

Computational Maths

**CT4032**

**Workbook – Week 3**

2023 – 24

**Dr Nasreen Anjum**

School of Computing and Engineering

**University of Gloucestershire 2023/24**

All rights reserved. No part of this publication may be reproduced, stored or transmitted in any form or by any means, including – but not limited to – photocopy, recording, or any information storage and retrieval system, without the specific prior written permission of University of Gloucestershire.

**Syntax and Explanation of for Loop**

Syntax: for loop

for variable in iterable:

# Code to be executed in each iteration

#

**Explanation:**

* for keyword is used to initiate a for loop.
* variable is a temporary variable that takes on the value of each item in the iterable during each iteration of the loop.
* iterable is the collection of items that you want to loop through, such as a list, tuple, string, or a range.
* The code inside the loop is executed in each iteration for each item in the iterable.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# **Practical Activity-1**

**Task: Write a python program to display numbers from 1 to 100 using for loop.**

for number in range(1, 101):

print(number)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# **Practical Activity-2**

**Task: Write a python program to display even numbers from 2 to 100 using for loop.**

for number in range(2, 101, 2):

print(number)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# **Practical Activity-3**

**Task: Write a python program to display numbers from 1 to 100 using while loop.**

number = 1

while number <= 100:

print(number)

number += 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# **Coding Challenge-1**

**Activity: Write a python program that generate the sequence 3,5,7,……..100**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

# **Coding Challenge-2**

**Activity: Write a python program that generate the sequence 1,4,9,16,……..102.**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

# **Coding Challenge-3**

**Activity: Write a python program that generate the sequence 80, 75, 70, 65, 60**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

# **Practical Activity-4**

**Task: Following program calculate the sum of an arithmetic series. Modify the code by changing the values of first\_term, common\_difference, and num\_terms and observe how the sum of the arithmetic series changes.**

def arithmetic\_series\_sum(a, d, n):

# a = first term, d = common difference, n = number of terms

# Calculate the sum of the arithmetic series

series\_sum = (n \* (2 \* a + (n - 1) \* d)) / 2

return series\_sum

# Example usage:

first\_term = 1

common\_difference = 2

num\_terms = 5

result = arithmetic\_series\_sum(first\_term, common\_difference, num\_terms)

print("Sum of the arithmetic series:", result)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# **Practical Activity-5**

**Task: Following program calculate the sum of a geometric series. Modify the code by changing the values of first\_term, common\_ratio, and num\_terms and observe how the sum of the geometric series changes.**

def geometric\_series\_sum(a, r, n):

# a = first term, r = common ratio, n = number of terms

# Calculate the sum of the geometric series

series\_sum = a \* (1 - r\*\*n) / (1 - r)

return series\_sum

# Example usage:

first\_term = 2

common\_ratio = 3

num\_terms = 4

result = geometric\_series\_sum(first\_term, common\_ratio, num\_terms)

print("Sum of the geometric series:", result)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# **Practical Activity-6**

**Task: Following program print the square numbers sequence. Do the following**

* **Modify the code by changing the value of num\_terms to generate different lengths of the sequence.**
* **What is the pattern in the square numbers sequence?**
* **Are the square numbers always increasing?**
* **How does the sequence change as num\_terms increases?**
* **Can you find the nth square number without generating the entire sequence?**
* **experiment with different values of num\_terms and observe how the plot changes.**
* **Explore real-world examples where square numbers are used, such as in geometry, area calculations, or Pythagoras' theorem.**

def geometric\_series\_sum(a, r, n):

# a = first term, r = common ratio, n = number of terms

# Calculate the sum of the geometric series

series\_sum = a \* (1 - r\*\*n) / (1 - r)

return series\_sum

# Example usage:

first\_term = 2

common\_ratio = 3

num\_terms = 4

result = geometric\_series\_sum(first\_term, common\_ratio, num\_terms)

print("Sum of the geometric series:", result)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# **Coding Challenge-4**

**Activity: Printing a Geometric Sequence in Python**

**Objective:** write a Python program that generates and prints a geometric sequence up to a given number.

**Instructions:**

1. Write a Python program that prints a geometric sequence.
2. The sequence should start with the number 1 and double each time to create the next term.
3. The program should continue generating terms until the next term exceeds a given maximum number specified by the user.
4. The program should take user input for the maximum number.
5. The program should print each term of the sequence.

