## VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**JNANA SANGAMA, BELAGAVI-590010**



PROJECT REPORT

ON

## “GRAPH COLORING”

Submitted by:

1. SARITHA B N 1VE22CS139

2. TAI SANJANA 1VE22CS167

3. VARSHA V 1VE22CS177

4. YASHASWINI B 1VE22CS183

5. MEGHANA D E 1VE23CS406

UNDER THE GUIDENCE

OF

**PROF.MRS.M.ARCHANA**

**ASSISTANT PROFESSOR**



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**SRI VENKATESHWARA COLLEGE OF ENGINEERING**

**VIDYANAGAR,BANGALORE -562157**

**2023-24**

# 



**CERTIFICATE**

This is to Certify that the Activities work entitled **“GRAPH COLORING”** carried out by **SARITHA B N USN 1VE22CS139 ,TAI SANJANA USN 1VE22CS167,VARSHA V USN 1VE22CS177 ,YASHASHWINI B USN 1VE22CS183,MEGHANA D USN 1VE23CS406** who is a Bonafide student of III Semester, Computer Science and Engineering, Sri Venkateshwara College of Engineering, in partial fulfillment for the award of Bachelor of Engineering in the Visvesvaraya Technological University, Belagavi during the academic year 2023-2024. The Activity report has been approved as it satisfies the academic requirements in the respect of Data Structures & Algorithm (BCS304) work prescribed for the said degree.

**…………………… ..…………………. …………………….**

**Signature of the Guide Signature of the HOD Signature of the Principal**

**Mrs. Archana M Dr Hema M S Dr. Nageswara Guptha M**

Assistant Professor, Professor and HOD, Principal,

SVCE Bangalore SVCE Bangalore SVCE Bangalore

**ACKNOWLEDGEMENT**

I express my sincere gratitude to everyone who contributed to the successful completion of this mini project on the graph coloring in python . Firstly, I extend my heartfelt thanks to my project guide MRS.M.ARCHANA , whose guidance and insights were invaluable throughout the development process. Their expertise in python and constructive feedback significantly shaped the project's trajectory.

I would like to acknowledge the support and encouragement from my peers and friends who provided assistance during brainstorming sessions and code reviews. Their collaborative spirit fostered an environment conducive to learning and innovation.

Special thanks are extended to the academic institution for providing the necessary resources, infrastructure, and a conducive environment for project work. The project could not have materialized without the support of “SRI VENKATESHWARA COLLEGE OF ENGINEERING”.

I appreciate the insights gained from the literature survey, which laid the foundation for the project. The wealth of knowledge from existing studies informed critical decisions during the development phase. Lastly, I would like to express my gratitude to my family for their unwavering support and understanding during the project's duration.

This project stands as a testament to the collective effort, dedication, and collaborative spirit of all those involved, and I am sincerely thankful for the contributions that have brought this graph coloring in python to fruition.

# ABSTRACT

1. This project focuses on addressing the graph coloring problem, a fundamental challenge in graph theory with widespread applications.
2. The objective is to implement and optimize a graph coloring algorithm, exploring various methodologies and data representations.
3. The project involves an in-depth examination of existing algorithms, their limitations, and potential enhancements.
4. Practical implementation details, including the choice of programming languages and tools, are presented.
5. Testing and evaluation criteria are established, leading to a thorough analysis of the algorithm's performance. Results are visually represented, and insights gained from the project are discussed.
6. The abstract concludes with a summary of key findings, implications, and potential avenues for future research, offering a comprehensive

|  |  |  |  |
| --- | --- | --- | --- |
| Chapter No. | Chapter Title | | Page No. |
| 1. | INTRODUCTION | | 1 |
| 2. | LITERATURE SURVEY | | 2 |
| 3. | OBJECTIVE | | 3 |
| 4. | METHODOLOGY | | 4-5 |
| 5. | PROGRAM CODE | | 6 |
| 6. | SNAPSHOT | | 7-8 |
| 7. | SUMMARY | | 9 |
| BIBLOGRAPHY | | 10 | |

