

## Assignment 1 : Fundamentals

1. Write a Python script to complement the binary combination of decimal 126.

**Code :**

```
n = int(input("Enter the number to print compliment of binary : "))
print("The binary equivalent of {} : {}".format(n, bin(~n)))
```

**Output:**

```
Set Bits Count(give in decimal) : 4
```

2. Write a Python script to print hex and binary value of 100.

**Code:**

```
n = int(input("Enter the number to print Hex and Bin value : "))
print("100 in hex : {} and in bin {}".format(hex(n), bin(n)))
```

**Output:**

```
Enter the number to print Hex and Bin value : 100
100 in hex : 0x64 and in bin 0b1100100
```

3. Write a Python script to take a floating point number upto 6 decimal places and print the equivalent 3 decimal.

**Code:**

```
f_num = int(input("Enter custom float number atleast 8 decimals or use pre defined input (24.0436346456) by typing 0 : "))
if f_num != 0:
    print("upto 6 decimals {:.6f} and 3 decimal equivalent {:.3f}".format(f_num, f_num))
else:
    f_num = 24.0436346456
    print("upto 6 decimals {:.6f} and 3 decimal equivalent {:.3f}".format(f_num, f_num))
```

**Output:**

```
Enter custom float number atleast 8 decimals or use pre defined input (24.0436346456) by typing 0 : 0
upto 6 decimals 24.043635 and 3 decimal equivalent 24.044
```

4. How can you represent binary literals in Python?

**Code:**

```
n = int(input("Enter the number for binary literal : "))
```

```
print("binary literal : {} \n".format(bin(n)))
```

**Output:**

```
Enter the number for binary literal : 15
binary literal : 0b1111
```

5. Write a Python code to check if the second bit of a binary number is set or not.

**Code:**

```
def getlthBit():
    b = int(input("Enter the number : "))
    if b & (1 << 1) != 0:
        return 1
    return 0
```

```
x = getlthBit()
```

```
def checkBit(x):
    if(x):
        print("The bit is set !!\n")
    else:
        print("The bit is not set !!\n")
```

```
checkBit(x)
```

**Output:**

```
Enter the number : 5
The bit is not set !!
```

6. Write a Python program to convert a decimal number to binary.

**Code:**

```
n = int(input("Enter the decimal number : "))
print(bin(n))
```

**Output:**

```
Enter the decimal number : 12
0b1100
```

7. Create a Python program to perform bitwise OR operation on two binary numbers.

**Code:**

```
a = int(input("Enter the number 1 : "))
b = int(input("Enter the number 2 : "))
print(bin(a | b))
```

**Output:**

```
Enter the number 1 : 9
Enter the number 2 : 3
0b1011
```

8. Write a Python function to count the number of set bits (1s) in a binary number.

**Code:**

```
def countSetBits(d):
    count = 0

    while d > 0:
        if d & 1 == 1:
            count += 1
        d = d >> 1

    return count

print("Set Bits Count(give in decimal) : {}".format(countSetBits(15)))
```

**Output:**

```
Set Bits Count(give in decimal) : 4
```

9. Create a Python function to check if a binary number is a palindrome or not.

**Code:**

```
binary_num = bin(11)
binary_str = binary_num[2:]
binary_num_rev = binary_str[::-1]

def checkBinPalindrome(binary_str):
    if binary_num_rev == binary_str:
        return True
```

```
return False
```

```
print("Is Palindrome : {}".format(checkBinPalindrome(binary_str)))
```

**Output:**

```
Is Palindrome : False
```

10. Write a Python program to read the contents of a text file and display them.

**File:**

```
# cat textfile.txt
File for testing python codes
```

**Code:**

```
fid = open('textfile.txt','r')
data = fid.read(-1)
fid.close()
```

```
print(data)
```

**Output:**

```
File for testing python code
```

11. Create a Python program to write a list of strings to a text file. Take each element of the list from user input and put them in separate line.

