King Abdelaziz University

Faculty of Computing and Information Technology

Department of Information Technology

CPIT-425 - Information Security

(Massage Authentication Using MD5)

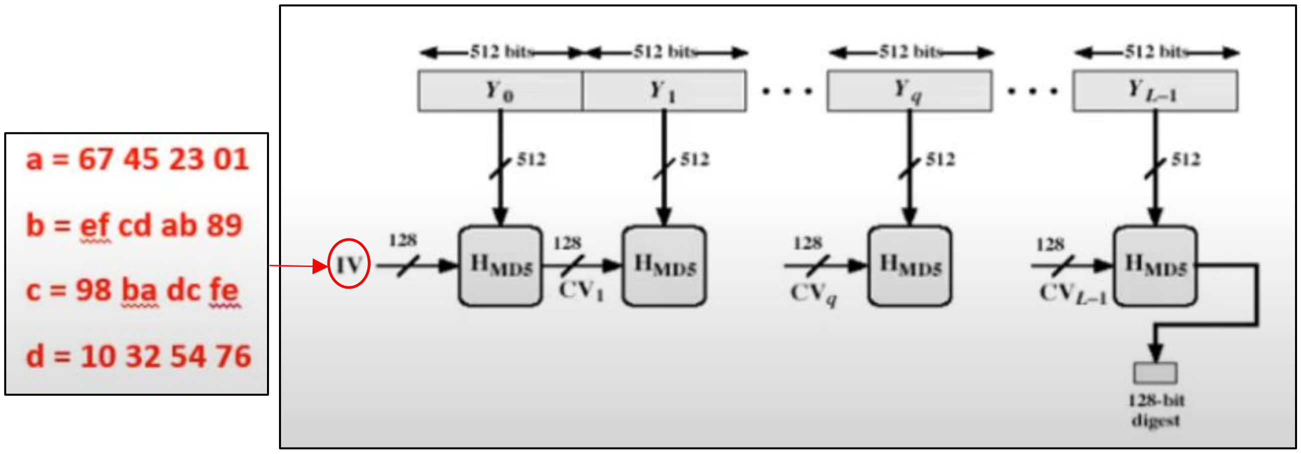
Final Project

Instructor Name: Dr. Omar Abdullah Batarfi

Name	ID
Shehab *****	******
Sattam *****	******

Table of Contents

MD5 Algorithm



MD5 is considers as **hash function**, means it's **one way** process and it's impossible to recover the original data out of its output.

How is it working?

- Data in any size will be divided into blocks of 512-bit for each block.
- four buffers with a total size of 128-bit, 32-bit for each, contains initial values (initial buffer) must be created (We can see it in the above figure "**IV"** the values are standard in the original algorithm).
- The result will be with size of 128-bit, and it will be used in the next block and so on until the last block
- The last block contains the result of the MD5 algorithm.

Let us take a detailed example…

Let us assume that the length of the data is **3000-bit**.

- We can calculate the number of **blocks** by the following equation:

(Length of the data (in bit)) / **512** (**length of each block** (in bit**))** = Number of the blocks.

3000 / 512 = 5.8 ≈ 6 blocks (We ceil the result if it's decimal number (not integer)).

So, 512+512+512+512+512 = 2560 bit for the first five blocks.

For the **last block**, we have the following equation:

Last block = Data + Padding + Length of the data

Where:

Data = (The remaining data)

Padding = (The purpose of the padding is to complete the bits until it reaches to 512-bit)

Length of the message = (Represent the data in binary form)

Last block =

(3000-(512*5)) + (512-(440+64) + (represent the 3000bit as 64-bit in binary)

Last block =

440 + 8 + **(This the represent of 3000 in binary >> (101110111000), now we need to add set of 0s left this number until it become 64 bit)**.

