VILNIUS UNIVERSITY
ŠIAULIAI ACADEMY

BACHELOR PROGRAMME SOFTWARE ENGINEERING

Object Oriented Programming

Practical 8 (Eight).

Student: Sunday Emmanuel Sanni

Lecturer: Prof. Donatas Dervinis

Šiauliai,

19/05/2024

FINAL REPORT ONCREATING BRUTE FORCE ALGORITHHM TO ATTEMPT CRACKING AES ENCRYPTION WITH C#

My report details the functions I have in my program, the work they do and how they are related to one another. How the work of one flows into another and I present print screen reports.

1. **BooleanToVisibilityConverter: IValueConverter:** This function is what made it possible for me to use the check box feature in my DecryptPasswordWindow.xaml.cs and DecryptPasswordWindow.xaml files. The Check box allows a user to specify if they want to use multi-threading feature for running the programme or not.

2. **CreatePasswordViewModel:** This is a simple program that connects the create password page by receiving its data and connects it to the Encryption algorithm, before the data saved to file by the save to file method.

3. **CreatePasswordWindow:** This class is a window that hosts the create password button and break existing password button. It allows the user to either choose to create a new password to be encrypted, saved to file, and broken, or select the option to attempt breaking an already encrypted password that already exists.

4. **DecryptionMethod:** this method detailed the method to decrypt an encrypted password. An IV that is constant is already declared in the method and the method receives the encrypted password and the salt with which it attempts to decrypt the password.

5. **DecryptPasswordWindow:** This is a window that shows when a user wants to decrypt a password. It allows the user select threading and input how many threads they want to use.

6. **DisplayResultWindow:** This is a window shows the result of a completely run process of brute force attack performed. It shows the time taken for the algorithm to find the Salt combination, it shows the salt used and the plain text gotten from decryption.

7. **EncryptionMethod:** This is a method that that receives a plain text from user and combines it with a constant Salt and IV which has been specified. It uses AES encryption method. It returns a string variable which is the encrypted text from the algorithm.

8. **MainWindow:** This is a window that shows up when the program is run. It hosts a button that leads the user to create a password.

9. **MessageDisplayWindow:** This is a window that hosts a static function that is used to display window messages, notifications, or errors encountered anywhere in the programme.

10. **PermutationGenerationMethod:** This is a very important method that is responsible for recursively generating any permutation of alphanumeric texts. It also holds the method for the user to apply multithreading.

11. **PrintToScreenWindow:** This is a window that shows the list of all permutations generated and had to be used to attempts decrypting a password by brute force.

12. **ReadCoreUnits:** This is a method that returns the value for the number of cores the system host has in order to show the capabilities of the computer for multithreading.

13. **ReadEncryptedPasswordMethod:** This is a method that reads from file, the saved encrypted password and makes it available for the decryption algorithm to attempt to use it for decryption.

14. **SaveEncryptedPasswordMethod:** This is a method that receives the encrypted password from the encryption algorithm, and saves it file.

15. **SuccessWindow:** This window is launched when a user successfully creates a password, and allows the user to select whether to begin process of applying brute force or to return and create another password

16. **TimerMethod:** This is a method that returns the value for the number of cores the system host has in order to show the capabilities of the computer for multithreading.