**TABLE OF CONTENTS**

ACKNOWLEDGEMENT.…….…………………………………………………………………i

ABSTRACT ………………………………………………………………………………...……ii

**CHAPTER 1**

**INTRODUCTION**

## Graph coloring, a classical problem in graph theory, serves as the focal point for this mini-project, aiming to explore, implement, and optimize algorithms to address the intricate challenges associated with assigning colors to vertices in a graph. Graphs, fundamental mathematical structures composed of vertices and edges, find ubiquitous application in modeling relationships, dependencies, and networks across various disciplines. The process of graph coloring involves assigning colors to vertices in a way that no two adjacent vertices share the same color. This seemingly simple task carries profound implications and has practical relevance in diverse fields such as scheduling, resource allocation, register allocation in compilers, and even map coloring.

## The significance of graph coloring lies in its ability to model and solve complex real-world problems. For instance, in scheduling applications, graph coloring can represent tasks that need to be executed without any conflicts, ensuring optimal resource utilization and efficient task completion. In the context of register allocation in compilers, graph coloring is employed to assign registers to variables, minimizing the number of registers needed for a program, thus enhancing computational efficiency. Furthermore, in map coloring problems, graph coloring can be applied to represent countries or regions such that no two adjacent entities share the same color, reflecting geopolitical considerations.

## This mini-project seeks to delve into the depths of graph coloring, examining established algorithms, and proposing optimizations to contribute to the ongoing research in this domain. By understanding and implementing efficient solutions to graph coloring problems, we aim to address challenges associated with algorithmic complexity and real-world applicability. The subsequent sections of this project will unfold with a comprehensive review of existing graph coloring algorithms, an exploration of appropriate methodologies, the selection of data structures for efficient representation, the practical implementation of the chosen algorithm, and rigorous testing and evaluation.

# CHAPTER 2

# LITERATURE SURVEY

# The literature surrounding graph coloring is rich and diverse, reflecting the problem's fundamental nature and its applications across various domains. Numerous algorithms have been proposed and studied, each contributing to our understanding of efficient graph coloring techniques. This literature review provides an overview of key contributions in the field, highlighting the strengths and limitations of existing approaches.

# The Greedy Coloring algorithm, a foundational method, is often the first choice due to its simplicity. It iteratively colors vertices, choosing the smallest available color at each step. While efficient, Greedy Coloring may not always yield an optimal solution, especially for complex graphs.

# Backtracking algorithms, such as the Welsh–Powell and DSATUR algorithms, offer more sophisticated approaches. They explore different coloring possibilities and backtrack when conflicts arise. DSATUR, in particular, considers the saturation degree of vertices, prioritizing those with higher degrees during the coloring process. These algorithms tend to produce better results but may suffer from increased computational complexity.

# Genetic Algorithms (GAs) represent a different paradigm, inspired by evolutionary processes. GAs evolve a population of potential solutions, applying genetic operators like crossover and mutation to improve coloring fitness. While promising, GAs can be computationally intensive and may require fine-tuning for optimal performance.

# Recent research has also explored parallel and distributed graph coloring algorithms to address the growing complexity of large-scale graphs. The emphasis is on reducing computation time by leveraging parallel processing capabilities.

# Despite the wealth of existing literature, there remains room for improvement in terms of algorithmic efficiency, adaptability to different graph types, and applicability to real-world scenarios. This mini-project aims to contribute to this body of knowledge by exploring existing algorithms, identifying their nuances, and proposing potential optimizations for enhanced graph coloring solutions

# CHAPTER 3

# OBJECTIVE

The objective of this mini project is to delve into the graph coloring problem, a fundamental challenge in graph theory, and contribute to the field's knowledge and practical applications. The specific aims include a comprehensive review of existing graph coloring algorithms, such as Greedy Coloring, Backtracking, and Genetic Algorithms, to understand their strengths and limitations. The project seeks to identify an algorithmic approach that not only addresses theoretical complexities but also proves practical in real-world scenarios.