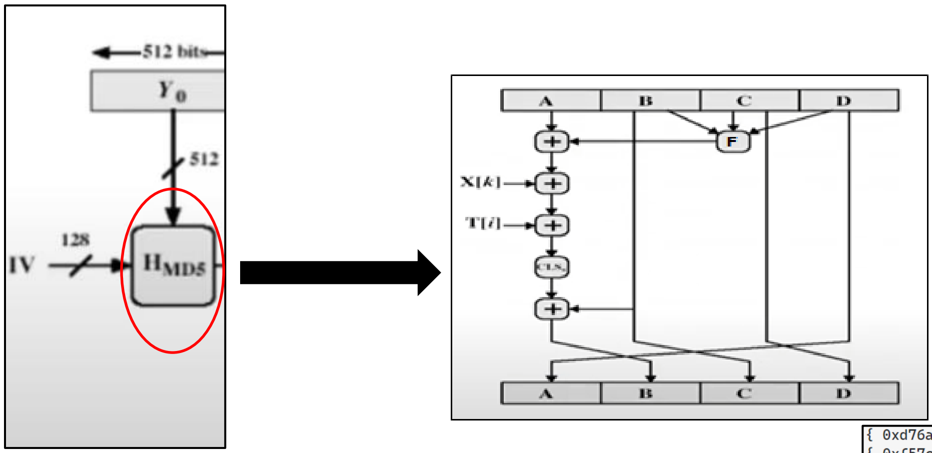
Last block =

440 + 8 + 000000000000000000000000000000000000000000000000000101110111000

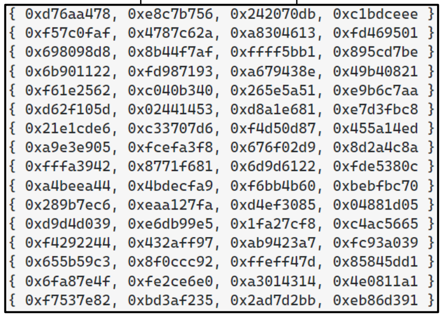
Every 512-bit(block) need to be split in 16 sectors.

512-bit

1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	11th	12th	13th	14th	15th	16th
32-bit	32-bit	32-bit	32-bit	32-bit	32-bit	32-bit	32-bit	32-bit	32-bit	32-bit	32-bit	32-bit	32-bit	32-bit	32-bit



Constant T



```
{ 0xd76aa478, 0xe8c7b756, 0x242070db, 0xc1bdceee }
{ 0xf57c0faf, 0x4787c62a, 0xa8304613, 0xfd469501 }
{ 0x698098d8, 0x8b44f7af, 0xffff5bb1, 0x895cd7be }
{ 0x6b901122, 0xfd987193, 0xa679438e, 0x49b40821 }
{ 0xf61e2562, 0xc040b340, 0x265e5a51, 0xe9b6c7aa }
{ 0xd62f105d, 0x02441453, 0xd8a1e681, 0xe7d3fbc8 }
{ 0x21e1cde6, 0xc33707d6, 0xf4d50d87, 0x455a14ed }
{ 0xa9e3e905, 0xfcefa3f8, 0x676f02d9, 0x8d2a4c8a }
{ 0xfffa3942, 0x8771f681, 0x6d9d6122, 0xfde5380c }
{ 0xa4beea44, 0x4bdecfa9, 0xf6bb4b60, 0xbebfbc70 }
{ 0x289b7ec6, 0xeaa127fa, 0xd4ef3085, 0x04881d05 }
{ 0xd9d4d039, 0xe6db99e5, 0x1fa27cf8, 0xc4ac5665 }
{ 0xf4292244, 0x432aff97, 0xab9423a7, 0xfc93a039 }
{ 0x655b59c3, 0x8f0ccc92, 0xffeff47d, 0x85845dd1 }
{ 0x6fa87e4f, 0xfe2ce6e0, 0xa3014314, 0x4e0811a1 }
{ 0xf7537e82, 0xbd3af235, 0x2ad7d2bb, 0xeb86d391 }
```

So, what is happening inside theses blocks?

