UNIT 8
Seminar 4:

Read the Cryptography with Python blog at tutorialspoint.com.

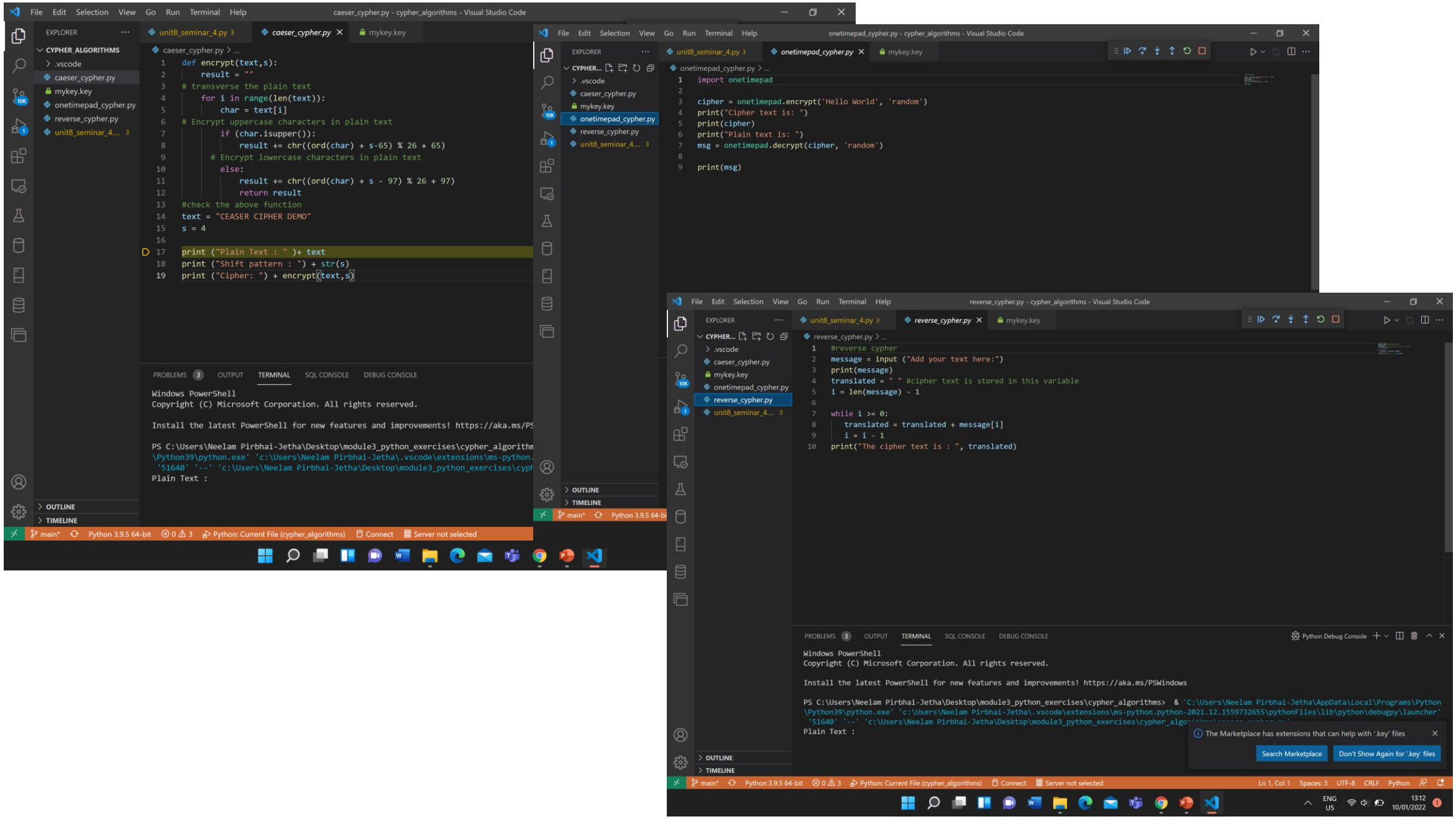
Select one of the methods described/ examples given and create a python program that can take a short piece of text and encrypt it.

Answer the following questions in your e-portfolio:

•Why did you select the algorithm you chose?

•Would it meet the GDPR regulations? Justify your answer.

