append(int(n))

print(f"list of armstrong numbers: {a}")
```

Output:

```
>>> 1, 153, 135
list of armstrong numbers: [1, 153]
>>> |
```

Program 43

Aim: To input two lists and merge them one after the other

Modules used: N/A

Data types used: List

Script:

```
print(list(eval(input("list 1: "))) + list(eval(input("list 2: "))))
```

Output:

```
list 1: 1,2,3,5,8
list 2: 89,63,2.5,6
[1, 2, 3, 5, 8, 89, 63, 2.5, 6]
>>>
```

Program 44

Aim: To input two lists A and B merge the list into a third list C in ascending order

Modules used: N/A

Data types used: List

Script:

```
l = list(eval(input("list 1: "))) + list(eval(input("list 2: ")))

stack = [(0, len(l) - 1)]
top = -1

while top != -1 or not stack:
    if top == -1:
        start, end = stack[0]
        top += 1
    else:
        start, end = stack[top]
        top -= 1

    if start < end:
        pivot = l[end]
        p_index = start

        for i in range(start, end):
            if l[i] < pivot:
                l[i], l[p_index] = l[p_index], l[i]
                p_index += 1

        l[p_index], l[end] = l[end], l[p_index]

        top += 1
        stack.append((start, p_index - 1))

        top += 1
        stack.append((p_index + 1, end))

print(f"sorted list : {l}")
```

Output:

```
list 1: 1,2,3,5,4
list 2: 8,9,4,85
sorted list : [1, 2, 3, 5, 4, 8, 9, 4, 85]
>>>
```

Program 45

Aim: To input two lists A and B merge the list into a third list C. Where A is in ascending and B and C are in descending order

Modules used: N/A

Data types used: List

Script:

```
l1 = list(eval(input("list 1: ")))
l2 = list(eval(input("list 2: ")))

for i in range(len(l1)):
    for j in range(i + 1, len(l1)):
        if l1[i] > l1[j]:
            l1[i], l1[j] = l1[j], l1[i]

for i in range(len(l2)):
    for j in range(i + 1, len(l2)):
        if l2[i] < l2[j]:
            l2[i], l2[j] = l2[j], l2[i]

o = [0] * (len(l1) + len(l2))
for i in range(len(l1)):
    o[i] = l1[i]
for j in range(len(l2)):
    o[len(l1) + j] = l2[j]

for i in range(len(o)):
    for j in range(i + 1, len(o)):
        if o[i] < o[j]:
            o[i], o[j] = o[j], o[i]

print(f"First list: {l1}")
print(f"Second list: {l2}")
print(f"Combined list: {o}")
```

Output:

```
>>> list 1: 1,2,33,85
list 2: 87,64,18,2.5
First list: [1, 2, 33, 85]
Second list: [87, 64, 18, 2.5]
Combined list: [87, 85, 64, 33, 18, 2.5, 2, 1]
```

Program 46

Aim: Take two lists and perform a zipper merge on them

Modules used: N/A

Data types used: List

Script:

```
l1 = list(eval(input(">>> ")))
l2 = list(eval(input(">>> ")))

o = []
if len(l1) < len(l2):
    min_ = len(l1)
else:
    min_ = len(l2)

for i in range(min_):
    o.append(l1[i])
    o.append(l2[i])

for i in range(min_, len(l1)):
    o.append(l1[i])

for i in range(min_, len(l2)):
    o.append(l2[i])

print(o)
```

Output:

```
>>> 1,2,3,4
>>> 'a', 'b', 'c', 'd'
[1, 'a', 2, 'b', 3, 'c', 4, 'd']
>>>
```

Program 47

Aim: To input n numbers into a list and add another number in the Lth index

Modules used: N/A

Data types used: List

Script:

```
l, o = [], []

for i in range(int(input("Enter n: "))):
    l.append(int(input(f"{i+1}. ")))

L = int(input("enter index where you want to add element: "))

o += l[:L+1] + list(eval(input("enter number you wanna add: ")+"")) + l[L+1:]
print(o)
```

Output:

```
Enter n: 5
1. 81
2. 64
3. 79
4. 23
5. 45
enter index where you want to add element: 3
enter number you wanna add: 589
[81, 64, 79, 23, 589, 45]
>>> |
```


Program 48

Aim: To take a list and delete a number from that list

Modules used: N/A

Data types used: List

Script:

```
l = list(eval(input("enter list: ")))
n = input("Enter number to remove: ")
o = []

for i in l:
    if str(i) == n:
        continue
    o.append(i)

print(o)
```

Output:

```
>>> | enter list: 1,2,3,5,4
      | Enter number to remove: 5
      | [1, 2, 3, 4]
```

Program 49

Aim: To input an $m \times n$ matrix and find the sum of each row

Modules used: N/A

Data types used: List

Script:

```
m = int(input("m: "))
n = int(input("n: "))
M = []

for i in range(m):
    l_ = []
    for j in range(n):
        l_.append(int(input(">>> ")))
    M.append(l_)

print("Matrix: ")
for i in M:
    print(i)
print()

for i, row in enumerate(M):
    row_sum = sum(row)
    print(f"Sum of row {i + 1}: {row_sum}")
```

Output:

```
m: 3
n: 2
>>> 1
>>> 2
>>> 3
>>> 4
>>> 5
>>> 6
Matrix:
[1, 2]
[3, 4]
[5, 6]

Sum of row 1: 3
Sum of row 2: 7
Sum of row 3: 11
>>> |
```

Program 50

Aim: To input an $m \times n$ matrix and find the sum of each column

Modules used: N/A

Data types used: List

Script:

```
m = int(input("m: "))
n = int(input("n: "))
M = []

for i in range(m):
    l_ = []
    for j in range(n):
        l_.append(int(input(">>> ")))
    M.append(l_)

print("Matrix: ")
for i in M:
    print(i)
print()

s = [0] * len(M[0])

for row in M:
    for j in range(len(row)):
        s[j] += row[j]

for index, total in enumerate(s):
    print(f"Sum of col {index + 1}: {total}")
```

Output:

```
m: 3
n: 2
>>> 1
>>> 2
>>> 3
>>> 4
>>> 5
>>> 6
Matrix:
[1, 2]
[3, 4]
[5, 6]

Sum of col 1: 9
Sum of col 2: 12
>>> |
```

Program 51

Aim: To input an $m \times m$ matrix and print as well as find the sum of the first diagonal

Modules used: N/A

Data types used: List

Script:

```
m = int(input("m: "))
M = []

for i in range(m):
    l_ = []
    for j in range(m):
        l_.append(int(input(">>> ")))
    M.append(l_)

s = 0

for i in range(len(M)):
    for j in range(len(M[i])):
        if i == j:
            print(f"{M[i][j]} ", end="")
            s += M[i][j]
        else:
            print(" ", end="")
    print()

print(f"\nsum of above diagonal is: {s}")
```

Output:

```
m: 3
>>> 1
>>> 2
>>> 3
>>> 4
>>> 5
>>> 6
>>> 7
>>> 8
>>> 9
1
  5
    9
sum of above diagonal is: 15
>>>
```

Program 52

Aim: To input an $m \times m$ matrix and print the lower triangle

Modules used: N/A

Data types used: List

Script:

```
m = int(input("m: "))
M = []

for i in range(m):
    l_ = []
    for j in range(m):
        l_.append(int(input(">>> ")))
    M.append(l_)

for i in range(m):
    for j in range(m):
        if i >= j:
            print(M[i][j], end=" ")
        else:
            print(" ", end=" ")
    print()
```

Output:

```
m: 3
>>> 1
>>> 2
>>> 3
>>> 4
>>> 5
>>> 6
>>> 7
>>> 8
>>> 9
1
4 5
7 8 9
>>> |
```

Program 53

Aim: To input an $m \times m$ matrix and print the transpose of it

Modules used: N/A

Data types used: List

Script:

```
m = int(input("m: "))
M = []

for i in range(m):
    l_ = []
    for j in range(m):
        l_.append(int(input(">>> ")))
    M.append(l_)

T = [[0] * m for _ in range(m)]

for i in range(m):
    for j in range(m):
        T[i][j] = M[j][i]

for i in T:
    print(i)
```

Output:

```
m: 3
>>> 1
>>> 2
>>> 3
>>> 4
>>> 5
>>> 6
>>> 7
>>> 8
>>> 9
[1, 4, 7]
[2, 5, 8]
[3, 6, 9]
>>> |
```

Program 54

Aim: To input n names into a list and search for a name

Modules used: N/A

Data types used: List

Script:

```
l = []

for i in range(int(input("Enter n: "))):
    l.append(input(f"{i+1}. "))

s = input("search term: ")

for i in l:
    if i == s:
        print(f"name '{s}' found")
        break
    else:
        print(f"name '{s}' not found")
```

Output:

```
>>> Enter n: 3
      1. john
      2. asdf
      3. fdsa
      search term: asdf
      name 'asdf' found

>>> =====
>>> Enter n: 2
      1. john marton
      2. marton john
      search term: asdf
      name 'asdf' not found

>>>
```

Program 55

Aim: To input n names into a list and print all names starting with 'a'

Modules used: N/A

Data types used: List

Script:

```
a = []

for i in range(int(input("Enter n: "))):
    temp = input(f"{i+1}. ")
    a.append(temp) if temp[0] in "Aa" else ...

print("\nNames starting with 'a' :")
for i in a:
    print(i)
```

Output:

```
Enter n: 5
1. asdf
2. abyaz
3. hii
4. fdsa
5. wee woo wee woo

Names starting with 'a' :
asdf
abyaz
>>> |
```


Program 56

Aim: To input n names into a list and print all names that have either 'b' or 'v'

Modules used: N/A

Data types used: List

Script:

```
o = []

for i in range(int(input("Enter n: "))):
    temp = input(f"{i+1}. ")
    o.append(temp) if "B" in temp or "b" in temp or "V" in temp or "v" in temp else ...

print("\nNames containing either 'b' or 'v' :")
for i in o:
    print(i)
```

Output:

```
Enter n: 5
1. bivabasu
2. abhinav
3. star.stalker9160
4. asdf
5. fdsab

Names containing either 'b' or 'v' :
bivabasu
abhinav
fdsab
>>> |
```

Program 57

Aim: To input a line of text and count all words that start with 'a'

Modules used: N/A

Data types used: List

Script:

```
l = input(">>> ").split()
c = 0

for i in l:
    if i[0] in "Aa":
        c += 1

print(f"no. of words starting with a are: {c}")
```

Output:

```
>>> | >>> amazing aligators are always hungry
    | no. of words starting with a are: 4
>>> |
```

Program 58

Aim: To input a sentence and print all palindrome words

Modules used: N/A

Data types used: List

Script:

```
l = input(">>> ").split()
for i in l:
    print(i) if i[::-1] == i else ...
```

Output:

```
>>> I like my mom !!
I
mom
!!
>>> |
```

Program 59

Aim: To input a list and search and replace a specific word

Modules used: N/A

Data types used: List

Script:

```
l = input("enter sentence: ").split()
o = input("pick word to replace: ")
n = input(f"pick word to replace '{o}' with: ")

for i, v in enumerate(l):
    if v == o:
        l[i] = n

print("New sentence: ")
print(' '.join(l))
```

Output:

```
enter sentence: i hole i dont make a typo
pick word to replace: hole
pick word to replace 'hole' with: hope
New sentence:
i hope i dont make a typo
>>> |
```

Program 60

Aim: Enter train details and print trains going from Trivandrum to Mumbai

Modules used: N/A

Data types used: List

Script:

```
trains = []

print(r"""
          o o o o
          | PMD \_ |[]| _' _Y
          |-----|_|_|_|_|}
=====00--00==00--000\=====
""")

for i in range(int(input("enter n: "))):
    print("\n")
    t = []
    endPts = []
    t.append(int(input("Train no: ")))
    t.append(input("Train name: "))
    endPts.append(input("Start point: "))
    endPts.append(input("Destination: "))
    t.append(endPts)
    trains.append(t)

for i in trains:
    if i[2] == ["trivandrum", "mumbai"]:
        print(f"Train '{i[1]}' with train no. {i[0]} is traveling from Benares to Kolkata.")

print("\n\n")
print("Train table:\n")
print(f"{'train number':^15} | {'train name':^12} | {'starting point':^16} | {'ending point':^14}")
print("-" * 65)

for item in trains:
    print(f"{'item[0]:^15} | {'item[1]:^12} | {'item[2][0]:^16} | {'item[2][1]:^14}")
```

Output:

```

          o o 0 0
          | PMD \_[]\_'\_Y
          |_____|_|_|_|}
=====00--00==00--000\\=====

enter n: 4

Train no: 1
Train name: Express 1
Start point: trivandrum
Destination: mumbai

Train no: 2
Train name: Express 2
Start point: kolkata
Destination: chennai

Train no: 3
Train name: Express 3
Start point: mumbai
Destination: trivandrum

Train no: 4
Train name: asdf
Start point: dubai
Destination: karnataka
Train 'Express 1' with train no. 1 is traveling from Benares to Kolkata.