Efficient data representation will be explored, emphasizing the selection of appropriate structures for optimal computational performance. Through meticulous implementation and testing, the objective is to evaluate the chosen algorithm's effectiveness, comparing it with established solutions and identifying areas for improvement.

Furthermore, the project aims to contribute new insights and potential optimizations to enhance the efficiency of graph coloring algorithms. The overarching goal is to bridge the gap between theoretical foundations and practical utility, providing valuable knowledge for both researchers and practitioners in the broader context of graph theory.

By achieving these objectives, the mini project aspires to advance our understanding of graph coloring algorithms and offer practical solutions applicable to a diverse range of scenarios, from scheduling problems to resource allocation and beyond.

### CHAPTER 4

### METHODOLOGY

The methodology for the graph coloring mini project involves a systematic approach to explore, implement, and optimize algorithms for efficient graph coloring. The key steps include:

**1. Literature Review:**

- Conduct an in-depth review of existing graph coloring algorithms, including Greedy Coloring, Backtracking algorithms, and Genetic Algorithms.

- Analyze their strengths, weaknesses, and applications to gain insights into current methodologies.

**2. Algorithm Selection:**

- Based on the literature review, select a suitable algorithmic approach that aligns with the project's objectives.

- Consider algorithmic complexity, practical implementation aspects, and real-world applicability.

**3. Data Representation:**

- Investigate and choose efficient data structures for graph representation, balancing considerations of space and time complexity.

- Ensure that the selected structures enhance the algorithm's computational efficiency.

**4. Algorithm Implementation:**

- Implement the chosen algorithm, focusing on practical aspects and considering potential modifications or optimizations identified during the literature review.

**5. Testing Criteria:**

- Develop comprehensive testing criteria to evaluate the algorithm's performance.

- Include test cases with varying graph sizes, structures, and complexities to ensure robust evaluation.

**6. Comparative Analysis:**

- Compare the implemented algorithm against existing solutions, using established metrics to assess its efficiency.

- Identify strengths, weaknesses, and areas for potential improvement through the analysis.

**7. Results Evaluation:**

- Analyze and interpret the results obtained from testing, drawing insights into the algorithm's performance and behavior across different scenarios.

**8. Optimization Strategies:**

- Explore potential optimizations or modifications to enhance the algorithm's efficiency.

- Consider feedback from the comparative analysis and results evaluation to guide optimization e.

**9. Documentation and Reporting:**

- Document the methodology, implementation details, and results comprehensively.

- Present findings, insights, and optimization strategies in a clear and organized manner for effective communication.

**10. Feedback and Iteration:**

- Seek feedback from peers or mentors and consider potential iterations based on the received feedback.

- Refine the algorithm, methodology, or implementation as needed to enhance the project's overall quality.

By following this methodology, the mini project aims to provide a structured and thorough exploration of graph coloring algorithms, offering valuable insights and potential optimizations.

**CHAPTER 5**

**PROGRAM CODE**

# Take user input for the number of nodes (V) and colors (m)

V = int(input("Enter the number of nodes (V): "))

m = int(input("Enter the number of colors (m): "))

def print\_solution(color):

print("Solution Exists: Following are the assigned colors")

print(" ".join(map(str, color)))

def is\_safe(v, graph, color, c):

for i in range(V):

if graph[v][i] and c == color[i]:

return False

return True

def graph\_coloring\_util(graph, m, color, v):

if v == V:

return True

for c in range(1, m + 1):

if is\_safe(v, graph, color, c):

color[v] = c

if graph\_coloring\_util(graph, m, color, v + 1):

return True

color[v] = 0

return False

def graph\_coloring(graph, m):

color = [0] \* V

if not graph\_coloring\_util(graph, m, color, 0):

print("Solution does not exist")

return False

print\_solution(color)

return True

# Driver code

graph = []

for i in range(V):

row = list(map(int, input(f"Enter the adjacency row for node {i + 1}: ").split()))

graph.append(row)