TEST REPORT SHOWN BELOW

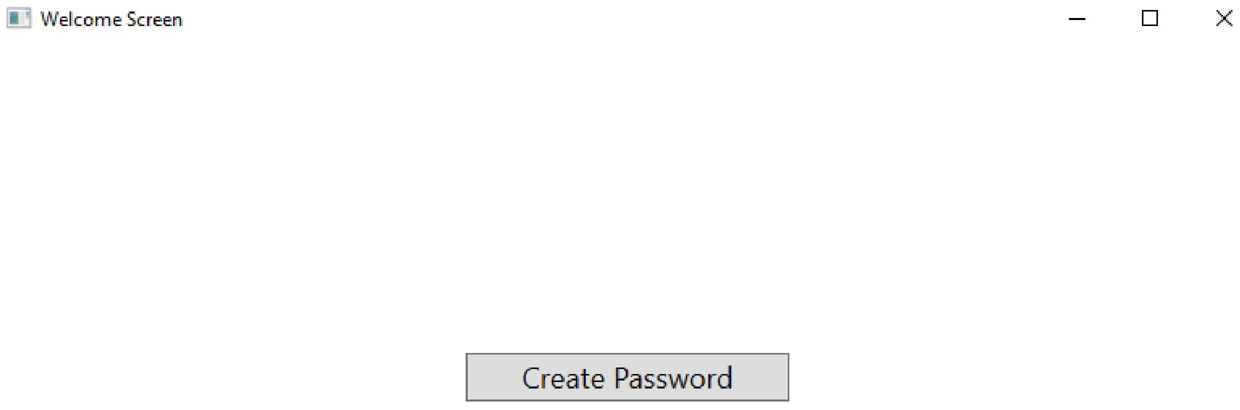
