

**Artificial Intelligence and Data Science Department CSS/Odd Sem 2024-25/Experiment/MD5, SHA- 1**

|  |  |  |
| --- | --- | --- |
| **Name : Aakash Borji** | **Class/Roll No.:D6ADA/8** | **Grade :** |

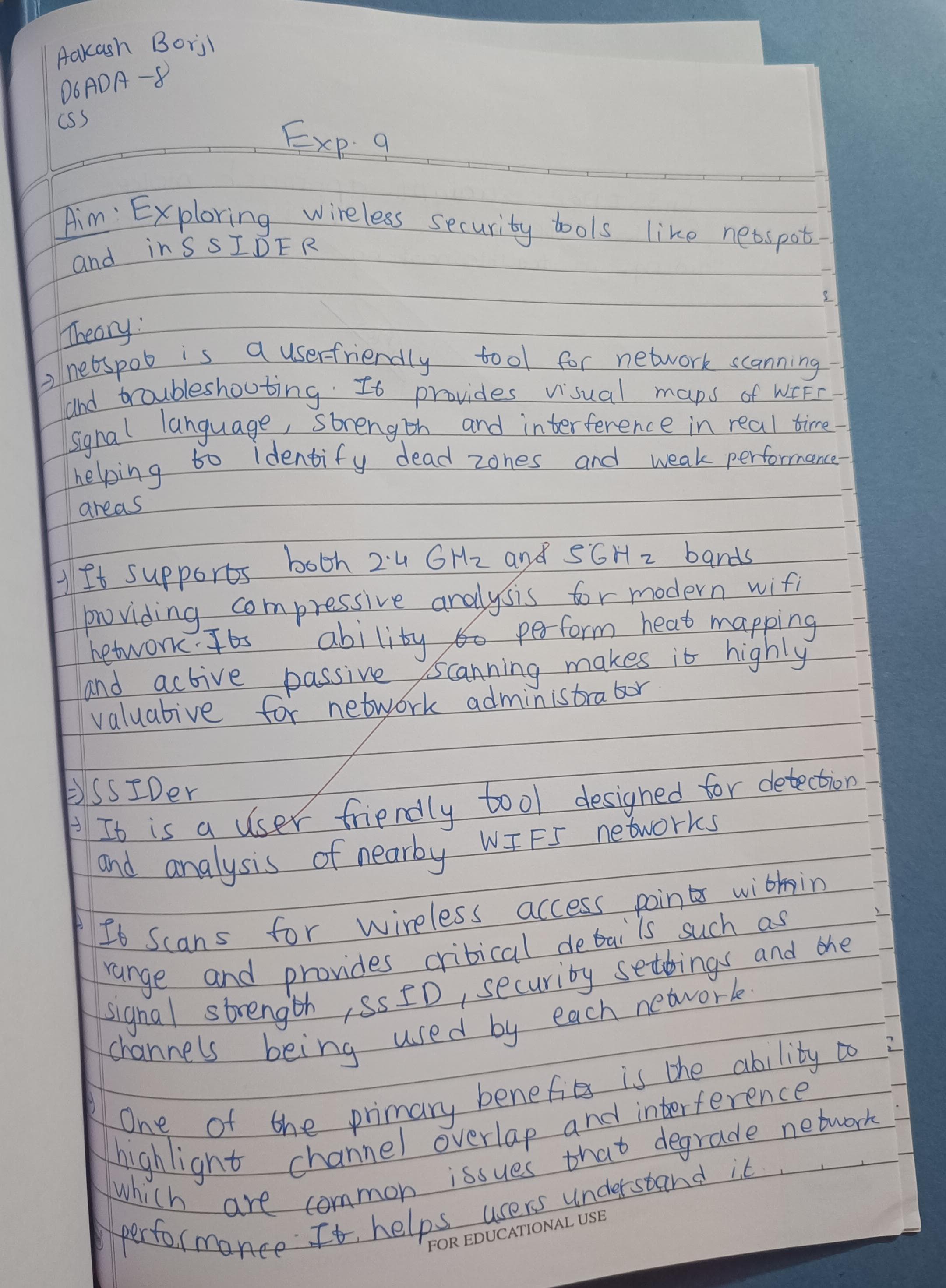
# Title :- For varying message sizes, test integrity of message using MD-5, SHA-1, and analyze the performance of the two protocols. Use crypt APIs