**Code:**

```
def write_list_to_file(string_list):
    with open("str.txt", 'w') as file:
        for item in string_list:
            file.write(item + '\n')

n = int(input("Enter the number of strings :"))
my_list = []
for i in range(n):
    element = input("Enter the string : ")
    my_list.append(element)

write_list_to_file(my_list)
```

```

fid = open('str.txt','r')
data = fid.read(-1)
fid.close()

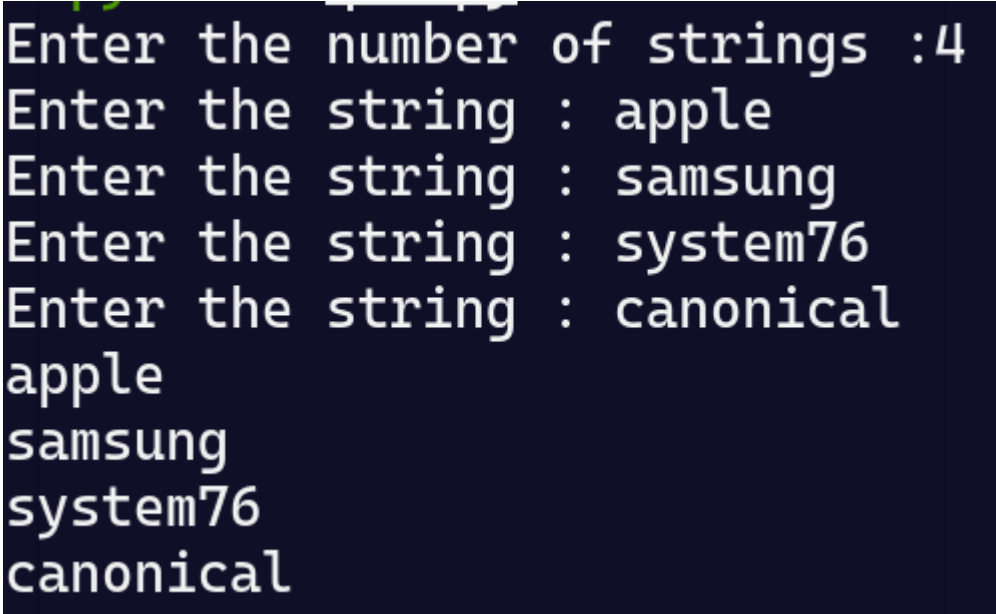
```

```

print(data)

```

**Output:**



```

Enter the number of strings :4
Enter the string : apple
Enter the string : samsung
Enter the string : system76
Enter the string : canonical
apple
samsung
system76
canonical

```

12. Write a Python function to count the number of lines in a text file.

**Code:**

```

fid = open('textfile.txt','r')
lineCount = len(fid.readlines())
fid.close()

```

```

print(lineCount)

```

**Output:**



```

1

```

13. Implement a Python program to copy the contents of one text file to another.

**Code:**

```

with open('str.txt', 'r') as fid:
    data = fid.read()

```

```

with open('thirt.txt', 'w') as fid_w:
    fid_w.write(data)

```

```
with open('thirt.txt', 'r') as fid_r:  
    data = fid_r.read()
```

```
print(data)
```

**Output:**

```
apple  
samsung  
system76  
canonical
```

14. Create a Python program to display the result of a mathematical operation using formatted strings.

**Code:**

```
print("Summation of {} and {} is : {}".format(2,4,2+4))
```

**Output:**

```
Summation of 2 and 4 is : 6
```

## Assignment 2 : Number String

1. Write a program to calculate the area of a rectangle. Take inputs from user.

**Code:**

```
l = 4
```

```
b = 8
```

```
print("Area of rectangle of length {} and breadth {} is : {}".format(l,b, l * b))
```