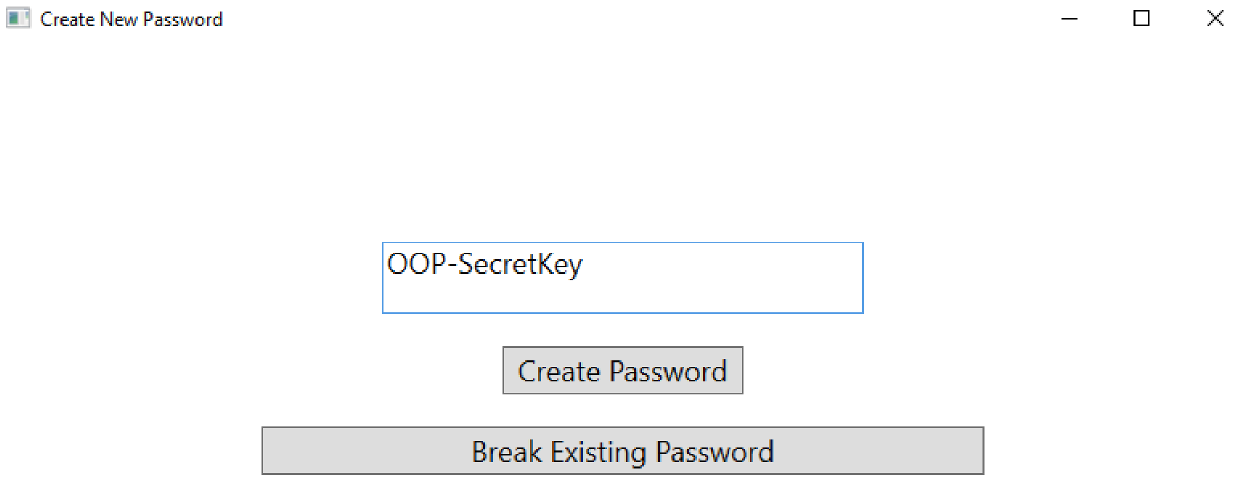
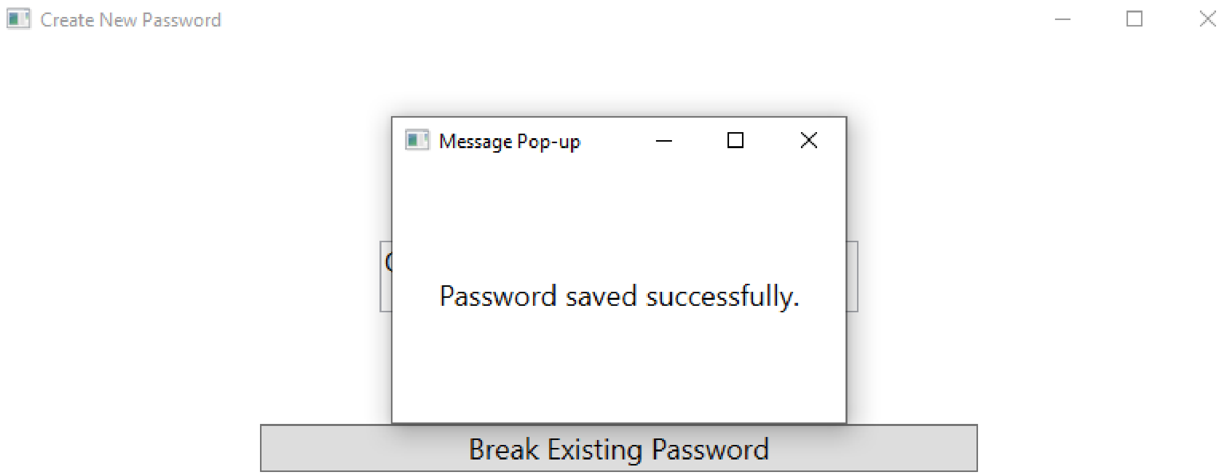