**Example:**

If the user specifies a maximum number of 100, the program should generate and print the sequence as follows:

1, 2, 4, 8, 16, 32, 64

**Hints:**

* You can use a while loop to keep generating terms until the next term exceeds the maximum number.
* Start with an initial value of 1, and inside the loop, double the value in each iteration.
* Check whether the current term exceeds the maximum number in each iteration, and if it does, exit the loop.

**Challenge Extension:**

* print the sum of all terms in the generated sequence.
* Calculate and print the number of terms in the sequence that are less than a given threshold (e.g., less than 50).
* Modify the program to generate a geometric sequence with a different common ratio (e.g., 3, 6, 12, 24, ...).

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# **Coding Challenge-5**

**Activity: Perfect Square Checker**

**Objective:** Write a Python program that generates and prints a sequence by multiplying the previous term by 2, up to a given maximum number.

**Instructions:**

1. Start the sequence with the number 5.
2. Continue generating terms until the next term exceeds a maximum number that you'll specify.
3. You should take user input for the maximum number.
4. Finally, your program should print each term of the sequence.

Example:

If you decide the maximum number is 50, your program should generate and print the sequence like this:

5, 10, 20, 40

**Helpful Hints:**

Utilize a while loop to keep generating terms until the next term exceeds the maximum number.

Start with an initial value of 5, and inside the loop, double the value with each iteration.

Check in each iteration whether the current term exceeds the maximum number, and if it does, exit the loop.

**Challenge Extensions (Optional):**

For those of you looking for an extra challenge:

* Calculate and print the sum of all terms in the generated sequence.
* Determine and print the number of terms in the sequence that are less than a given threshold (e.g., less than 30).
* Try to modify your program to generate a sequence with a different initial value and common ratio (e.g., starting with 2 and multiplying by 3, 6, 12, 24, ...).

This coding challenge is an excellent opportunity for you to practice using loops and conditional statements in Python to generate and print a sequence. Each term in the sequence should be obtained by doubling the previous term, and you'll be able to control when the sequence stops by specifying a maximum number.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# **Practical Activity-7**

**Task: Execute the following program that generate and print the Fibonacci sequence up to a given number of terms. Do following**

* modify the code by changing the value of num\_terms to generate different lengths of the sequence.
* What is the pattern in the Fibonacci sequence?
* How does each term relate to the previous two terms?
* Can they find a formula to calculate the nth Fibonacci number without generating the entire sequence?

def generate\_fibonacci\_sequence(n):

fibonacci\_sequence = [0, 1]

while len(fibonacci\_sequence) < n:

next\_term = fibonacci\_sequence[-1] + fibonacci\_sequence[-2]

fibonacci\_sequence.append(next\_term)

return fibonacci\_sequence

# Example usage:

num\_terms = 10

fibonacci\_sequence = generate\_fibonacci\_sequence(num\_terms)

print("Fibonacci Sequence:", fibonacci\_sequence)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# **Practical Activity-8**

**Plot the Fibonacci sequence using a Python plotting library like Matplotlib. Experiment with different values of num\_terms and observe how the plot changes.**

import matplotlib.pyplot as plt

def plot\_fibonacci\_sequence(n):

fibonacci\_sequence = generate\_fibonacci\_sequence(n)

plt.plot(range(1, n+1), fibonacci\_sequence, marker='o')

plt.xlabel("Term Number")

plt.ylabel("Fibonacci Number")

plt.title("Fibonacci Sequence")

plt.grid(True)

plt.show()

# Example usage:

num\_terms = 10

plot\_fibonacci\_sequence(num\_terms)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Congratulations on successfully completing all of the coding challenges!**

*Your dedication and effort in solving these challenges demonstrate your commitment to learning and improving your programming skills. It's a significant achievement to tackle a variety of coding problems and develop practical solutions using Python.*

*By working through these activities, you've gained valuable experience in areas like problem-solving, algorithmic thinking, user input handling, and conditional statements. These skills are essential for any programmer and will serve as a strong foundation for your future coding endeavors.*

*Keep up the fantastic work and continue to challenge yourself with more coding tasks. Each challenge you conquer brings you one step closer to becoming a proficient coder. Remember that learning is a continuous journey, and the skills you've acquired will undoubtedly benefit you in your academic and professional pursuits.*

*Stay motivated, keep coding, and reach for even greater heights in your programming journey!*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*