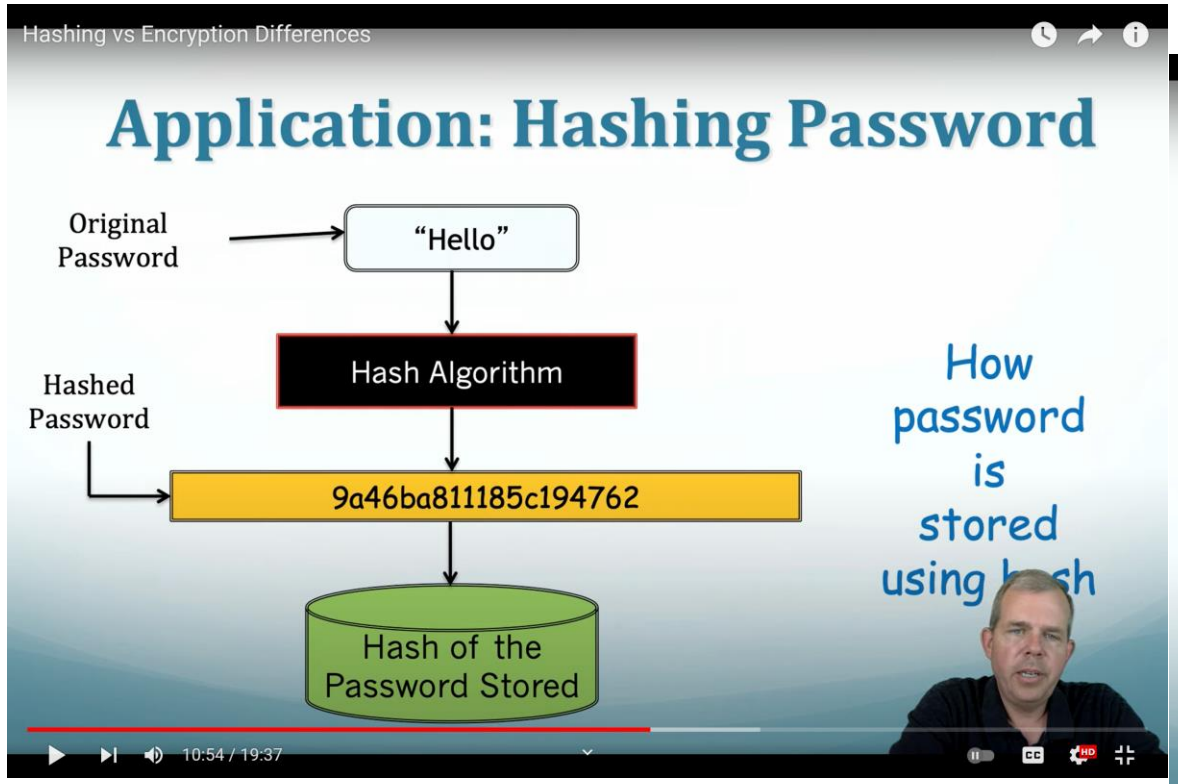
Cypher Algorithms	Drawbacks/Security Vulnerabilities	Advantages
Algorithm of Reverse Cipher	Very weak and easily hacked	
Caeser Cypher	Can be hacked by Brute Force Technique (i.e trying every possible encryption key)	
ROT13 Cypher (Rotate by 13 places)	Easy to hack – by shifting 13 places in reverse	
Transposition Cipher (columnar)		Better than the previous three+re-encrypting the cypher text using the transposition cypher creates better security
Base64 algorithm	Main drawback: stores password in a database	
XOR algorithm		Hard to crack by brute force method
Affine cipher		Includes two functions for encryption and decryption – harder to hack
Monoalphabetic Cipher	Letters change, but frequency doesn't. Therefore, can be cracked by using frequency table	Hard to crack by brute force method
Simple Substitution Cipher		Lots of combinations of letters
Vignere Cipher/polyalphabetic Cipher (One Time Pad Cipher (a type of Vignere Cipher)	Can be cracked if hacker knows the Kasiski method or the Friedman test	Very difficult to hack – considered a secure encryption method Onetime Pad Cypher: almost unbreakable
Data Encryption Standard (Symmetric Cryptography)		Simple and faster; both parties exchange the key in a secure way
RSA Algorithm (Asymmetric Cryptography): has 2 keys – public and private		Most secure way of encryption: Brute force attack won't work – too many keys to try; only numeric so dictionary attack not possible; frequency analysis not possible; no specific mathematical tricks

