**Module 1**

## Case Study

1. Write a program which will find factors of given number and find whether the factor is even or odd. Hint: Use Loop with if-else statements
2. Write a code which accepts a sequence of words as input and prints the words in a sequence after sorting them alphabetically. Hint: In case of input data being supplied to the question, it should be assumed to be a console input.
3. Write a program, which will find all the numbers between 1000 and 3000 (both included) such that each digit of a number is an even number. The numbers obtained should be printed in a comma separated sequence on a single line. Hint: In case of input data being supplied to the question, it should be assumed to be a console input. Divide each digit with 2 and verify is it even or not.
4. Write a program that accepts a sentence and calculate the number of letters and digits. Suppose if the entered string is: Python0325 Then the output will be: LETTERS: 6 DIGITS:4 Hint: Use built-in functions of string.
5. Design a code which will find the given number is Palindrome number or not. Hint: Use built-in functions of string.

## 1. Write a program which will find factors of a given number and also find whether the factor is even or odd. Hint: Use Loop with if-else statements.

The first step to make this program would be to obtain a number from the user through an input, this can be accomplished with the following code using the console input.

# Input from the user

num = int(input("Get the factors of a number: "))

Second step would be to write a function that breaks down the number into factors and returns an array.

# Return array of factors

def find\_factors(num):

factors = []

for i in range(1, num + 1):

if num % i == 0:

factors.append(i)

return factors

Third step is to write a function to loop through an array of factors to determine if a factor is odd or even.

# is number even or odd?

def check\_even\_odd(factors):

even\_odd = {}

for factor in factors:

if factor % 2 == 0:

even\_odd[factor] = 'Even'

else:

even\_odd[factor] = 'Odd'

return even\_odd

Fourth step is to execute the functions.

# Find factors

factors = find\_factors(num)

# Check if factors are even or odd

even\_odd\_factors = check\_even\_odd(factors)

Last step is to print the results.

# Display each factor and parity (oddness or evenness)

for factor, parity in even\_odd\_factors.items():

print(f"Factor: {factor}, {parity}")

## 2. Write a code which accepts a sequence of words as input and prints the words in a sequence after sorting them alphabetically. Hint: In case of input data being supplied to the question, it should be assumed to be a console input.

The first step to make this program would be to obtain string of words using the console input and assign that input data to a variable named input\_words.

# Input from the user

input\_words = input("Enter sequence of words separated by spaces: ")

Second step is to use a method that breaks up the words using the space character to determine where the words start and end.

# Split the input string into words

words = input\_words.split()

Third step is to write a function that sorts the words and returns the words after sorting them.

# Function to sort words alphabetically

def sort\_words(words):

return sorted(words)

Fourth step is to execute the sort function with the words derived from the variable input\_words from the user.

# execute the sort function

sorted\_words = sort\_words(words)

Fifth step is to print the words that have been sorted to the console.

# Print the sorted words

print(“Sorted words:”)

for word in sorted\_words:

print(word)

## 3. Write a program, which will find all the numbers between 1000 and 3000 (both included) such that each digit of a number is an even number. The numbers obtained should be printed in a comma separated sequence on a single line. Hint: In case of input data being supplied to the question, it should be assumed to be a console input. Divide each digit with 2 and verify is it even or not.

The first step to write a function to determine if all digits are even and return a Boolean.

# return true when even

def is\_all\_digits\_even(num):

while num > 0:

digit = num % 10

if digit % 2 != 0:

return False

num //= 10

return True

The second step is to define an array that would be used to store even numbers.

# store numbers that are even

even\_digit\_numbers = []

The third step to make a for loop that takes all the numbers between 1000 and 3001 and loops through them to determine if they are even and then store them if they are.

# loop through 1000-3000

for num in range(1000, 3001):

if is\_all\_digits\_even(num):

even\_digit\_numbers.append(str(num))

The fourth and final step is to print to the console an explanation of what the program was doing, and show the results.

# print results of the program to the console.

print(“This program finds the even numbers between 1000 and 3000:”)

print(“, “.join(even\_digit\_numbers))

## 4. Write a program that accepts a sentence and calculate the number of letters and digits. Suppose if the entered string is: Python0325 Then the output will be: LETTERS: 6 DIGITS:4

The first step would be to accept input from console and put it in a variable.

# input from the user for a string like Python0325

input\_string = input(“Enter a sentence: “)

The second step would be to create a function that accepts an input sting and returns letters and a count for letters and a count for numbers.

# print results of the program to the console.

def count\_letters\_and\_digits(input\_string):

letters = 0

digits = 0

for char in input\_string:

if char.isalpha():

letters += 1

elif char.isdigit():

digits += 1

return letters, digits

The third step would be to execute the function and save the output into variables that are outside the scope of the function.

# input from the user for a string like Python0325

letters, digits = count\_letters\_and\_digits(input\_string)

The fourth step would be to execute the function and save the output into variables that are outside the scope of the function.

# input from the user for a string like Python0325

print(f”LETTERS: {letters}”)

print(f”DIGITS: {digits}”)

## 5. Design a code which will find the given number is Palindrome number or not. Hint: Use built-in functions of string

The first step would be to accept input from console and put it in an integer variable.

# input from the user for a number

number = int(input(“Enter a number: “))

The second step would be to write a function that determines if a number is a palindrome:

def is\_palindrome(number):

# Convert the number to a string

number\_str = str(number)

# Check if the string is equal to its reverse

return number\_str == number\_str[::-1]

The third step is to execute a print out on the output depending on whether it is a palindrome:

if is\_palindrome(number):

print(f”{number} is a palindrome.”)

else:

print(f”{number} is not a palindrome.”)