Figure 1: Welcome page



OOP-SecretKey

Figure 2: Next page where user decide to create new password or break existing pasword



Password saved successfully.

Figure 3: User created password inputed in figure 3 and got the message above

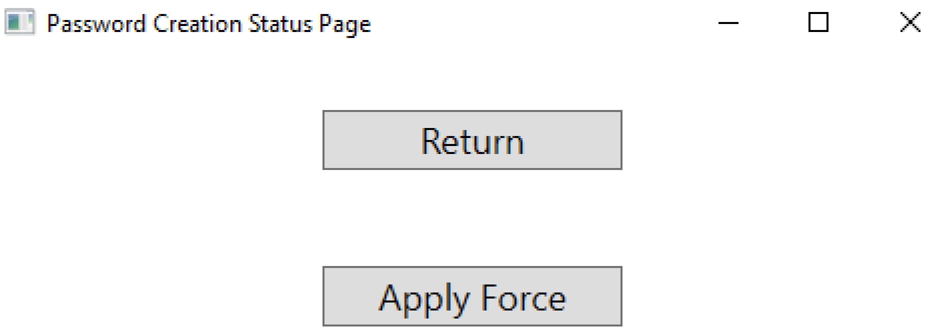


Figure 4: After user cancels window notification

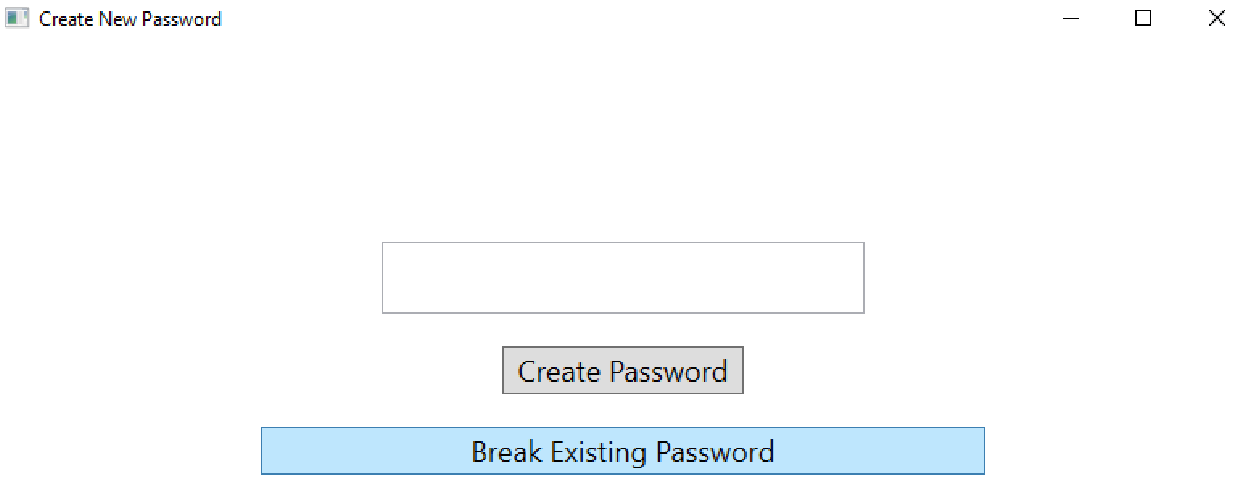
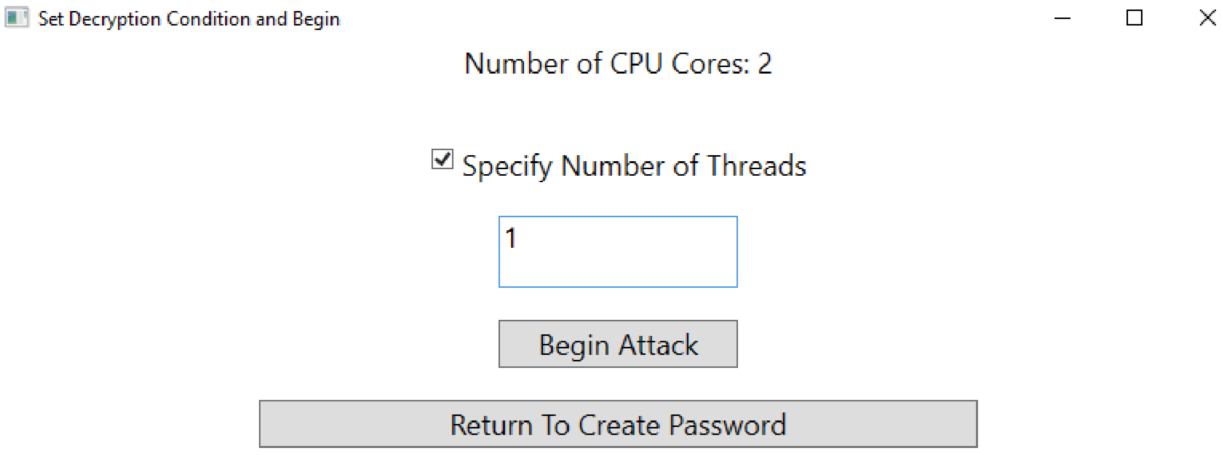


Figure: 5: If user clicked "Break Existing Password" from figure 2.