This block contains 4 rounds, each round contains 16 steps

For the 1st round:

R = B + ((A + F(B,C,D) + X[K] + T[I]) <<<S)

Where:

- **A,B,C,D** are the buffers
- **F** is a law, F(b,c,d) = (B&C) | (!B & D)
- **X** is the data(32-bit) that got split in the previous step, and each step next sector is used.
- **T** is constant, for the first round the first 16 constant is used, and for the next round the next 16 constant is used and so on.
- **S** is the amount of shift to left, standard, each line used for 1 round

Shift table

```
{7 ,12 ,17 ,22 ,7 ,12 ,17 ,22 ,7 ,12 ,17 ,22 ,7 ,12 ,17 ,22}
{5 ,9 ,14 ,20 ,5 ,9 ,14 ,20 ,5 ,9 ,14 ,20 ,5 ,9 ,14 ,20}
{4 ,11 ,16 ,23 ,4 ,11 ,16 ,23 ,4 ,11 ,16 ,23 ,4 ,11 ,16 ,23}
{6 ,10 ,15 ,21 ,6 ,10 ,15 ,21 ,6 ,10 ,15 ,21 ,6 ,10 ,15 ,21}
```

Note: in the end of each step, we must rearrange the buffers.

(New) A = (Old) D

(New) D = (Old) C

(New) C = (Old) B

(New) B = (Old) "R => The result of the above law"

The difference between each round is the **function** that will be used and here is a table to explain it:

Round	1st	2nd	3rd	4th
Function	**F**(b,c,d) = (b&c) || (!b & d)	**G**(b,c,d) = (b&d) || (c & !d)	**H**(b,c,d) = (b XOR c XOR d)	**I**(b,c,d) = c XOR (b || !d)

And in the last round the final buffers are:

A = A(IV) + A(new), **B** = B(IV) + B(new), **C** = C(IV) + C(new), **D** = D(IV) + D(new)

A B C D(concatenate) = 128 bit

The reason for using MD5 over SHA2.

- We used MD5 because less time to calculate then SHA-2.
- SHA2 is difficult to handle because of its size (MD5 result = 128-bit)(SHA2 result = 256-bit)
- MD5 can convert data of any length while SHA2 only can handle at maximum size less than (2^64).