**Output:**

```
Area of rectangle of length 4 and breadth 8 is : 32
```

## 2. Convert Celsius to Fahrenheit

**Code:**

```
C = 32
F = C * 9/5 + 32
print("Fahrenheit : {}".format(F))
```

**Output:**

```
Fahrenheit : 89.6
```

## 3. Find the Square Root of a Number

**Code:**

```
import math

n = 17
print('{:.4f}'.format(math.sqrt(n)))
```

**Output:**

```
4.1231
```

## 4. Check if a Number is Even or Odd

**Code:**

```
n = 4
print("Even" if n % 2 == 0 else "Odd")
```

**Output:**

```
Even
```

## 5. Find the Maximum of Three Numbers

**Code:**

```
a = int(input("Enter the first number: "))
b = int(input("Enter the second number: "))
c = int(input("Enter the third number: "))
```

```
if (a > b and a > c):
    print("{} is the greatest".format(a))
elif (b > a and b > c):
    print("{} is the greatest".format(b))
else:
    print("{} is the greatest".format(c))
```

**Output:**

```
Enter the first number: 4
Enter the second number: 8
Enter the third number: 2
8 is the greatest
```

6. Removing whitespace from a String “ Python String ”.

**Code:**

```
string = "Python string"
new_string = string.replace(" ", "")
print(new_string)
```

**Output:**

```
Pythonstring
```

7. Check if a string contains a substring

**Code:**

```
#check if a string contains substring
def check_substring(main_string, substring):
    if substring in main_string:
        return True
    else:
        return False
```

# Example usage:

```
main_string = "Hello, world!"
substring = "world"
```

```
if check_substring(main_string, substring):
    print("Substring found!")
else:
    print("Substring not found.")
```

**Output:**

```
Substring found!
```

8. Reverse the string “RCCIIT ECE”

**Code:**



```
a = "RCCIIT ECE"
reverse = a[::-1]
print("The reversed string is {}".format(reverse))
```

**Output:**

```
The reversed string is ECE TIICCR
```

9. Count the occurrences of a substring in a string  
sentence = "Python is easy, Python is powerful, Python is fun"  
substring = "Python"

**Code:**

```
#Counting the substring in a sentence
sentence = "Python is easy, Python is powerful, Python is fun"
substring = "Python"
a = sentence.count(substring)
print(a)
```

**Output:**

```
3
```

10. Count Vowels in a String = "RCCIIT ECE"

**Code:**

```
string = "RCCIIT ECE"
```

```
count = 0
```

```
i = 0
```

```
for i in range (len(string)):
```

```
    if((string[i] == "A") or (string[i] == "E") or (string[i] == "I") or (string[i] == "O") or (string[i] ==
    "U")):
```

```
        count +=1
```

```
print("The count is {}".format(count))
```

**Output:**

```
The count is 4
```

11. Check the string anagrams (contain the same characters with the same frequency)

**Code:**

```
def check(s1,s2):  
    if (sorted(s1) == sorted(s2)):  
        print("its anagram")  
    else:  
        print("Its not anagram")
```

```
s1= "dad"  
s2 = "bad"  
check(s1,s2)
```

**Output:**

Its not anagram

## Assignment 3 : List

1. Write a function to calculate the sum of all elements in a list.

**Code:**

```
L = [1, 2, 3, 4]  
print(sum(L))
```

**Output:**

10

2. Find the maximum element from a list

**Code:**

```
L = [1, 2, 3, 4]  
print(max(L))
```

**Output:**

4

3. Reverse a list

**Code:**

```
L = [1, 2, 3, 4]  
L.reverse() #L = L[::-1]
```

```
print(L)
```

**Output:**

```
[4, 3, 2, 1]
```

#### 4. Remove duplicate from a list

**Code:**

```
L = [1, 2, 2, 3, 4, 3, 5, 2, 1]
uL = []
```

```
for item in L:
    if item not in uL:
        uL.append(item)
print(uL)
```

**Output:**

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

#### 5. Check palindrome list

**Code:**

```
def palindrome(L):
    return "Palindrome" if L == L[::-1] else "Not Palindrome"
```

```
L = [1, 2, 1]
print(palindrome(L))
```

**Output:**

```
Palindrome
```

#### 6. Write a function to find the intersection of two lists.

**Code:**

```
L = [1, 2, 3, 4, 5, 6]
L2 = [3, 4, 6, 1]
```

```
L3 = [item for item in L if item in L2]
print(L3)
```

**Output:**

```
[1, 3, 4, 6]
```

7. Write a function to flatten a nested list.

**Code:**

```
def flatten_list(nested_list):  
    return [item for sublist in nested_list for item in (flatten_list(sublist) if isinstance(sublist, list)  
    else [sublist])]
```

```
nested_list = [1, [2, 3], [4, [5, 6, 7]]]  
result = flatten_list(nested_list)  
print(result)
```