Number of CPU Cores: 2

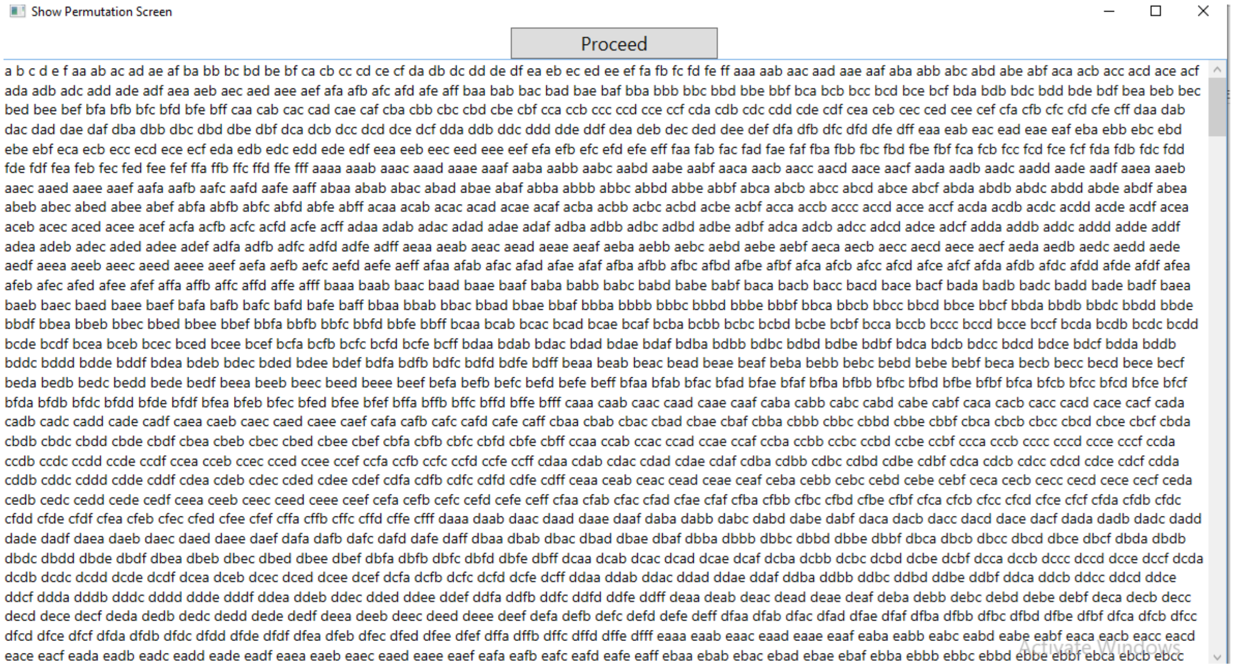
☑ Specify Number of Threads

1

Figure 6: This page shows if user chooses second option from figure 5

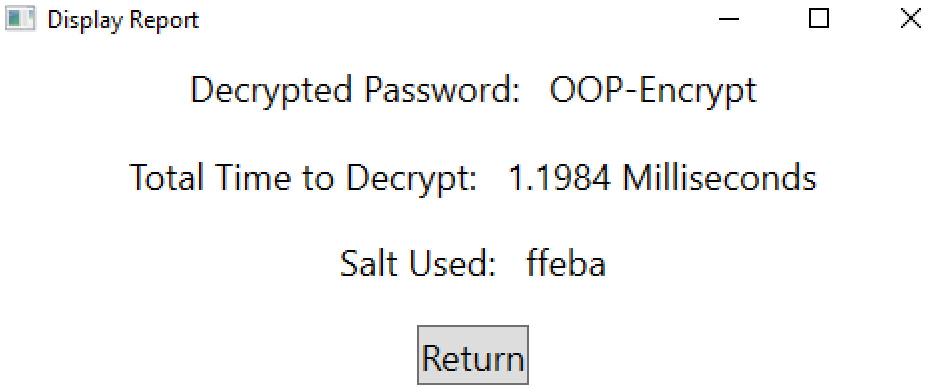
From figure 4, if user clicked return, they are taken to figure 5 and user clicked Apply force, figure 6 above is shown.