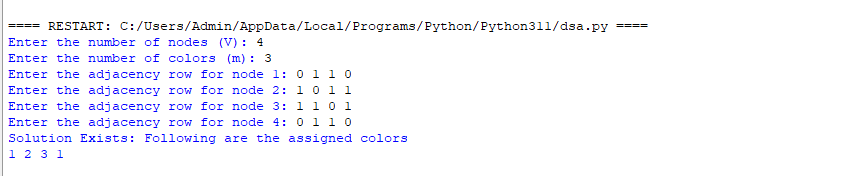
# Function call

graph\_coloring(graph, m)

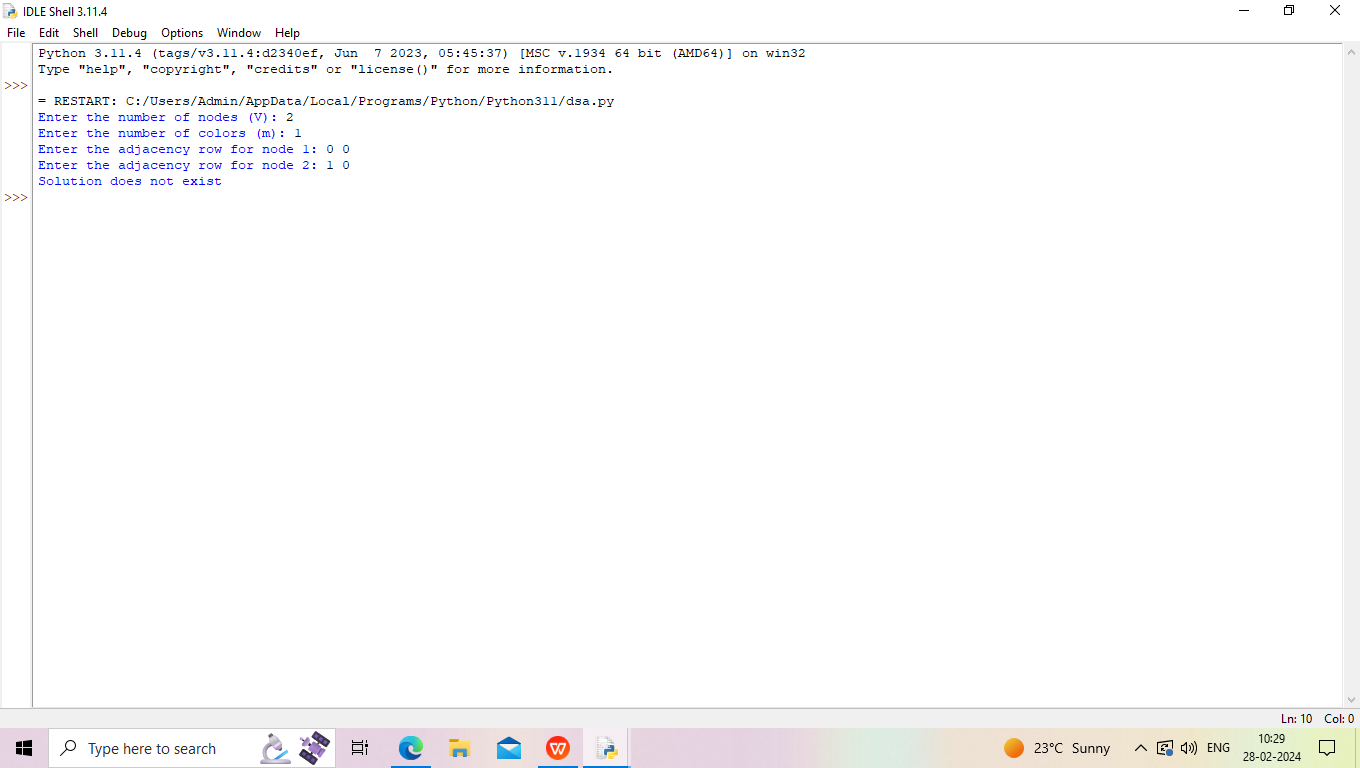
**CHAPTER 6**

**SNAPSHOT**

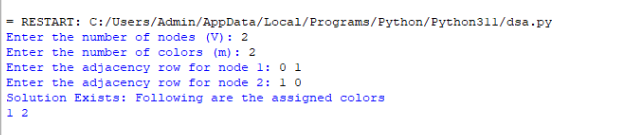
1. For V=4,m=3



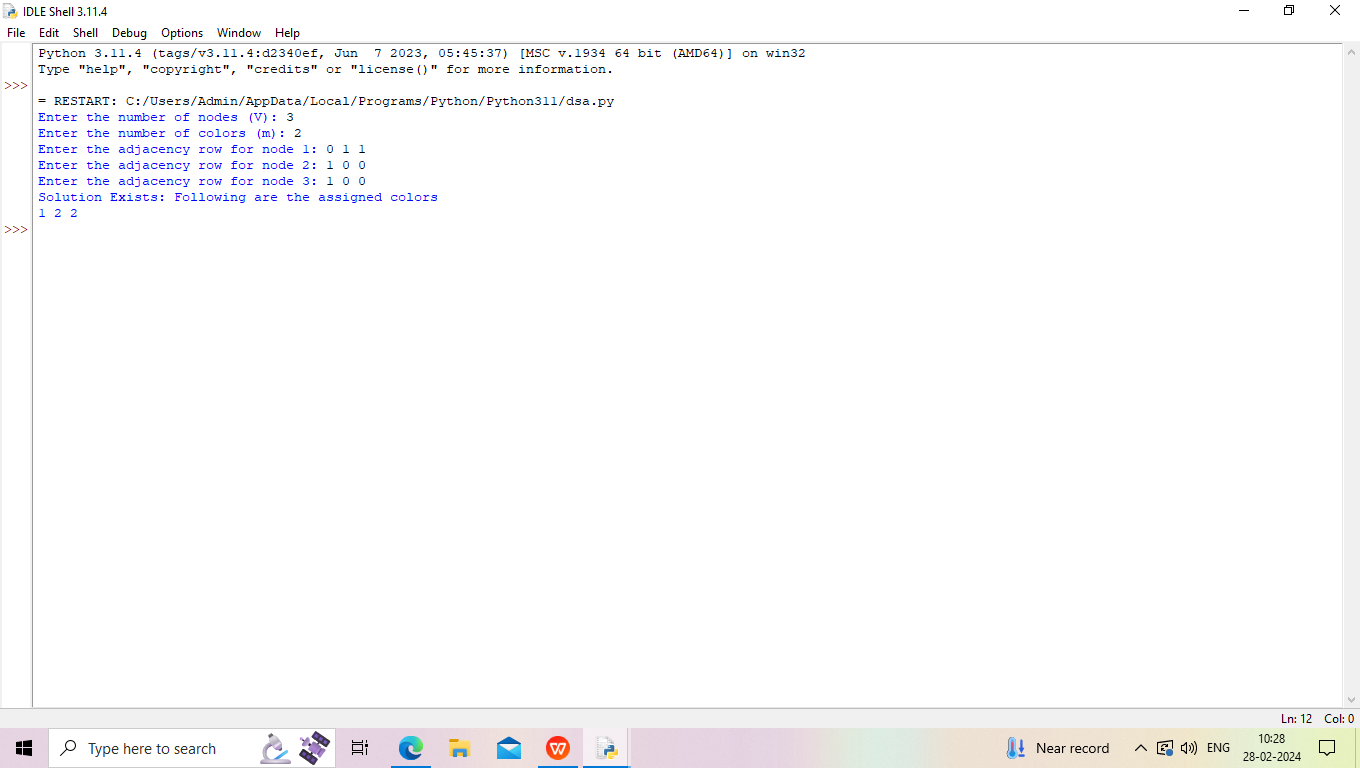
1. For V=2,m=1



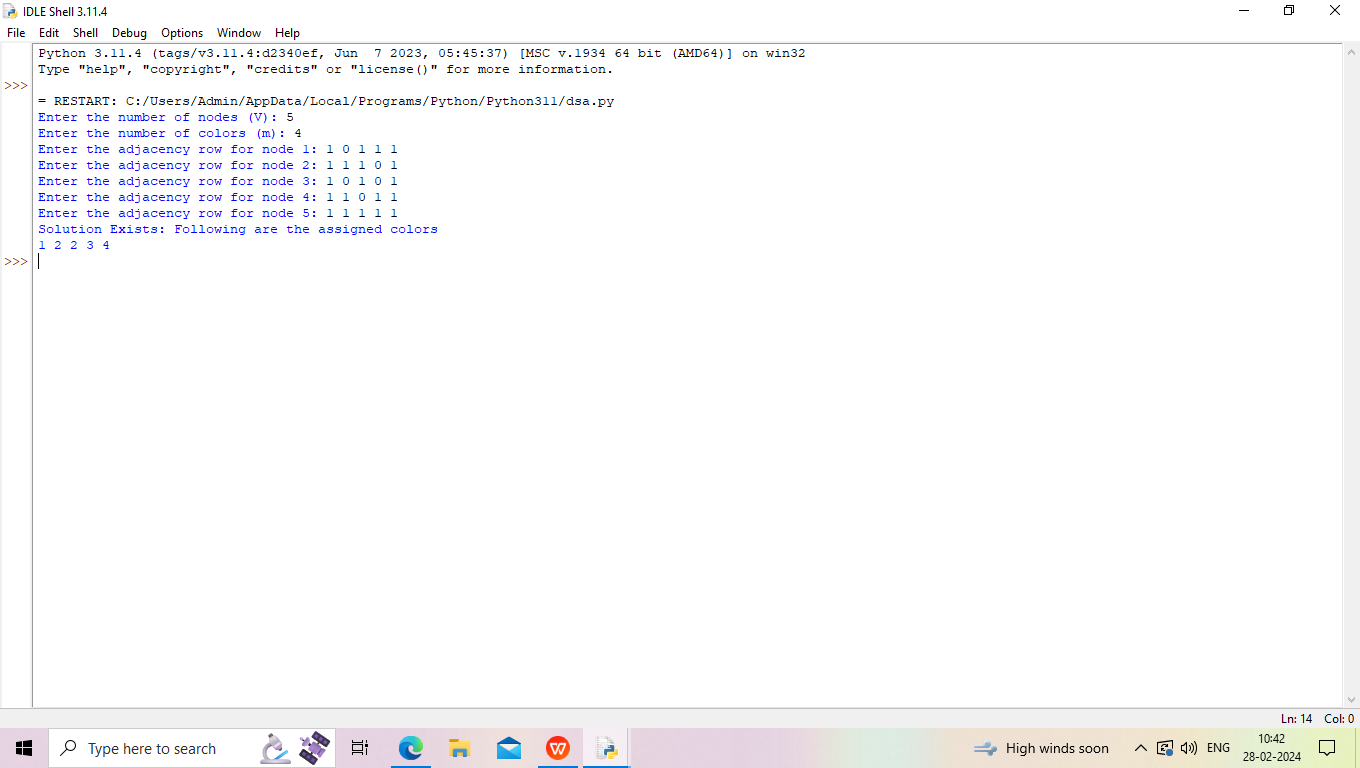
1. For V=2, m=2



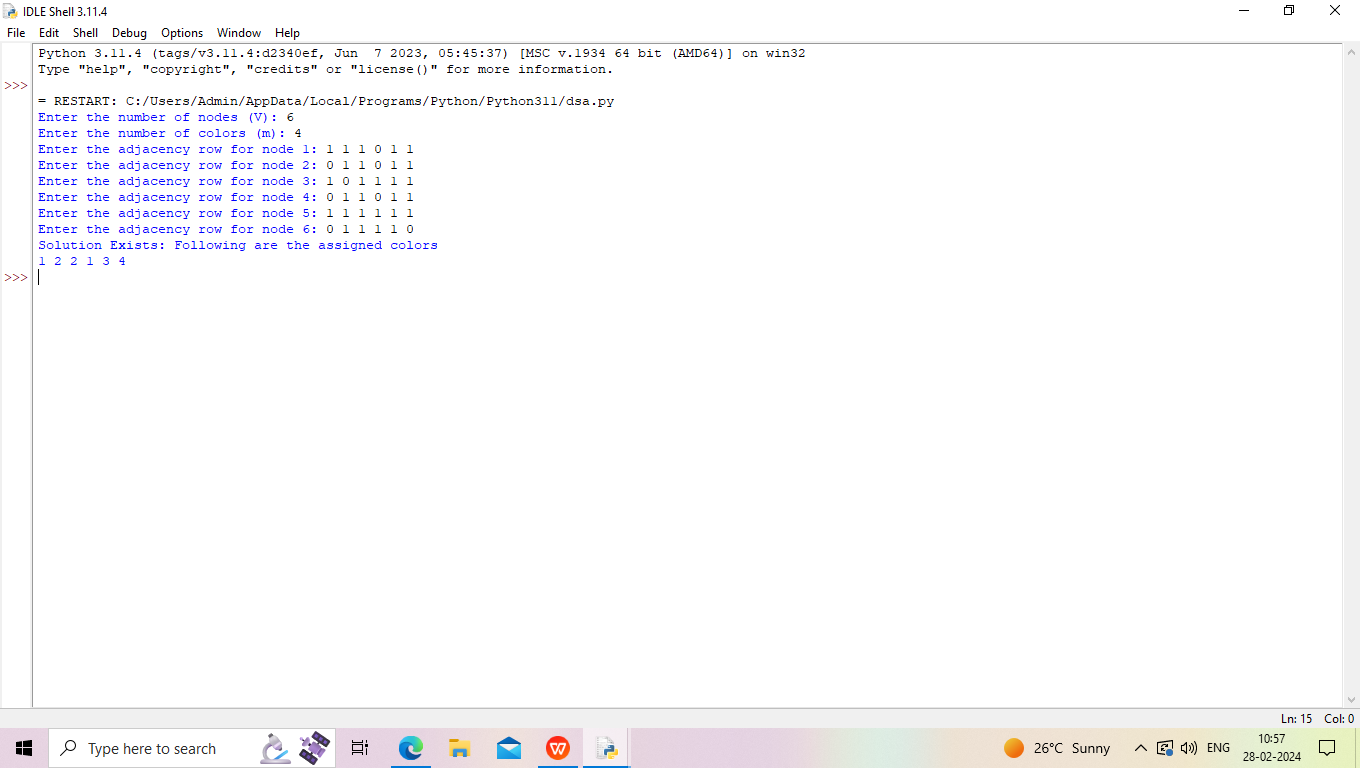
1. For V=3, m=2



1. For V=5,m=4



### 6. For V=6,m=4



**CHAPTER 7**

**SUMMARY**

Graph coloring, a fundamental concept in graph theory, involves assigning colors to vertices while ensuring adjacent vertices have distinct colors. This concept finds practical applications in diverse domains. In map coloring, it delineates geographical regions without color conflicts.

Scheduling benefits from optimized resource allocation using graph coloring. Register allocation in compilers enhances program execution efficiency. Wireless networks employ graph coloring for interference-free frequency assignment. Job shop scheduling in manufacturing optimizes machine usage. Advantages of graph coloring include algorithmic efficiency,visualization of relationships, and versatile applicability. Overall, graph coloring proves to be a powerful and adaptable tool, contributing to efficient problem-solving across various real-world scenarios.

## BIBLOGRAPHY

* A Guide to Graph Coloring (Algorithms and Applications)- R.M.R.Lewis.
* https://www.geeksforgeeks.org/graph-coloring-set-2-greedy-algorithm/amp/
* https://github.com/topics/graph-coloring
* https://www.tutorialspoint.com/Graph-Coloring
* https://youtu.be/052VkKhIaQ4?si=E4LEFpTLwgVyHyaV