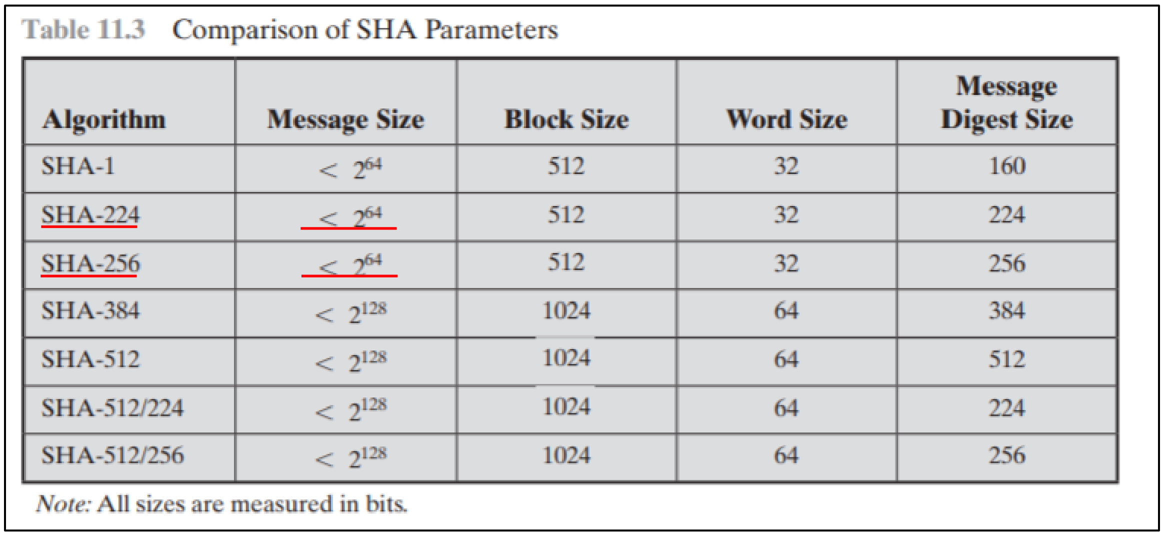


Table 11.3 Comparison of SHA Parameters

Algorithm	Message Size	Block Size	Word Size	Message Digest Size
SHA-1	$< 2^{64}$	512	32	160
SHA-224	$< 2^{64}$	512	32	224
SHA-256	$< 2^{64}$	512	32	256
SHA-384	$< 2^{128}$	1024	64	384
SHA-512	$< 2^{128}$	1024	64	512
SHA-512/224	$< 2^{128}$	1024	64	224
SHA-512/256	$< 2^{128}$	1024	64	256

Note: All sizes are measured in bits.

Methods Explanation

- **rseadfileName();** returns file name without the extension for example if we send

  (Info.txt) to this method it'll return (info).

file name as input

Creating string variable so later we store file name in it

For loop to compare every letter through the filename until it reaches '.'

Return the file name without the extension

```
private String readFileName(String fileName) {

    String fileNameWithoutExtension = "";


     for (int i = 0; i < fileName.length(); i++) {
         if (fileName.charAt(i) == '.') {
             return fileNameWithoutExtension;
         }
         fileNameWithoutExtension = fileNameWithoutExtension +
fileName.charAt(i);
     }
     return fileNameWithoutExtension;
}
```

- **hashValue();** returns the hash as a string also creates TheHashedFileName(hash).txt containing the hash.

file name as input

getMD5Checksum(); is a method that calculates the hash and return it as a string then it'll be stored in hash.

The rest of the lines just to store the hash in a txt file which has the same name of the file + (hash) in the end it'll return the hash as a string.

```
private String HashValue(String fileName) throws Exception {

         String hash = getMD5Checksum(fileName);




         PrintWriter out = new PrintWriter(readFileName(fileName) +
"(hash).txt");
         out.println(hash);out.close();
         return hash;
     }
}
```

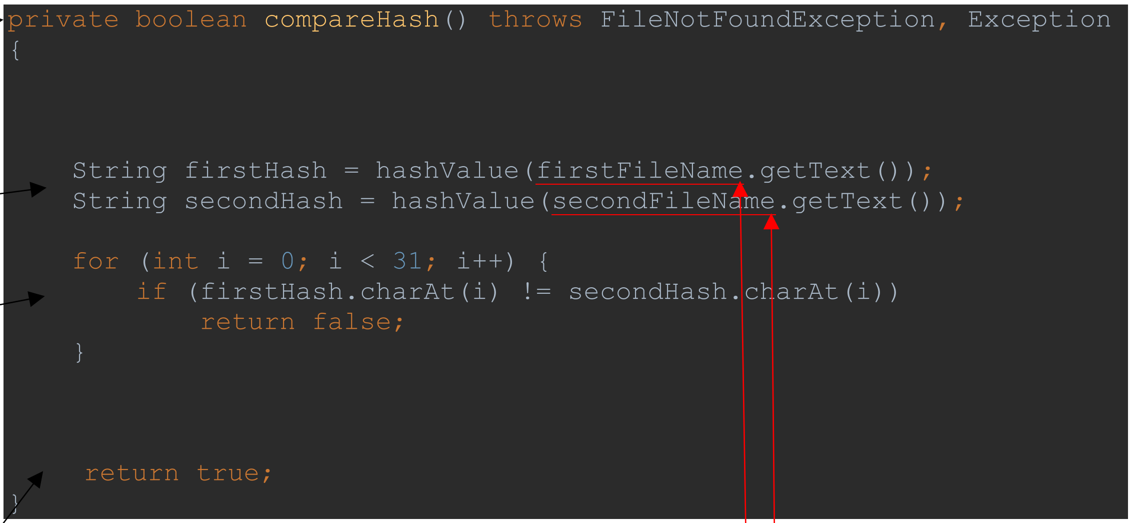
- **compareHash();** this method compares the two provided files hash if they are the same it'll return true else it'll return false.

