14/08/25

① Modify following program with walrus operator.

Hint :- Combine lines 7, 8 and 10 to a single line with walrus operator.

```
a = 'Hyd is green city. Hyd is hitec city. Hyd is his city'
index = a.find('is')
while index != -1:
    print(index)
    index = a.find('is', index+1)
print('End')
```

```
a = 'Hyd is green city. Hyd is hitec city. Hyd is his city'
while (index := a.find('is', index+1) if 'index' in locals() else 0) != -1:
    print(index)
print('End')
```

② index() method demo program

Modify the following program with index() method above

```
a = 'Hyd is green city. Hyd is hitec city. Hyd is his city'
try:
    index = a.index('is')
    while True:
        print(index)
        index = a.index('is', index+1)
except ValueError:
    print('End')
```

③ find() method

Modify following program with rfind method

```
a = 'Hyd is green city. Hyd is hitec city. Hyd is his city'
index = a.rfind('is')
while index != -1:
    print(index)
    index = a.rfind('is', 0, index) # search from left to right the last found position
print('End')
```

④ rindex() method demo program

Modify following program with rindex program

Hint: use try and except

```
a = 'Hyd is green city. Hyd is hitec city. Hyd is his city'
try:
    index = a.rindex('is')
    while True:
        print(index) ('is', 0, index) # search before current index
except ValueError:
    print('End')
```

⑤ # count() method demo program

```
a = 'Hyd is green city. Hyd is hitec city. Hyd is his city'
print(a.count('is'))        # 4  count total occurances of 'is'
print(a.count('is', 19, 48)) # 48
print(a.count('was'))       # 0 # count occurances of 'was' not present
```

⑥ find outputs

```
a = 'Hyd Is It green In city. Hyd is Ith itecln city. Hyd is Ithis In city'
print (a.count (' ')) # 4
print (a.count ('It')) # 3.
print (a.count ('In')) # 3
```

⑦ Write a program to replace every occurance of first character in the string with '*' except first character

Let input be 'babble'

What is the output ? ---> ba**le

```
s = babble
first_chas = s[0]
result = first_char + s[1:].replace (first_char, '*')
print result
```

⑧ find outputs

```
a = '15:36:48'
print (a.split (':')) # ['15', '36', '48']
print (a). # 15:36:48
```

⑨ find output

```
a = 'Hyd In is green It city'
print (a.split (''))  # ['Hyd' In is', 'green It city']
print (a.split ())    # ['Hyd', 'is', 'green', 'city']
print (a.split ('green')) # ['Hyd In is', 'It city']
print (a.split (''))  # Error split () cannot take an empty string separator.
```

(10) find output

```
a = 'Hyd' is green city'.  # There is tab b/w the words
print (a.split('It'))  # ['Hyd', 'is', 'green', 'city']
print (a.split())  # ['Hyd', 'is', 'green', 'city']
print (a.split(''))  # ['Hyd It is It green It city]
```

(11) find output

```
a = 'Hyd is green city'  # 3 spaces b/w the words
print (a.split())  # ['Hyd', 'is', 'green', 'city']
print (a.split(''))  # ['Hyd', '', '', 'is', '', '', 'green
                     '', '', 'city
```

(12) find outputs

```
a = 'www.gmail.com'
print (a.split('.'))  # ['www', 'gmail', 'com']
```

(13) Write a program to evalute an expression which contain only + symbols.

```
let input be 23+45+6+789

expr = input ("Enter an expression with only '+': ")
parts = expr.split ('+')
total = 0
for p in parts
    total += int(p)
print ("sum =", total)
```

find outputs

```
a=['15','36','48']
print(':'.join(a))                    # 15:36:48
b=('Hyd','is','green','city')
print(' '.join(b))                    # Hyd is green city
c={'10','20',15,25,'52'}
print(' '.join(c))                    # 25 10 20 15 52
d=['www','gmail','com']
print('.'.join(d))                    # www.gmail.com
e=[15,36,48]
print(':'.join(e))                    # all element should be in string
f=('Sankar','Dayal','Sarma')
print(''.join(f))                     # Sankar Dayal sarma
g=range(5)
print('_'.join(g))                    # gives string not an integer
```

endswith()

```
a='Hyd is green city'
print(a.endswith('city'))             # True
print(a.endswith('town'))             # false
print(a.endswith('green',3,12))       # True
print(a.endswith('green',3,13))       # False
print(a.endswith('',3,13))            # True
```

16. Write a program to append 'ing' to input string

```
s=input("Enter a string!")
if len(s)<3:
    result=s
```

```
elif s.endswith("ing"):
    result = s + "ly"
else:
    result = s + "ing"
print(result)
```

17) isalpha() method demo program

```
print('H4d'.isalpha())      # True
print('Rama Rao'.isalpha()) # False
print('H4d4'.isalpha())     # False
print('H4d$'.isalpha())     # False
print('9247'.isalpha())     # False
print('+-$'.isalpha())      # False
print('A2#'.isalpha())      # False
print("".isalpha())         # False
print(' '.isalpha())        # False
```