In figure 6 above, user can specify the number of threads to user for the brute force attack or use maximum threading option available.



```
a b c d e f aa ab ac ad ae af ba bb bc bd be bf ca cb cc cd ce cf da db dc dd de df ea eb ec ed ee ef fa fb fc fd fe ff aaa aab aac aad aae aaf aba abb abc abd abe abf aca acb acc acd ace acf
ada adb adc add ade adf aea aeb aec aed aee aef afa afb afc afd afe aff baa bab bac bad bae baf bba bbb bbc bbd bbe bbf bca bcb bcc bcd bce bcf bda bdb bdc bdd bde bdf bea beb bec
bed bee bef bfa bfb bfc bfd bfe bff caa cab cac cad cae caf cba cbb cbc cbd cbe cbf cca ccb ccc ccd cce ccf cda cdb cdc cdd cde cdf cea ceb cec ced cee cef cfa cfb cfc cfd cfe cff daa dab
dac dad dae daf dba dbb dbc dbd dbe dbf dca dcb dcc dcd dce dcf dda ddb ddc ddd dde ddf dea deb dec ded dee def dfa dfb dfc dfd dfe dff eaa eab eac ead eae eaf eba ebb ebc ebd
ebe ebf eca ecb ecc ecd ece ecf eda edb edc edd ede edf eea eeb eec eed eee eef efa efb efc efd efe eff faa fab fac fad fae faf fba fbb fbc fbd fbe fbf fca fcb fcc fcd fce fcf fda fdb fdc fdd
fde fdf fea feb fec fed fee fef ffa ffb ffc ffd ffe fff aaaa aaab aaac aaad aaae aaaf aaba aabb aabc aabd aabe aabf aaca aacb aacc aacd aace aacf aada aadb aadc aadd aade aadf aaea aaeb
aaec aaed aaee aaef aafa aafb aafc aafd aafe aaff abaa abab abac abad abae abaf abba abbb abbc abbd abbe abbf abca abcb abcc abcd abce abcf abda abdb abdc abdd abde abdf abea
abeb abec abed abee abef abfa abfb abfc abfd abfe abff acaa acab acac acad acae acaf acba acbb acbc acbd acbe acbf acca accb accc accd acce accf acda acdb acdc acdd acde acdf acea
aceb acec aced acee acef acfa acfb acfc acfd acfe acff adaa adab adac adad adae adaf adba adbb adbc adbd adbe adbf adca adcb adcc adcd adce adcf adda addb addc addd adde addf
adea adeb adec aded adee adef adfa adfb adfc adfd adfe adff aeaa aeab aeac aead aeae aeaf aeba aebb aebc aebd aebe aebf aeca aecb aecc aecd aece aecf aeda aedb aedc aedd aede
aedf aeea aeeb aeec aeed aeee aeef aefa aefb aefc aefd aefe aeff afaa afab afac afad afae afaf afba afbb afbc afbd afbe afbf afca afcb afcc afcd afce afcf afda afdb afdc afdd afde afdf afea
afeb afec afed afee afef affa affb affc affd affe afff baaa baab baac baad baae baaf baba babb babc babd babe babf baca bacb bacc bacd bace bacf bada badb badc badd bade badf baea
baeb baec baed baee baef bafa bafb bafc bafd bafe baff bbaa bbab bbac bbad bbae bbaf bbba bbbb bbbc bbbd bbbe bbbf bbca bbcb bbcc bbcd bbce bbcf bbda bbdb bbdc bbdd bbde
bbdf bbea bbeb bbec bbed bbee bbef bbfa bbfb bbfc bbfd bbfe bbff bcaa bcab bcac bcad bcae bcaf bcba bcbb bcbc bcbd bcbe bcbf bcca bccb bccc bccd bcce bccf bcda bcdb bcdc bcdd
bcde bcdf bcea bceb bcec bced bcee bcef bcfa bcfb bcfc bcfd bcfe bcff bdaa bdab bdac bdad bdae bdaf bdba bdbb bdbc bdbd bdbe bdbf bdca bdcb bdcc bdcd bdce bdcf bdda bddb
