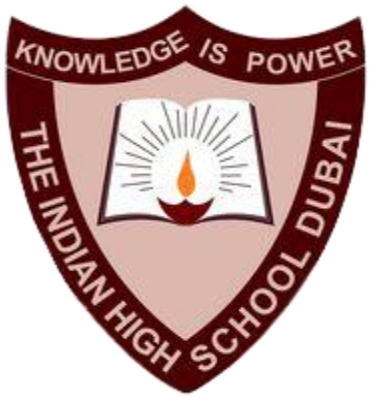
# THE INDIAN HIGH SCHOOL - DUBAI




## COMPUTER SCIENCE

## JOURNAL

## 2024-25

NAME: Abyaz Javid

Roll no.: 4

# INDEX

# CONTROL STRUCTURES & FLOW OF CONTROL

# 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\A
Enter two numbers seperated by a comma: 1, 2
Largest number: 2.0
Smallest number: 1.0
>>>
================ RESTART: D:\School Coding\11TH\A
Enter two numbers seperated by a comma: 2, 1
Largest number: 2.0
Smallest number: 1.0
>>>
================ RESTART: D:\School Coding\11TH\A
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: $1 + x^2 + x^3 \ldots + x^n$

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: $x - \frac{x^2}{2!} + \frac{x^3}{3!} - \frac{x^4}{4!} \ldots \pm \frac{x^n}{n!}$

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:

```
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
>>>
```

# STRING MANIPULATION

# 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:

```
s = input(">>> ")
a = 0
for i in s:
    if i == 'a':
        a += 1
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:

```
s = input(">>> ")

# pattern 1
for i in range(len(s)):
    print(s[i] * (i+1))

print()

# pattern 2
for i in range(len(s)):
    print(s[:i+1])

print()

# pattern 3
for i in range(len(s), 0, -1):
    print(s[:i])

print()

# pattern 4
for i in range(len(s), 0, -1):
    print(s[::-1][:i])

print()

# pattern 5
for i in range(1, len(s) + 1):
    print(s[:i] * i)

```

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:

```
w = input(">>> ").split()
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:

```
s = input(">>> ")
v = 0
for i in s:
    if i in "aeiouAEIOU":
        v += 1
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:

```
s = input(">>> ")
if s == s[::-1]:
    print(f"'{s}' is a palindrome")
else:
    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:

```
w = input("Enter word: ")
s = input("Enter sentence: ")
if w in s:
    print(f"yes, word is in sentence")
else:
    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:

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

for i in range(n):
    c = input(f"{i+1}. ")
    if len(c) > len(l):
        l = c

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:

```
n = int(input("Enter the number of strings: "))
s = None

for i in range(n):
    c = input(f"{i+1}. ")
    if s is None or len(c) < len(s):
        s = c

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:

```
s = input(">>> ")
a, d, ob = 0, 0, 0
for i in s:
    if i in "0123456789":
        d += 1
    elif i in "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ":
        a += 1
    else:
        ob += 1

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:

```
s = input(">>> ")
o = ""
for i in s:
    if "a" <= i <= "z":
        o += chr(ord(i) - 32)
    elif "A" <= i <= "Z":
        o += chr(ord(i) + 32)
    else:
         o += i
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:

```
s = input(">>> ")
o = chr(ord(s[0]) - 32)

for i in range(1, len(s)):
    if s[i] != " " and s[i-1] == " ":
        o += chr(ord(s[i]) - 32)
    else:
        o += s[i]

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:

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

Output:

```
>>> 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:

```
t = input("Enter text: ")
w = input("Enter word: ")
c = 0

for i in t.split():
    if i == w:
        c += 1

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:

```
s = input(">>> ").split()
o = input("Pick word to replace: ")
n = input(f"Pick word to replace '{o}' with: ")

o_ = ""
for i in s:
    if i == o:
        o_ += f"{n} "
    else:
        o_ += f"{i} "

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:

```
s = input(">>> ").split()
w = input("Pick word to reverse: ")

o_ = ""
for i in s:
    if i == w:
        o_ += f"{w[::-1]} "
    else:
        o_ += f"{i} "

print(o_[:-1])
```

Output:

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

# LISTS

# 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 L$^{th}$ 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*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*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*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*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*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  0  0
             ,-----  ____    0
            | PMD \_|[]|_'__Y
            |_______|__|_|__|}
==============oo--oo==oo--OOO\\====================
""")