**Output:**

```
[1, 2, 3, 4, 5, 6, 7]
```

8. Write a function to rotate a list to the left by a given number of positions.

**Code:**

```
def rotate_left(lst, positions):  
    positions %= len(lst) # Handle cases where positions is greater than the length of the list  
    return lst[positions:] + lst[:positions]
```

```
my_list = [1, 2, 3, 4, 5]  
rotated_list = rotate_left(my_list, 8)  
print(rotated_list)
```

**Output:**

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

## Assignment 4 : Tuple

1. Create a tuple with elements (1, 2, 3) and print its length.

**Code:**

```
t = (1, 2, 3)  
print(len(t))
```

**Output:**

```
3
```

2. Concatenate two tuples (4, 5, 6) and (7, 8, 9).

**Code:**

```
t1 = (4, 5, 6)
t2 = (7, 8, 9)
t = t1 + t2
print(t)
```

**Output:**

```
(4, 5, 6, 7, 8, 9)
```

3. Check if the value 5 is present in the tuple (1, 2, 3, 4, 5).

**Code:**

```
t = (1, 2, 3, 4, 5)
result = "Exists" if 5 in t else "Do not Exist"
print(result)
```

**Output:**

```
Exists
```

4. Find the maximum element in the tuple (8, 3, 6, 2, 7).

**Code:**

```
t = (8, 3, 6, 2, 7)
print(max(t))
```

**Output:**

```
8
```

5. Create a tuple with the elements ('apple', 'banana', 'cherry') and convert it to a list.

**Code:**

```
t = ('apple', 'banana', 'cherry')
l = list(t)
print(l)
```

**Output:**

```
['apple', 'banana', 'cherry']
```

6. Count the number of occurrences of the value 2 in the tuple (1, 2, 3, 2, 4, 2, 5).

**Code:**

```
t = (1, 2, 3, 2, 4, 2, 5)
print(t.count(2))
```

**Output:**

3

7. Given a tuple (10, 20, 30, 40, 50), slice it to get the sub-tuple (20, 30, 40).

**Code:**

```
t = (10, 20, 30, 40, 50)
t = t[1:-1]
print(t)
```

**Output:**

(20, 30, 40)

8. Find the index of the value 3 in the tuple (1, 2, 3, 4, 5).

**Code:**

```
t = (1, 2, 3, 4, 5)
print(t.index(3))
```

**Output:**

2

9. Repeat the tuple (6, 7, 8) three times.

**Code:**

```
t = (6, 7, 8)
t = t*3
print(t)
```

**Output:**

(6, 7, 8, 6, 7, 8, 6, 7, 8)

10. Given two tuples (1, 2, 3) and ('a', 'b', 'c'), create a new tuple by combining them.

**Code:**

```
t1 = (1, 2, 3)
t2 = ('a', 'b', 'c')
t = t1 + t2
print(t)
```

**Output:**

```
(1, 2, 3, 'a', 'b', 'c')
```

## Assignment 5 : Dictionary

1. How to iterate through all keys and values in a dictionary?

**Code:**

```
x = {'key1': 'RCCIIT', 'key2': 1999, 'key3': 3.14}
for key, value in x.items():
    print("Key:", key, "Value:", value)
```

**Output:**

```
Key: key1 Value: RCCIIT
Key: key2 Value: 1999
Key: key3 Value: 3.14
```

2. How to remove a key from a dictionary?

**Code:**

```
x = {'key1': 'RCCIIT', 'key2': 1999, 'key3': 3.14}
x.pop('key2')
print(x)
```

**Output:**

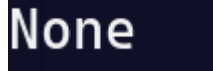
```
{'key1': 'RCCIIT', 'key3': 3.14}
```