bddc bddd bdde bddf bdea bdeb bdec bded bdee bdef bdfa bdfb bdfc bdfd bdfe bdff beaa beab beac bead beae beaf beba bebb bebc bebd bebe bebf beca becb becc becd bece becf
beda bedb bedc bedd bede bedf beea beeb beec beed beee beef befa befb befc befd befe beff bfaa bfab bfac bfad bfae bfaf bfba bfbb bfbc bfbd bfbe bfbf bfca bfcb bfcc bfcd bfce bfcf
bfda bfdb bfdc bfdd bfde bfdf bfea bfeb bfec bfed bfee bfef bffa bffb bffc bffd bffe bfff caaa caab caac caad caae caaf caba cabb cabc cabd cabe cabf caca cacb cacc cacd cace cacf cada
cadb cadc cadd cade cadf caea caeb caec caed caee caef cafa cafb cafc cafd cafe caff cbaa cbab cbac cbad cbae cbaf cbba cbbb cbbc cbbd cbbe cbbf cbca cbcb cbcc cbcd cbce cbcf cbda
cbdb cbdc cbdd cbde cbdf cbea cbeb cbec cbed cbee cbef cbfa cbfb cbfc cbfd cbfe cbff ccaa ccab ccac ccad ccae ccaf ccba ccbb ccbc ccbd ccbe ccbf ccca cccb cccc cccd ccce cccf ccda
ccdb ccdc ccdd ccde ccdf ccea cceb ccec cced ccee ccef ccfa ccfb ccfc ccfd ccfe ccff cdaa cdab cdac cdad cdae cdaf cdba cdbb cdbc cdbd cdbe cdbf cdca cdcb cdcc cdcd cdce cdcf cdda
cddb cddc cddd cdde cddf cdea cdeb cdec cded cdee cdef cdfa cdfb cdfc cdfd cdfe cdff ceaa ceab ceac cead ceae ceaf ceba cebb cebc cebd cebe cebf ceca cecb cecc cecd cece cecf ceda
cedb cedc cedd cede cedf ceea ceeb ceec ceed ceee ceef cefa cefb cefc cefd cefe ceff cfaa cfab cfac cfad cfae cfaf cfba cfbb cfbc cfbd cfbe cfbf cfca cfcb cfcc cfcd cfce cfcf cfda cfdb cfdc
cfdd cfde cfdf cfea cfeb cfec cfed cfee cfef cffa cffb cffc cffd cffe cfff daaa daab daac daad daae daaf daba dabb dabc dabd dabe dabf daca dacb dacc dacd dace dacf dada dadb dadc dadd
dade dadf daea daeb daec daed daee daef dafa dafb dafc dafd dafe daff dbaa dbab dbac dbad dbae dbaf dbba dbbb dbbc dbbd dbbe dbbf dbca dbcb dbcc dbcd dbce dbcf dbda dbdb
dbdc dbdd dbde dbdf dbea dbeb dbec dbed dbee dbef dbfa dbfb dbfc dbfd dbfe dbff dcaa dcab dcac dcad dcae dcaf dcba dcbb dcbc dcbd dcbe dcbf dcca dccb dccc dccd dcce dccf dcda
dcdb dcdc dcdd dcde dcdf dcea dceb dcec dced dcee dcef dcfa dcfb dcfc dcfd dcfe dcff ddaa ddab ddac ddad ddae ddaf ddba ddbb ddbc ddbd ddbe ddbf ddca ddcb ddcc ddcd ddce
ddcf ddda dddb dddc dddd ddde dddf ddea ddeb ddec dded ddee ddef ddfa ddfb ddfc ddfd ddfe ddff deaa deab deac dead deae deaf deba debb debc debd debe debf deca decb decc
decd dece decf deda dedb dedc dedd dede dedf deea deeb deec deed deee deef defa defb defc defd defe deff dfaa dfab dfac dfad dfae dfaf dfba dfbb dfbc dfbd dfbe dfbf dfca dfcb dfcc
dfcd dfce dfcf dfda dfdb dfdc dfdd dfde dfdf dfea dfeb dfec dfed dfee dfef dffa dffb dffc dffd dffe dfff eaaa eaab eaac eaad eaae eaaf eaba eabb eabc eabd eabe eabf eaca eacb eacc eacd
eace eacf eada eadb eadc eadd eade eadf eaea eaeb eaec eaed eaee eaef eafa eafb eafc eafd eafe eaff ebaa ebab ebac ebad ebae ebaf ebba ebbb ebbc ebbd ebbe ebbf ebca ebcb ebcc
```