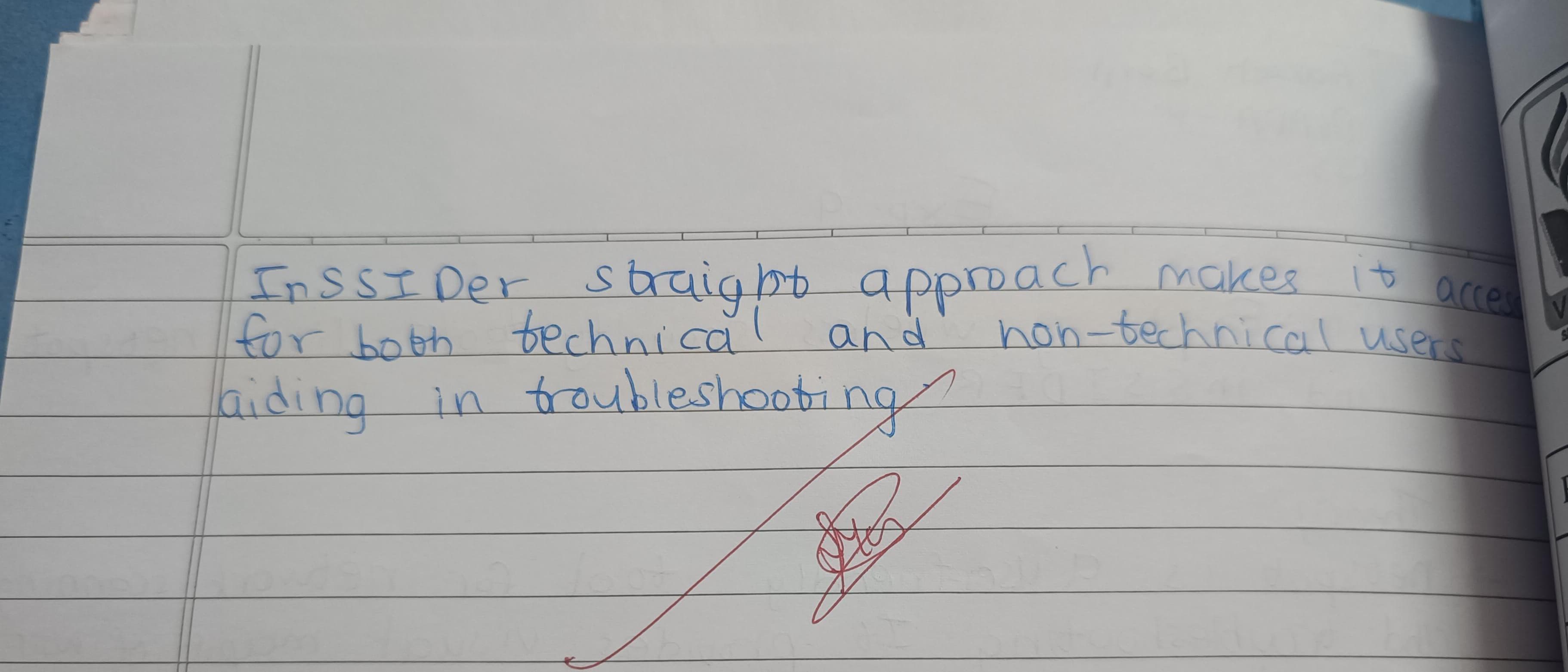
**Objective of Experiment :**

1. **Understand MD5:** Grasp the workings of the MD5 algorithm, including its structure and how it produces a 128-bit hash value.
2. **Implement MD5:** Write code to implement the MD5 hashing algorithm and understand its input-output behavior.
3. **Evaluate Security Weaknesses:** Analyze the vulnerabilities of MD5, including known collision attacks, and understand why it is considered weak for cryptographic purposes.
4. **Understand SHA-1**: Learn how the SHA-1 algorithm functions and its design, resulting in a 160-bit hash value.
5. **Implement SHA-1**: Write code to implement the SHA-1 hashing algorithm, examining its input-output characteristics.
6. **Evaluate Security Weaknesses**: Investigate the vulnerabilities of SHA-1, including known attacks like the SHAttered attack, and understand the reasons for its decline in use.

**Artificial Intelligence and Data Science Department CSS/Odd Sem 2024-25/Experiment/MD5, SHA- 1**

# Description / Theory :







**Artificial Intelligence and Data Science Department CSS/Odd Sem 2024-25/Experiment/MD5, SHA- 1**

# Problem Statement :

1) TIME import time import hashlib

from prettytable import PrettyTable

# Example timing function for hashing

def time\_hash\_operation(hash\_type, data): start\_time = time.time()

if hash\_type == "MD5": hashlib.md5(data.encode()).hexdigest()

elif hash\_type == "SHA": hashlib.sha256(data.encode()).hexdigest()

time.sleep(0.5) # Simulate some processing time return time.time() - start\_time

# Simulate some data data\_1 = "Hi"

data\_2 = "This is a longer paragraph."

data\_3 = "This is an even longer page of text that needs to be hashed."