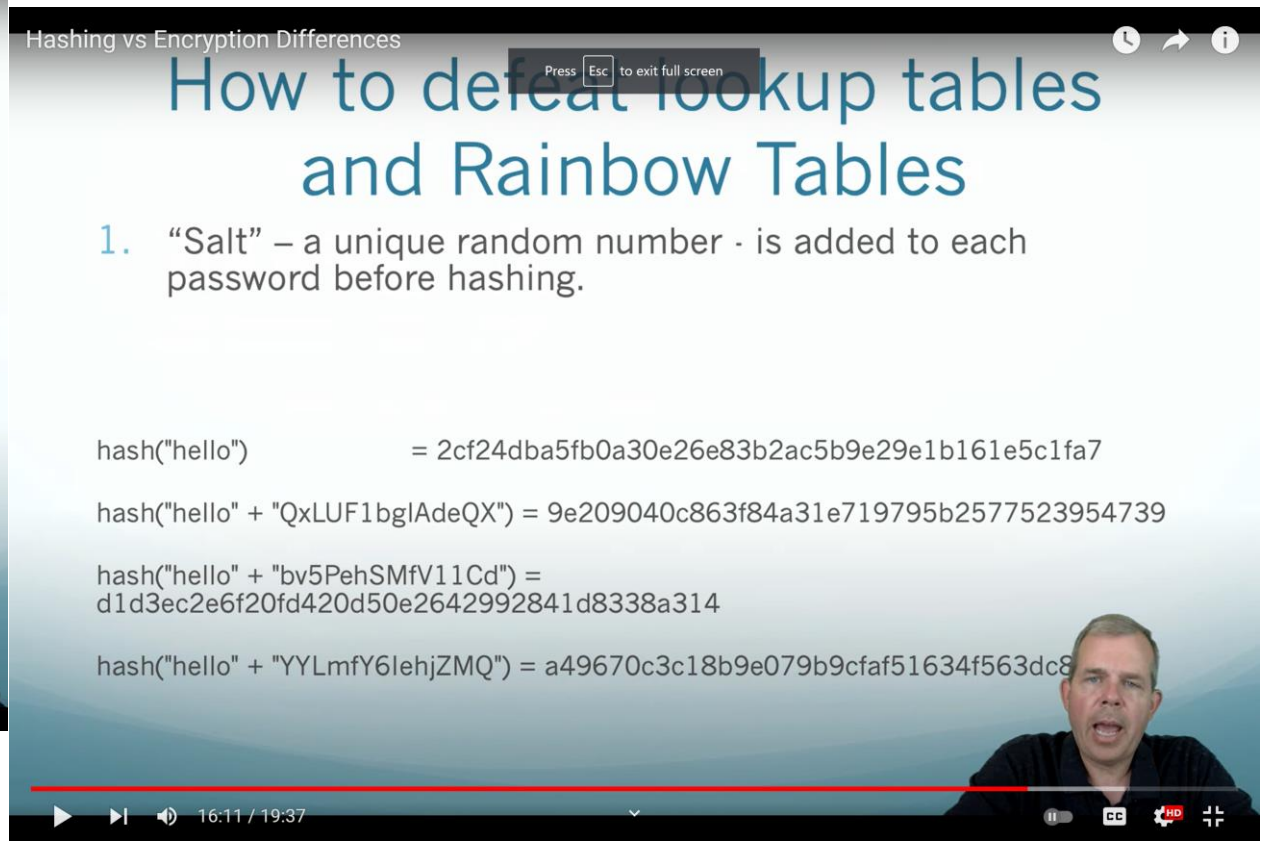

caeser_cypher.py - cypher_algorithms - Visual Studio Code
onetimepad_cypher.py - cypher_algorithms - Visual Studio Code
reverse_cypher.py - cypher_algorithms - Visual Studio Code
def encrypt(text,s):
result = ""
# transverse the plain text
for i in range(len(text)):
char = text[i]
# Encrypt uppercase characters in plain text
if (char.isupper()):
result += chr((ord(char) + s-65) % 26 + 65)
# Encrypt lowercase characters in plain text
else:
result += chr((ord(char) + s - 97) % 26 + 97)
return result
#check the above function
text = "CEASER CIPHER DEMO"
s = 4
print ("Plain Text : ")+ text
print ("Shift pattern : ") + str(s)
print ("Cipher: ") + encrypt(text,s)
import onetimepad
cipher = onetimepad.encrypt('Hello World', 'random')
print("Cipher text is: ")
print(cipher)
print("Plain text is: ")
msg = onetimepad.decrypt(cipher, 'random')
print(msg)
#reverse cypher
message = input ("Add your text here:")
print(message)
translated = " " #cipher text is stored in this variable
i = len(message) - 1
while i >= 0:
translated = translated + message[i]
i = i - 1
print("The cipher text is : ", translated)
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
Plain Text :
The Marketplace has extensions that can help with '.key' files
Search Marketplace
Don't Show Again for '.key' files

Why did you select the algorithm you chose? Would it meet the GDPR regulations? Justify your answer

- Best algorithm : RSA algorithm
- A bit complicated but Might be better to protect from SQL Injection and Broken Authentication (OWASP, “Top 10”)

Additional Notes: Hash Password – Never save plain text in the database+salting





https://www.youtube.com/watch?v=GI790E1JMgw