Train table:

  train number | train name | starting point | ending point
-----
          1   | Express 1  | trivandrum    | mumbai
          2   | Express 2  | kolkata       | chennai
          3   | Express 3  | mumbai        | trivandrum
          4   | asdf       | dubai         | karnataka
>>> |
```

Program 61

Aim: Enter employee details and increase salary for managers by 1000 and 500 for everyone else

Modules used: N/A

Data types used: List

Script:

```
employees = []

for i in range(int(input("enter n: "))):
    print("\n")
    e = []
    e.append(int(input("ecode: ")))
    e.append(input("name: "))
    e.append(input("designation: "))
    sal = float(input("salary: "))
    e.append(sal)
    employees.append(e)

print("\npre-change employee table:\n")
print(f"{'ecode':^10} | {'name':^15} | {'designation':^15} | {'salary':^10}")
print("-" * 60)

for e in employees:
    print(f"{'e[0]:^10} | {'e[1]:^15} | {'e[2]:^15} | {'e[3]:^10.2f}")

for e in employees:
    if e[2] == "manager":
        e[3] += 1000
    else:
        e[3] += 500

print("\npost-change employee table:\n")
print(f"{'ecode':^10} | {'name':^15} | {'designation':^15} | {'salary':^10}")
print("-" * 60)

for e in employees:
    print(f"{'e[0]:^10} | {'e[1]:^15} | {'e[2]:^15} | {'e[3]:^10.2f}")
```

Output:

```
enter n: 4

ecode: 101
name: dude 1
designation: manager
salary: 10_000

ecode: 102
name: dude 5
designation: developer
salary: 5000

ecode: 103
name: dude 2
designation: developer
salary: 5000

ecode: 104
name: dude 9
designation: manager
salary: 10_000

pre-change employee table:

  ecode |      name      | designation | salary
-----|-----|-----|-----
  101   | dude 1         | manager    | 10000.00
  102   | dude 5         | developer  | 5000.00
  103   | dude 2         | developer  | 5000.00
  104   | dude 9         | manager    | 10000.00

post-change employee table:

  ecode |      name      | designation | salary
-----|-----|-----|-----
  101   | dude 1         | manager    | 11000.00
  102   | dude 5         | developer  | 5500.00
  103   | dude 2         | developer  | 5500.00
  104   | dude 9         | manager    | 11000.00
>>>
```


Program 62

Aim: Enter student details, calculate total marks and add them to the list.
Print the name of the student getting the highest marks

Modules used: N/A

Data types used: List

Script:

```
students = students = []

for i in range(int(input("enter n: "))):
    print("\n")
    s = []
    s.append(input("name: "))
    theory = int(input("theory marks: "))
    practical = int(input("practical marks: "))
    s.append(theory)
    s.append(practical)
    totalMarks = theory + practical
    s.append(totalMarks)
    students.append(s)

for s in students:
    totalMarks = s[1] + s[2]
    s.append(totalMarks)

print(f"{'name':^15} | {'theory marks':^15} | {'practical marks':^17} | {'total marks':^12}")
print("-" * 65)
for s in students:
    print(f"{s[0]:^15} | {s[1]:^15} | {s[2]:^17} | {s[3]:^12}")

nerd = students[0]
for s in students:
    if s[3] > nerd[3]:
        nerd = s

print("\n")
print(f"the student with the highest marks is: {nerd[0]} with {nerd[3]} marks.")
```

Output:

```
enter n: 4

name: ree
theory marks: 78
practical marks: 85

name: nerd 2
theory marks: 100
practical marks: 100

name: dumbass
theory marks: 61
practical marks: 54

name: asdf
theory marks: 75
practical marks: 85
  name      | theory marks | practical marks | total marks
-----|-----|-----|-----
    ree     |          78  |          85     |        163
  nerd 2    |         100  |         100     |        200
  dumbass   |          61  |          54     |        115
    asdf    |          75  |          85     |        160

the student with the highest marks is: nerd 2 with 200 marks.
>>> |
```

Program 63

Aim: Sort the previous table in alphabetical order

Modules used: N/A

Data types used: List

Script:

```
students = [  
    ["ree", 78, 85],  
    ["nerd 2", 100, 100],  
    ["dumbass", 61, 54],  
    ["fdsa", 75, 85]  
]  
  
for i in range(len(students)):  
    minIndex = i  
    for j in range(i + 1, len(students)):  
        if students[j][0] < students[minIndex][0]:  
            minIndex = j  
    students[i], students[minIndex] = students[minIndex], students[i]  
  
print("\nsorted student table:\n")  
print(f"{'name':^15} | {'theory marks':^15} | {'practical marks':^17}")  
print("-" * 50)  
for s in students:  
    print(f"{s[0]:^15} | {s[1]:^15} | {s[2]:^17}")
```

Output:

```
>>> sorted student table:  
  
      name      |  theory marks  |  practical marks  
-----  
    dumbass     |         61     |          54  
      fdsa      |         75     |          85  
    nerd 2      |        100     |         100  
       ree      |         78     |          85
```

Program 64

Aim: Write a menu driven program to do the following:

- To input n item details (item code, item name, price and quantity) as one list (nested list)
- To print a particular item detail taking key fields as code
- To print all items
- To print all items where the quantity is 0
- To add more stock to a particular item
- To add a new product to the list
- To delete an item key field as code
- Sort the list

Modules used: N/A

Data types used: List

Script:

```
i = []

print("\n\t#-----rEee-----#")
print("\t| 1. add item          |")
print("\t| 2. print item details  |")
print("\t| 3. print all items     |")
print("\t| 4. print out of stock  |")
print("\t| 5. add stock           |")
print("\t| 6. delete item         |")
print("\t| 7. sort items          |")
print("\t| 8. exit                |")
print("\t#-----#")

while True:
    c = int(input("\t>>> "))

    if c == 1:
        a = input("\n\tenter item code: ")
        b = input("\n\tenter item name: ")
        p = float(input("\n\tenter price: "))
        q = int(input("\n\tenter quantity: "))
        i.append([a, b, p, q])

    elif c == 2:
        k = input("\n\tenter item code to find: ")
        f = 0
        for t in i:
            if t[0] == k:
                print(f"\titem code {t[0]} name {t[1]} price {t[2]} quantity {t[3]}")
                f = 1
        if f == 0:
            print("\titem not found")

    elif c == 3:
        print("\n\tall items")
        print(f"{'item code':^15} | {'item name':^15} | {'price':^10} | {'quantity':^10}")
        print("-" * 60)
        for t in i:
            print(f"{t[0]:^15} | {t[1]:^15} | {t[2]:^10} | {t[3]:^10}")

    elif c == 4:
        print("\n\tout of stock items")
        print(f"{'item code':^15} | {'item name':^15} | {'price':^10} | {'quantity':^10}")
        print("-" * 60)
        for t in i:
            if t[3] == 0:
                print(f"{t[0]:^15} | {t[1]:^15} | {t[2]:^10} | {t[3]:^10}")
```

```

elif c == 5:
    k = input("\n\tenter item code to add stock: ")
    f = 0
    for t in i:
        if t[0] == k:
            a = int(input("\tenter quantity to add: "))
            t[3] += a
            f = 1
            print("\tstock updated")
    if f == 0:
        print("\titem not found")

elif c == 6:
    k = input("\n\tenter item code to delete: ")
    d = -1
    for j in range(len(i)):
        if i[j][0] == k:
            d = j
            break
    if d != -1:
        for j in range(d, len(i) - 1):
            i[j] = i[j + 1]
        i.pop()
        print("\titem deleted")
    else:
        print("\titem not found")

elif c == 7:
    for j in range(len(i)):
        for m in range(j + 1, len(i)):
            if i[j][1] > i[m][1]:
                i[j], i[m] = i[m], i[j]
    print("\titems sorted by name")

elif c == 8:
    break

else:
    print("\tinvalid input please try again")

```

Output:

```
#-----rEee-----#
| 1. add item          |
| 2. print item details|
| 3. print all items   |
| 4. print out of stock|
| 5. add stock         |
| 6. delete item       |
| 7. sort items        |
| 8. exit              |
#-----#
```

```
>>> 1
```

```
enter item code: 101
enter item name: ITEM 1
enter price: 50.00
enter quantity: 3
>>> 1
```

```
enter item code: 102
enter item name: ITEM 2
enter price: 25.13
enter quantity: 1
>>> 1
```

```
enter item code: 103
enter item name: ITEM 3
enter price: 99.99
enter quantity: 0
>>> 2
```

```
enter item code to find: 102
item code 102 name ITEM 2 price 25.13 quantity 1
>>> 3
```

```
all items
```

item code	item name	price	quantity
101	ITEM 1	50.0	3
102	ITEM 2	25.13	1
103	ITEM 3	99.99	0

```
>>> 4
```

```
out of stock items
```

item code	item name	price	quantity
103	ITEM 3	99.99	0

```
>>> 5
```

```
enter item code to add stock: 103
```

```
enter quantity to add: 88
```

```
stock updated
```

```
>>> 3
```

```
all items
```

item code	item name	price	quantity
101	ITEM 1	50.0	3
102	ITEM 2	25.13	1
103	ITEM 3	99.99	88

```
>>> 6
```

```
enter item code to delete: 101
```

```
item deleted
```

```
>>> 7
```

```
items sorted by name
```

```
>>> 3
```

```
all items
```

item code	item name	price	quantity
102	ITEM 2	25.13	1
103	ITEM 3	99.99	88

```
>>> 8
```

```
>>>
```

Program 65

Aim: To input a line of text and reverse it without reversing the words

Modules used: N/A

Data types used: List

Script:

```
print(" ".join(input(">>> ").split()[::-1]))
```

Output:

```
>>> hello how are you
you are how hello
>>>
```


Program 66

Aim: Write a program to bring all negative numbers to the right end of the list

Modules used: N/A

Data types used: List

Script:

```
l = list(eval(input(">>> ")))
p, n = [], []

for i in l:
    n.append(i) if i < 0 else p.append(i)

print(p + n)
```

Output:

```
>>> 10, -18, 59, -5, -9, 255, 198, -2.58
[10, 59, 255, 198, -18, -5, -9, -2.58]
>>> |
```

Program 67

Aim: Write a program to input n numbers into a list, and print the largest and second largest numbers

Modules used: N/A

Data types used: List

Script:

```
l = []

for i in range(int(input("Enter n: "))):
    l.append(int(input(f"{i+1}. ")))

largest = second_largest = float('-inf')

for i in range(len(l)):
    if l[i] > largest:
        second_largest = largest
        largest = l[i]
    elif l[i] > second_largest and l[i] != largest:
        second_largest = l[i]

if second_largest == float('-inf'):
    print("There is no second largest number.")
else:
    print(f"Largest number: {largest}")
    print(f"Second largest number: {second_largest}")
```

Output:

```
>>> Enter n: 5
      1. 58
      2. 98
      3. -295
      4. 95
      5. 2
      Largest number: 98
      Second largest number: 95
>>> |
```

Program 68

Aim: Split one tuple into two containing odd and even numbers

Modules used: N/A

Data types used: Tuple, Int

Script:

```
o, e = (), ()
for i in eval(input(">>> ")):
    if i % 2 == 0:
        e += (i,)
    else:
        o += (i,)
print(f"odd numbers : {o}\neven numbers: {e}")
```

Output:

```
>>> 1,2,3,4,5,6,7,8,9
odd numbers : (1, 3, 5, 7, 9)
even numbers: (2, 4, 6, 8)
>>> |
```

Program 69

Aim: Print max and min elements from a tuple

Modules used: N/A

Data types used: Tuple, Int

Script:

```
s, l = None, 0
for i in eval(input(">>> ")):
    if i > l:
        l = i
    if s == None or i < s:
        s = i
print(f"largest : {l}\nsmallest: {s}")
```

Output:

```
>>> 1,2,3,4,5,6,7,8,9
largest : 9
smallest: 1
>>> |
```

Program 70

Aim: Input n names into a tuple and print ones with 5 letters

Modules used: N/A

Data types used: Tuple, Str

Script:

```
t = eval(input(">>> "))
print("Names with 5 letters")
for i in t:
    if len(i) == 5:
        print(i)
```

Output:

```
>>> "alice", "name", "person 2", "abbas", "e"
Names with 5 letters
alice
abbas
>>> |
```

Program 71

Aim: Input nested tuple and make a new tuple containing max elements

Modules used: N/A

Data types used: Tuple, Int

Script:

```
T = ()
for i in eval(input(">>> ")):
    l = 0
    for j in i:
        l = j if j > l else ...
    T += (l,)
print(f"largest elements of entered tuples: {T}")
```

Output:

```
>>> (1, 2), (3,4,5), (000.1, 93202394)
largest elements of entered tuples: (2, 5, 93202394)
>>> |
```

Program 72

Aim: Input a nested tuple and create a new tuple with sub tuple lengths

Modules used: N/A

Data types used: Tuple

Script:

```
T = ()
for i in eval(input(">>> ")):
    T += (len(i),)
print(f"sub tuple lengths: {T}")
```

Output:

```
>>> (1,), (2, 2), (5, 5, 5, 5, 5)
sub tuple lengths: (1, 2, 5)
>>>
```

Program 73

Aim: To input nested tuple containing names and individual marks, print name and total marks

Modules used: N/A

Data types used: Tuple, Int

Script:

```
t = eval(input(">>> "))
print("student table: \n")
print(f"{'name':^15} | {'total marks':^15}")
print("-"*30)
for i in t:
    print(f"{i[0]:^15} | {(i[1]+i[2]):^15}")
```

Output:

```
>>> ("nerd 47", 100, 100), ("rEee", 50, 50), ("asdf", 20, 90)
student table:

      name      |  total marks
-----|-----
nerd 47         |          200
rEee           |          100
asdf           |          110
>>>
```


Program 74

Aim: To input a nested tuple of name and marks and print the name of the student getting highest marks

Modules used: N/A

Data types used: Tuple, Int

Script:

```
dude = None
m = 0
for i in eval(input(">>> ")):
    if (i[1]+i[2]) > m:
        m = i[1]+i[2]
        dude = i[0]
print(f"student with highest marks is {dude} with marks {m}")
```

Output:

```
>>> ("nerd 47", 100, 100), ("rEee", 50, 50), ("asdf", 20, 90)
student with highest marks is nerd 47 with marks 200
>>>
```

Program 75

Aim: Input nested tuples and print number of times 'kerala' appears

Modules used: N/A

Data types used: Tuple, Str

Script:

```
K = 0
for i in eval(input(">>> ")):
    if "kerala" in i:
        for j in i:
            if j == "kerala": K += 1
print(f"number of times kerala appears is {K}")
```

Output:

```
>>> ('karela', 'kerala'), ('australia', 'delhi', 'kerala')
number of times kerala appears is 2
>>>
```

Program 76

Aim: To input a nested tuple and add name and mark and add that mark to that student

Modules used: N/A

Data types used: Tuple

Script:

```
t = eval(input(">>> "))
n = input("name: ")
m = int(input("mark: "))
o = ()

for i in t:
    if i[0] == n: append(m)
    o += (i,)

print(o)
```

Output:

```
>>> ([ 'nerd 46', 100], [ 'abigail', 90, 90])
name: nerd 46
mark: 100
(['nerd 46', 100, 100], [ 'abigail', 90, 90])
>>> |
```

Program 77

Aim: To input a tuple with numbers and print the numbers in reverse

Modules used: N/A

Data types used: Tuple, Str

Script:

```
o = ()  
for i in eval(input(">>> ")):   
    o += (str(i)[::-1],)  
print(f"tuple with reversed values: {o}")
```

Output:

```
>>> 35, 23, 10, 94  
tuple with reversed values: ('53', '32', '01', '49')  
>>>
```

Program 78

Aim: To input a tuple of numbers and print the sum of the digits of each number

Modules used: N/A

Data types used: Tuple, Str, Int

Script:

```
o = ()
for i in eval(input(">>> ")):
    s = 0
    for j in str(i): s += int(j)
    o += (s,)
print(f"tuple of sums: {o}")
```

Output:

```
>>> 123,321,213
tuple of sums: (6, 6, 6)
>>>
```

Program 79

Aim: To enter email id of students in a list and make a tuple containing usernames and one containing domains

Modules used: N/A

Data types used: Tuple, List, Str

Script:

```
l = []

u, d = (), ()

for i in range(int(input("enter n: "))):
    l.append(input(f"{i+1}. "))

for i in l:
    u += (i.split("@")[0],)
    d += (i.split("@")[1],)

print(f"tuple of usernames: {u}\ntuple of domains: {d}")
```

Output:

```
enter n: 4
1. asdf@fdsa.com
2. fdsa@asdf.com
3. rEee@rEee.com
4. starstalker9160@rEee.com
tuple of usernames: ('asdf', 'fdsa', 'rEee', 'starstalker9160')
tuple of domains: ('fdsa.com', 'asdf.com', 'rEee.com', 'rEee.com')
>>>
```

Program 80

Aim: To print a tuple containing the Fibonacci series

Modules used: N/A

Data types used: Tuple, Int

Script:

```
a, b, c = 0, 1, 0
t = (0,)
for i in range(0, int(input("How long the sequence should be: "))-1):
    a, b = b, c
    c = a+b
    t += (c,)
print(t)
```

Output:

```
>>> | How long the sequence should be: 5
      | (0, 1, 1, 2, 3)
```

Program 81

Aim: To input a nested tuple of pairs and add it to output tuple only if the pair is even

Modules used: N/A

Data types used: Tuple, Int

Script:

```
c = 0

for i in eval(input(">>> ")):
    if (i[0] % 2 == 0) and (i[1] % 2 == 0):
        c += 1

print(f"number of even pairs: {c}")
```

Output:

```
>>> (1,2), (2,2), (4,6)
number of even pairs: 2
>>> |
```


Program 82

Aim: To input a tuple of numbers and find mode

Modules used: N/A

Data types used: Tuple, Int

Script:

```
T = eval(input(">>> "))

M, C = None, 0

for i in T:
    c = 0
    for j in T:
        if i == j: c += 1
    if c > C:
        M = i
        C = c

print(f"mode: {M}")
```

Output:

```
>>> (1, 2, 3, 3, 4, 5, 3, 6, 7)
mode: 3
>>>
```

Program 83

Aim: To input a nested tuple and print sum of alternate elements

Modules used: N/A

Data types used: Tuple, Int

Script:

```
t = ()
s = 0

for i in eval(input(">>> ")):
    for j in i: t += (j,)

for i in range(0, len(t), 2):
    s += t[i]

print(f"sum of alternate elements is : {s}")
```

Output:

```
>>> (1,2,3), (4,5,6), (7,8,9)
sum of alternate elements is : 25
>>>
```

Program 84

Aim: To input a tuple and check if there are multiple maximum elements

Modules used: N/A

Data types used: Tuple, Int

Script:

```
t = eval(input(">>> "))
l = float('-inf')

for i in t:
    if i > l: l = i

c = 0
for i in t:
    if i == l: c += 1
    if c > 1:
        print("max elements is repeated")
        break
else:
    print("max element is not repeated")
```

Output:

```
>>> 1,2,3,4,
max element is not repeated
>>>
===== RESTART: D:\Schc
>>> 1,2,3,3,3
max elements is repeated
>>> |
```

Program 85

Aim: To input a tuple and check if minimum number is in the middle of the tuple

Modules used: N/A

Data types used: Tuple, Int

Script:

```
t = eval(input(">>> "))
s = float('+inf')

for i in t:
    if i < s: s = i

print("least element is in the middle") if t[int(len(t)/2)] == s else print ("least element is not in the middle")
```

Output:

```
>>> 1,2,3
least element is not in the middle
>>>
===== RESTART: D:\School Cod
>>> 3,1,3
least element is in the middle
>>> |
```

Program 86

Aim: To check whether the inputted tuple is sorted

Modules used: N/A

Data types used: Tuple, Int

Script:

```
t = eval(input(">>> "))
k, l = list(t), list(t)
l.sort()

print("tuple is sorted") if k == l else print("tuple is not sorted")
```

Output:

```
>>> 1,2,3
tuple is sorted
>>>
===== RESTART
>>> 4,2,5,1
tuple is not sorted
>>>
```