# Measure elapsed times for different operations elapsed\_time\_1 = time\_hash\_operation("MD5", data\_1) # MD5 elapsed\_time\_4 = time\_hash\_operation("MD5", data\_1) # MD5 elapsed\_time\_2 = time\_hash\_operation("SHA", data\_2) # SHA elapsed\_time\_5 = time\_hash\_operation("SHA", data\_2) # SHA elapsed\_time\_3 = time\_hash\_operation("MD5", data\_3) # MD5 elapsed\_time\_6 = time\_hash\_operation("SHA", data\_3) # SHA

# Create the table

myTable = PrettyTable(["Strings", "MD5 Time (s)", "SHA Time (s)", "Difference (s)"]) myTable.add\_row([data\_1, elapsed\_time\_1, elapsed\_time\_4, abs(elapsed\_time\_1 - elapsed\_time\_4)]) myTable.add\_row([data\_2, elapsed\_time\_2, elapsed\_time\_5, abs(elapsed\_time\_2 - elapsed\_time\_5)]) myTable.add\_row([data\_3, elapsed\_time\_3, elapsed\_time\_6, abs(elapsed\_time\_3 - elapsed\_time\_6)])

# Print the table print(myTable)



**Artificial Intelligence and Data Science Department CSS/Odd Sem 2024-25/Experiment/MD5, SHA- 1**

1. LENGTH

from prettytable import PrettyTable import hashlib

# Function to compute hash lengths def get\_hash\_lengths(text):

md5\_hash = hashlib.md5(text.encode()).hexdigest()

sha256\_hash = hashlib.sha256(text.encode()).hexdigest() return len(md5\_hash), len(sha256\_hash)

# Strings to be used in the table strings = ["Hi", "Paragraph", "Page"]

# Compute hash lengths for each string len\_1, len\_4 = get\_hash\_lengths(strings[0]) len\_2, len\_5 = get\_hash\_lengths(strings[1]) len\_3, len\_6 = get\_hash\_lengths(strings[2])

# Specify the column names while initializing the table

myTable = PrettyTable(["Strings", "MD5 Length", "SHA-256 Length"])

# Add rows

myTable.add\_row([strings[0], len\_1, len\_4]) myTable.add\_row([strings[1], len\_2, len\_5]) myTable.add\_row([strings[2], len\_3, len\_6])

# Display the table print(myTable)



**Artificial Intelligence and Data Science Department CSS/Odd Sem 2024-25/Experiment/MD5, SHA- 1**

1. HASH

import hashlib

str1 = "Hi" str2 = "Ho" str3 = "CSS" str4 = "DSS"

result = hashlib.sha512(str1.encode())

print("The hexadecimal equivalent of SHA512 for 'Hi' is: ") print(result.hexdigest())

result = hashlib.sha512(str2.encode())

print("The hexadecimal equivalent of SHA512 for 'Ho' is: ") print(result.hexdigest())

result = hashlib.md5(str3.encode())

print("The hexadecimal equivalent of MD5 for 'CSS' is: ") print(result.hexdigest())

result = hashlib.md5(str4.encode())

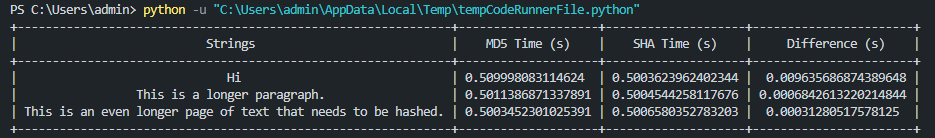
print("The hexadecimal equivalent of MD5 for 'DSS' is: ") print(result.hexdigest())



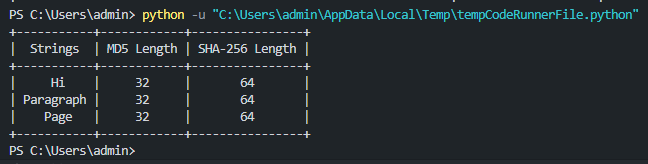
**Artificial Intelligence and Data Science Department**

**CSS/Odd Sem 2024-25/Experiment/MD5, SHA- 1**

**RESULTS:- 1.**



**2.**



**3.**