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
           ,_____  ____   0
           | PMD \_|[]|_'__Y
           |_______|__|_|__|}
==========oo--oo==oo--OOO\\====================

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("\tenter item name: ")
        p = float(input("\tenter price: "))
        q = int(input("\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
>>>
```

# TUPLES

# 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:\Sch
    >>> 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
>>>

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

# DICTIONARIES

# Program 87

Aim: Create a dict with roll no., name and cs mark and print it neatly

Modules used: N/A

Data types used: Dict

Script:

```
d = {"roll no: ": int(input("roll no.: ")), "name": input("name: "), "cs mark": float(input("cs mark: "))}

print("{")
for a, s in d.items():
    print(" " * 4 + f"\"{a}\": ", end="")
    if type(s) is dict:
        print("{")
        for a, s in s.items():
            print(" " * (8) + f"\"{a}\": \"{s}\",")
        print(" " * 4 + "},")
    elif type(s) is list:
        print("[")
        for item in s:
            print(" " * (8) + f"\"{item}\",")
        print(" " * 4 + "],")
    elif type(s) is str:
        print(f"\"{s}\",")
    else:
        print(f"{s},")
print("}")
```

Output:

```
roll no.: 2
name: nerd 43
cs mark: 100
{
    "roll no: ": 2,
    "name": "nerd 43",
    "cs mark": 100.0,
}
>>>
```

# Program 88

Aim: Make a dictionary with n key-value pairs and print it

Modules used: N/A

Data types used: Dict

Script:

```
d = {input("k: "): input("v: ") for _ in range(int(input(">>> ")))}
print(d)
```

Output:

```
>>> 2
k: key1
v: value1
k: key2
v: value2
{'key1': 'value1', 'key2': 'value2'}
>>>
```

# Program 89

Aim: Seach a dictionary and print out value

Modules used: N/A

Data types used: Dict

Script:

```
d = {input("k: "): input("v: ") for _ in range(int(input("n: ")))}
a = input("Enter search term: ")
if a in d:
    print(d[a])
else:
    print("no such key exists")
```

Output:

```
n: 2
k: key1
v: value1
k: key2
v: value2
Enter search term: key1
value1
>>>
```

# Program 90

Aim: Count occurrences of words in sentences

Modules used: N/A

Data types used: Dict

Script:

```
s = input(">>> ")
d = {word: s.split().count(word) for word in set(s.split())}
print(d)
```

Output:

```
>>> hello my name is abyaz and abyaz is my name
{'name': 2, 'hello': 1, 'and': 1, 'my': 2, 'is': 2, 'abyaz': 2}
>>>
```

# Program 91

Aim: Get student names as key and set value as 2500

Modules used: N/A

Data types used: Dict

Script:

```
s = eval(input(">>> "))
d = {i: 2500 for i in s}
print(d)
```

Output:

```
>>> ["abyaz", "abbas", "nerd 34"]
{'abyaz': 2500, 'abbas': 2500, 'nerd 34': 2500}
>>>
```

# Program 92

Aim: Menu driven program for inventory management

Modules used: N/A

Data types used: Dict

Script:

```
print("\n")
print("\t#------------rEee------------#")
print("\t|  1. Show item's price      |")
print("\t|  2. Add item               |")
print("\t|  3. Update item            |")
print("\t|  4. Delete item            |")
print("\t|  5. Show master record     |")
print("\t|                            |")
print("\t|  6. Exit                   |")
print("\t#----------------------------#")

while True:
    o = input("> ")

    if o == "1":
        i = input("item name: ")
        if i in d:
            print(f"{d[i]} AED")
        else:
            print("item not found")

    elif o == "2":
        i = input("name: ")
        p = float(input("price: "))
        d[i] = p
        print("item added successfuly")

    elif o == "3":
        i = input("item name: ")
        if i in d:
            p = float(input("new price: "))
            d[i] = p
            print(f"item updated successfuly")
        else:
            print("item not found")

    elif o == "4":
        i = input("item name: ")
        if i in d:
            del d[i]
            print(f"item deleted successfuly")
        else:
            print("item not found")

    elif o == "5":
        if d:
            print("\nmaster record:")
            for i, p in d.items():
                print(f"\t{i}: {p} AED")

    elif o == "6":
        break

    else:
        print("INVALID INPUT")
```

Output:

```
        #------------rEee------------#
        |  1. Show item's price      |
        |  2. Add item               |
        |  3. Update item            |
        |  4. Delete item            |
        |  5. Show master record     |
        |                            |
        |  6. Exit                   |
        #----------------------------#
> 2
name: asdf
price: 123
item added successfuly
> 2
name: fdsa
price: 321
item added successfuly
> 3
item name: asdf
new price: 111
item updated successfuly
> 5

master record:
        asdf: 111.0 AED
        fdsa: 321.0 AED
> 4
item name: asdf
item deleted successfuly
> 5

master record:
        fdsa: 321.0 AED
> 6
>>>
```

# Program 93

Aim: Given student names and marks get topper

Modules used: N/A

Data types used: Dict, int, string

Script:

```
d = {}

for _ in range(int(input(">>> "))):
    n = input("\nname: ")
    m = eval(input("marks: "))
    s = 0
    for i in m:
        s += i
    d[n] = s

a = None
s = -float('inf')

for k, v in d.items():
    if v > s:
        s = v
        a = k

print(f"class topper is '{a}'")
```

Output:

```
>>> 2

name: nerd 91
marks: [100, 100, 100]

name: asdf
marks: [66, 4, 21]
class topper is 'nerd 91'
>>>
```

# Program 94

Aim: Get all trains going to Chennai (its flooded)

Modules used: N/A

Data types used: Dict

Script:

```
d = {input("train no.: "): input("stops: ").split() for _ in range(int(input(">>> ")))}

for k, v in d.items():
    if "chennai" in v:
        print(k)
```

Output:

```
>>> 2
train no.: 123
stops: ['chennai', 'singapoor']
train no.: 1231241231231241123
stops: ['not chennai', 'not singapoor']
123
>>>
```

# Program 95

Aim: Get item with minimum cost price and selling price

Modules used: N/A

Data types used: Dict

Script:

```
d = {}

f = ''
for i in range(int(input("n: "))):
    a = input("\nname: ")

    if i == 0:
        f = a

    cp = int(input("cp: "))
    sp = int(input("sp: "))

    d[a] = (cp, sp)

for i in d:
    _cp=d[f][0]
    _sp=d[f][1]

    M, m = f, f

    if d[i][0]>_cp:
        _cp=d[i][0]
        M = i

    if d[i][1]<_sp:
        _sp=d[i][1]
        m = i

print(f"\nmax sp: {M}\nmin cp: {m}")
```

Output:

```
n: 2

name: asdf
cp: 1
sp: 99

name: fdsa
cp: 99
sp: 1

max sp: fdsa
min cp: fdsa
>>>
```

# Program 96

Aim: Arrange marks of students in ascending order

Modules used: N/A

Data types used: Dict

Script:

```
d = {}

for i in range(int(input("n: "))):
    d[input("name: ")] = eval(input("marks: "))
for i in d:
    for j in range(len(d[i])):
        for k in range(j + 1, len(d[i])):
            if d[i][j] > d[i][k]:
                d[i][j], d[i][k] = d[i][k], d[i][j]

print(d)
```

Output:

```
n: 2
marks: [100, 100]
name: nerd 86
marks: [0, 9]
name: asdf
{'nerd 86': [100, 100], 'asdf': [0, 9]}
>>>
```

# Program 97

Aim: Print details of students who are in grade 11

Modules used: N/A

Data types used: Dict

Script:

```
d = {}

for i in range(int(input(">>> "))):
    s = {}

    key = input("number: ")
    s["name"] = input("name: ")
    s["class"] = input("class: ")
    s["gender"] = input("gender: ")

    d[key] = s

for i in d:
    if d[i]["class"] == "11":
        print(f"{i}: {d[i]}")
```

Output:

```
>>> 2
number: 1
name: asdf
class: 11
gender: f
number: 2
name: fdsa
class: 3
gender: yes
1: {'name': 'asdf', 'class': '11', 'gender': 'f'}
>>>
```

# Program 98

Aim: Print student who is rank 1

Modules used: N/A

Data types used: Dict

Script:

```
d = {}

for i in range(int(input(">>> "))):
    s = {}

    key = input("\nroll no.: ")
    s["name"] = input("name: ")
    s["mark"] = eval(input("marks: "))

    d[key] = s

m = 0

for i in d:
    s = 0
    for j in d[i]["mark"]:
        s += j
    if s > m:
        m = s
        n = d[i]["name"]
        r = i

print(f"\n{r}: '{n}'")
```

Output:

```
>>> 2

roll no.: 1
name: asdf
marks: [2, 31]

roll no.: 2
name: nerd 44
marks: [100, 100]

2: 'nerd 44'
>>>
```

# Program 99

Aim: Voting system for 3 class representative

Modules used: N/A

Data types used: Dict

Script:

```
d = {}

for i in range(int(input(">>> "))):
    r = input("\nroll no.: ")
    d[r] = input("vote: ")
    if d[r] not in 'abc':
        print("nuh uh")
        del d[r]

a = 0
b = 0
c = 0

for i in d:
    if d[i] == "a":
        a += 1
    elif d[i] == "b":
        b += 1
    elif d[i] == "c":
        c += 1

if a > b and a > c:
  print("a won")
elif b > a and b > c:
  print("b won")
else:
  print("c won")
```

Output:

```
>>> 4

roll no.: 1
vote: d
nuh uh

roll no.: 2
vote: a

roll no.: 3
vote: a

roll no.: 4
vote: c
a won
>>>
```

# Program 100

Aim: Find people who live in Sharjah

Modules used: N/A

Data types used: Dict

Script:

```
d = {input("\nname: "): input("number: ") for _ in range(int(input(">>> ")))

print("\nppl that live in sharjah")
for i in d:
    if d[i][0:2]=='06':
        print(f"\t- {i}")
```

Output:

```
>>> 2

name: asdf
number: 061234

name: fdsa
number: 123123

ppl that live in sharjah
        - asdf
>>>
```

# Program 101

Aim: Get employees born in a given month

Modules used: N/A

Data types used: Dict

Script:

```
d = {}
for i in range(int(input(">>> "))):
    s1 = {}
    ecode = input("\nemploy code: ")
    s1["name"] = input("name: ")

    s2 = {}
    s2["dd"] = int(input("DD: "))
    s2["mm"] = int(input("MM: "))
    s2["yyyy"] = int(input("YYYY: "))

    s1["dob"] = s2
    d[ecode] = s1

m = int(input("month: "))

print("\n")
for i in d:
    if d[i]["dob"]["mm"] == m:
        print(f"{i}: {d[i]['name']}")
```

Output:

```
>>> 2

employ code: 101
name: asdf
DD: 3
MM: 3
YYYY: 3333

employ code: 102
name: fdsa
DD: 1
MM: 1
YYYY: 1111
month: 3


101: asdf
>>>
```

# Program 102

Aim: Count occurrences of characters in a sentence

Modules used: N/A

Data types used: Dict

Script:

```
s = input("string: ")
char = input("character: ")
d = {}
c = 0

for i in s:
    if i == char:
        c += 1

d[char] = c
print(d)
```

Output:

```
string: my name is abyaz and abyaz is my name
character: a
{'a': 7}
>>>
```

# Program 103

Aim: Compare teams based on wins and losses

Modules used: N/A

Data types used: Dict

Script:

```
d = {input("\nname: "): eval(input("no. of losses and wins: ")) for _ in range(int(input(">>> ")))}

print("\nteams with more wins than losses: ")
for i in d:
    if d[i][0]>d[i][1]:
        print(f"\t- {i}")
```

Output:

```
>>> 2

name: asdf
no. of losses and wins: [232, 5]

name: fdsa
no. of losses and wins: [34, 44]

teams with more wins than losses:
        - asdf
>>>
```

# Program 104

Aim: KG1 admission eligibility

Modules used: N/A

Data types used: Dict

Script:

```
d = {}

for i in range(int(input(">>> "))):
    s = {}

    no = input("\napplication no.: ")
    s["name"] = input("name: ")
    s["dob"] = eval(input("dob [dd, mm, yyyy]: "))
    s["mark"] = int(input("total mark: "))

    d[no] = s

print("\naccepted students")
for i in d:
    if 2016 <= d[i]["dob"][2] >= 2017 and d[i]["mark"] > 50:
        print(f"\t- {d[i]['name']}")
```

Output:

```
>>> 2

application no.: 101
name: asdf
dob [dd, mm, yyyy]: [11, 4, 2016]
total mark: 51

application no.: 102
name: nerd jr
dob [dd, mm, yyyy]: [11, 4, 2017]
total mark: 100

accepted students
        - nerd jr
>>>
```