Figure 7: All permutations used in bruteforce leading to the one that successfully breaks the encryption is shown, then user can click on proceed above



Decrypted Password: OOP-Encrypt

Total Time to Decrypt: 1.1984 Milliseconds

Salt Used: ffeba

Figure 8: Display Screen for the time taken, salt used and decrypted password in plain text

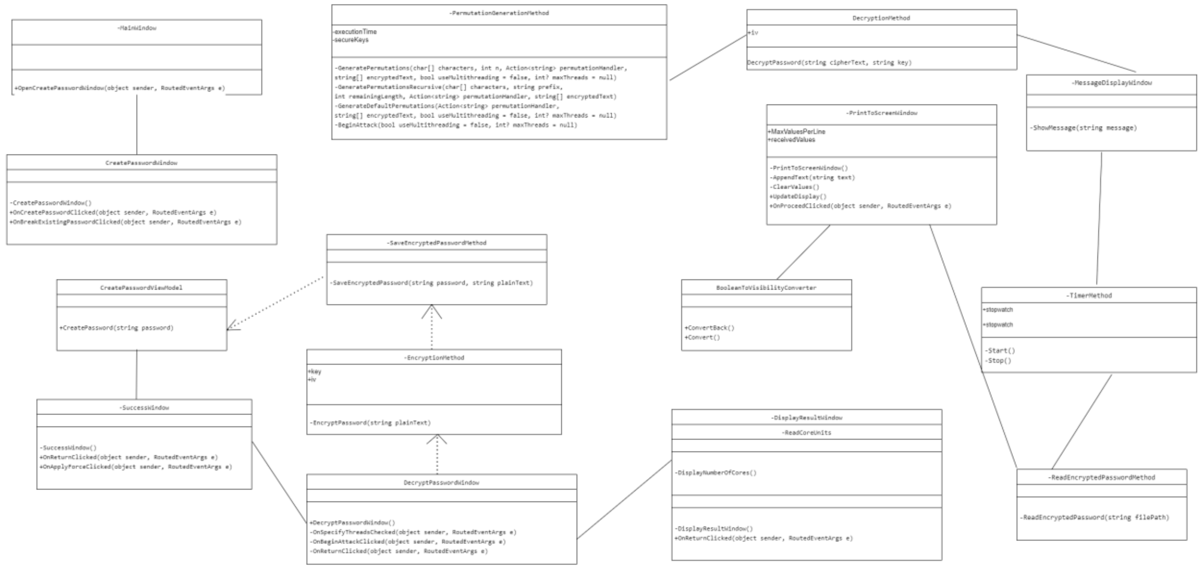

-MainWindow
+OpenCreatePasswordWindow(object sender, RoutedEventArgs e)
-PermutationGenerationMethod
-executionTime
-secureKeys
-GeneratePermutations(char[] characters, int n, Action<string> permutationHandler,
string[] encryptedText, bool useMultithreading = false, int? maxThreads = null)
-GeneratePermutationsRecursive(char[] characters, string prefix,
int remainingLength, Action<string> permutationHandler, string[] encryptedText)
-GenerateDefaultPermutations(Action<string> permutationHandler,
string[] encryptedText, bool useMultithreading = false, int? maxThreads = null)
-BeginAttack(bool useMultithreading = false, int? maxThreads = null)
DecryptionMethod
+iv
DecryptPassword(string cipherText, string key)
-MessageDisplayWindow
-ShowMessage(string message)
CreatePasswordWindow
-CreatePasswordWindow()
+OnCreatePasswordClicked(object sender, RoutedEventArgs e)
+OnBreakExistingPasswordClicked(object sender, RoutedEventArgs e)
-PrintToScreenWindow
+MaxValuesPerLine
+receivedValues
-PrintToScreenWindow()
-AppendText(string text)
-ClearValues()
+UpdateDisplay()
+OnProceedClicked(object sender, RoutedEventArgs e)
-SaveEncryptedPasswordMethod
-SaveEncryptedPassword(string password, string plainText)
CreatePasswordViewModel
+CreatePassword(string password)
BooleanToVisibilityConverter
+ConvertBack()
+Convert()
-TimerMethod
+stopwatch
+stopwatch
-Start()
-Stop()
-EncryptionMethod
+key
+iv
-EncryptPassword(string plainText)
-SuccessWindow
-SuccessWindow()
+OnReturnClicked(object sender, RoutedEventArgs e)
+OnApplyForceClicked(object sender, RoutedEventArgs e)
-DisplayResultWindow
-ReadCoreUnits
-DisplayNumberOfCores()
-DisplayResultWindow()
+OnReturnClicked(object sender, RoutedEventArgs e)
DecryptPasswordWindow
+DecryptPasswordWindow()
-OnSpecifyThreadsChecked(object sender, RoutedEventArgs e)
-OnBeginAttackClicked(object sender, RoutedEventArgs e)
-OnReturnClicked(object sender, RoutedEventArgs e)
-ReadEncryptedPasswordMethod
-ReadEncryptedPassword(string filePath)

Figure 9 ML Diagram

Link to Github repository: https://github.com/Emperor-Trillion/Brute-Force-C-VU

Conclusion: The use of brute force algorithm is not a very efficient means of decryption. I can take many years for supercomputers to decrypt a password this way.