18) isdigit() method demo program

```
print('9247'.isdigit())     # True
print('92a47'.isdigit())    # False
print('92$47'.isdigit())    # False
print('H4d'.isdigit())      # False
print('+-$'.isdigit())      # False
print('A2#'.isdigit())      # False
print("".isdigit())         # False
print(' '.isdigit())        # False
print(9247.isdigit())       # Error -> integers don't have isdigit()
```

(19) # isupper()

```
print ('H4d'.isupper())      # False -> Becaus 'd' is lower
print ('H4D'.isupper())      # True -> All letters are upper
print ('9247'.isupper())     # False -> No letter at all
print ('RAMA RAO'.isupper()) # True -> All letters are upper
print ('+-$'.isupper())      # False -> No letters
print ('H4D128'.isupper())   # True -> All letters are upper
print ('H4D+-$'.isupper())   # True -> ..  ..  ..
print (''.isupper())         # False -> Empty string -> no letter
print ('A2#!'.isupper())     # True -> 'A' is uppercase
```

(20) islower() method

```
print('h4D'.islower())       # False -> 'D' is upper case
print('h4d'.islower())       # True -> All letter are lowercase
print('9247'.islower())      # False -> No -> letter -> return False
print('rama rao'.islower())  # True -> All letter uppercase
print('+-$'.islower())       # False -> No letter
print('h4d+-$'.islower())    # True -> All letter, lowercase
print('abc123'.islower())    # True -> All letter lowercase
print(''.islower())          # False -> Empty string
print('a2#'.islower())       # True -> 'a' is lowercase
```

(21) print (" isalnum()

```
print('A7$g'.isalnum())      # False -> '$' is not a letter
print('H4D'.isalnum())       # True -> All are letter
print('+-$'.isalnum())       # False -> only symbol
print('h4d'.isalnum())       # True -> All are letter
print('hYd'.isalnum())       # True -> All are letter
```

print('9247'.isalnum()) # False → Empty string All are letters

print(''.isalnum()) # False → empty string

(22) # isspace()

print('\n Alt'.isspace()) # False → 'A' is not space

print('\n\t'.isspace()) # True → only spaces → all whitespace

print('\n 7\t'.ispace()) # False → '7' is not white space

print('\n'.isspace()) # True → only newline → white line

print('\n$\t'.isspace()) # False → '$' is not white

print('\t'.isspace()) # True → only tab → whitespace

print(''.isspace()) # False → Empty string → no characters

print(' '.isspace()) # True → single space → whitespace

(23) # Find outputs

a,b,c = 25, 10.8, 'Hyd'

print('a: {} \t b: {} \t c: {}'.format(a,b,c))

a: 25 b: 10.8 c: Hyd → # {} → insert values in order

print('a: {0} \t b: {1} \t c: {2}'.format(a,b,c))

a: 25 b: 10.8 c: Hyd → explicitly index arguments

print('a: {2} \t b: {1} \t c: {0}'.format(a,b,c))

a: Hyd (25) b: 10.8 c: 25 (Hyd) → # positions swapped by index

print('a: {2} \t b: {2} \t c: {2}'.format(a,b,c))

a: Hyd b: Hyd c: Hyd → All placeholders

print('a: {x} \t b: {y} \t c: {z}'.format(x=a, y=b, z=c))

a: 25 b: 10.8 c: Hyd → Named placeholders with match keys

print('a: {x} \t b: {y} \t c: {z}'.format(z=a, y=b, x=c))

a: Hyd b: 25 c: 25 → Names assigned different values

print('a: {z} \t b: {z} \t c: {z}'.format(z=a, y=b, x=c))

a: 25 b: 25 c: 25 # All alpha placeholders

(24) Write a program to determine user input is alphabet, digit, white space or special character.

```
ch = input("Enter any character:")
if ch.isalpha():
    print("Alphanumeric character")
    print("Alphabet character")
    if ch.isupper():
        print("upper case alphabet")
    else:
        print("lower case alphabet")
elif ch.isdigit():
    print("Alphanumeric character")
    print("digit character")
elif ch.isspace():
    print("white space")
else
    print("special char")
```

(25) Write a program to reverse a string without slice

```
a = input("Enter any string:")
b = ""
for i in range(1, len(a)+1):
    b = b+a[-i]
print("Reverse string:", b)
```

(26) Write a program to reverse order of words in the sentence without slice.

```
a = input("Enter a sentence:")
b = a.split()
c = ""
for i in range(1, len(b)+1):
    c = c+b[-i]+" "
print("Reversed sentence:", c.strip())
```

(27) Write a program to reverse each word of the sentence

```
a= input ("Enter a sentence:")
b = a.split ()
c = " "
for word in b:
    c = c+ word [::-1] + " "
print ("Reversed each word:", c.strip())
```

(28) Write a program to sort string in alphabets order

Let input be RAJESh

```
a = input ("Enter a string:")
b = sorted (a)
c = "". join (b)
print ("sorted string:", c)
```

(29) Write a program to sort string such that alphabets in alphabetical order and digits in ascending order.

```
S = input ("Enter a string:")
letter:[]
digits =[]
for ch in s:
    if ch. isalpha ():
        letter. append (ch)
    elif ch. isdigit ():
        digit. append (ch)
letter. sort (key = str. lower)
digit. sort (key = int)
sorted_string = "". join (letters + "". join (digits)
print ("sorted string:", sorted_string)
```