### 3. How to check the existence of key in a dictionary?

**Code:**

```
x = {'key1': 'RCCIIT', 'key2': 1999, 'key3': 3.14}
print(x.get('key4'))
```

**Output:**



None

### 4. How to merge two dictionaries?

**Code:**

```
x = {'key1': 'RCCIIT', 'key2': 1999, 'key3': 3.14}
y = {'key4': 'ECE', 'key5': 'A'}
x.update(y)
print(x)
```

**Output:**



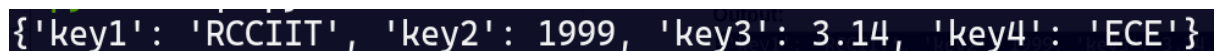
```
{'key1': 'RCCIIT', 'key2': 1999, 'key3': 3.14, 'key4': 'ECE', 'key5': 'A'}
```

### 5. How to create a dictionary with default values?

**Code:**

```
x = {'key1': 'RCCIIT', 'key2': 1999, 'key3': 3.14}
x.setdefault('key4', 'ECE')
print(x)
```

**Output:**



```
{'key1': 'RCCIIT', 'key2': 1999, 'key3': 3.14, 'key4': 'ECE'}
```

### 6. Write a Python function to find the key(s) with the minimum value in a dictionary.

**Code:**

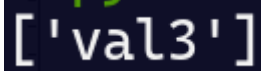
```
dict = {"val1": 2, "val2": 5, "val3": 1, "val4": 3, "val5": 4}
```

```
temp = min(dict.values())
res = [key for key in dict if dict[key] == temp]
```

```
print(res)
```

**Output:**





```
['val3']
```

7. Write a Python function to group a list of dictionaries by a common key. `[{'name': 'ABCD', 'group': 'ECE'}, {'name': 'EFGH', 'group': 'CSE'}, {'name': 'IJKL', 'group': 'ECE'}]` has to be converted as `{'ECE': [{'name': 'ABCD'}, {'name': 'IJKL'}], 'CSE': [{'name': 'EFGH'}]}`

**Code:**

```
data_list = [
    {'name': 'ABCD', 'group': 'ECE'},
    {'name': 'EFGH', 'group': 'CSE'},
    {'name': 'IJKL', 'group': 'ECE'}
]

key = 'group'
grouped_dict = {}
for item in data_list:
    group_key = item[key]
    item.pop(key)
    grouped_dict.setdefault(group_key, []).append(item)

print(grouped_dict)
```

**Output:**

```
{'ECE': [{'name': 'ABCD'}, {'name': 'IJKL'}], 'CSE': [{'name': 'EFGH'}]}
```

8. Write a Python function to find the intersection of two dictionaries.  
Input : dict1 = {'key1': 1, 'key2': 2, 'key3': 3}, and dict2 = {'key2': 2, 'key3': 30, 'key4': 4} Output : {'key2': (2, 2), 'key3': (3, 30)}

**Code:**

```
# Example usage:
dict1 = {'key1': 1, 'key2': 2, 'key3': 3}
dict2 = {'key2': 2, 'key3': 30, 'key4': 4}

common_keys = sorted(dict1.keys() & dict2.keys())
intersection_dict = {key: (dict1[key], dict2[key]) for key in common_keys}

print(intersection_dict)
```

**Output:**

```
{'key2': (2, 2), 'key3': (3, 30)}
```

9. Write a Python function to interchange keys & values of a dictionary.

**Code:**

```
og_dict = {'key1': 1, "key2":2}
```

```
res = {value:key for key, value in og_dict.items()}
```

```
print(res)
```

**Output:**

```
{1: 'key1', 2: 'key2'}
```

10. Find the "n" most frequent elements in a list using python dictionary.

**Code:**

```
from collections import Counter
```

```
input_list = [3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5, 4, 4, 1]
```

```
n = 3
```

```
element_counts = Counter(input_list)
```

```
final_dict = {key: value for key, value in element_counts.items() if value == n}
```

```
sorted_final_dict = dict(sorted(final_dict.items()))
```

```
print(sorted_final_dict)
```

**Output:**

```
{1: 3, 4: 3, 5: 3}
```