Returns boolean and no parameters needed since the files will be obtained from the text areas

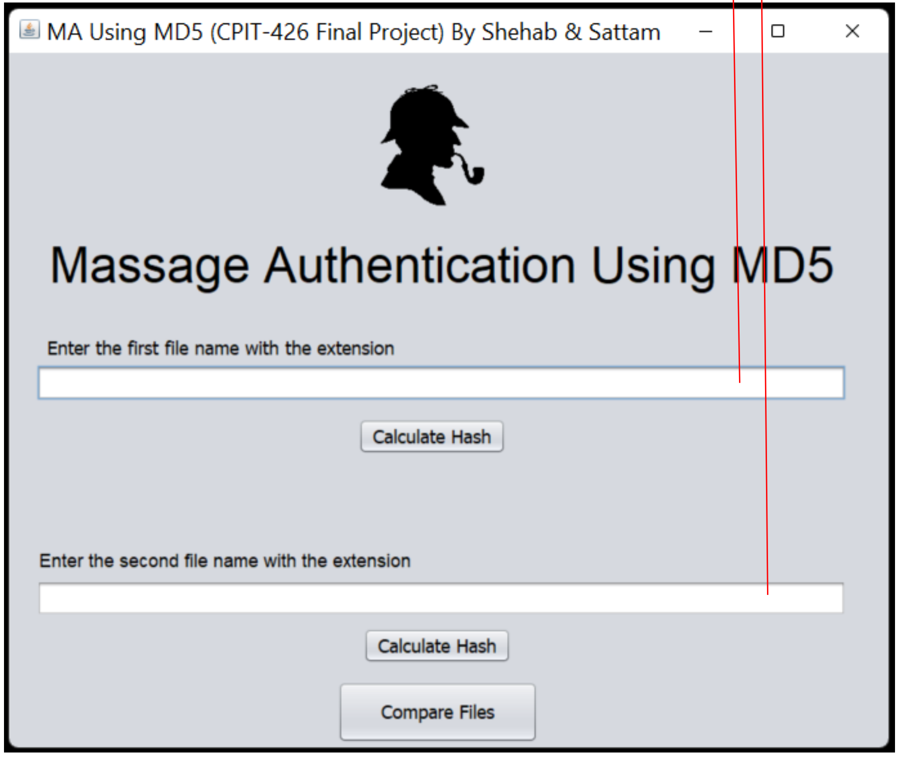
Bringing the hash values and store them in two strings

For loop is set to 31(since MD5 implementations produce a 32-character, hexadecimal-formatted hash) compares every two char at the same index return false if not equal

Otherwise return true.



```
private boolean compareHash() throws FileNotFoundException, Exception
{


    String firstHash = hashValue(firstFileName.getText());
    String secondHash = hashValue(secondFileName.getText());

    for (int i = 0; i < 31; i++) {
        if (firstHash.charAt(i) != secondHash.charAt(i))
            return false;
    }



     return true;
}
```



MA Using MD5 (CPIT-426 Final Project) By Shehab & Sattam
Massage Authentication Using MD5
Enter the first file name with the extension
Calculate Hash
Enter the second file name with the extension
Calculate Hash
Compare Files

- createChecksum(); this method takes the file name as a parameter then convert all its characters to bytes and send it to md5 algorithm and return 16 bytes hashed in bytes.

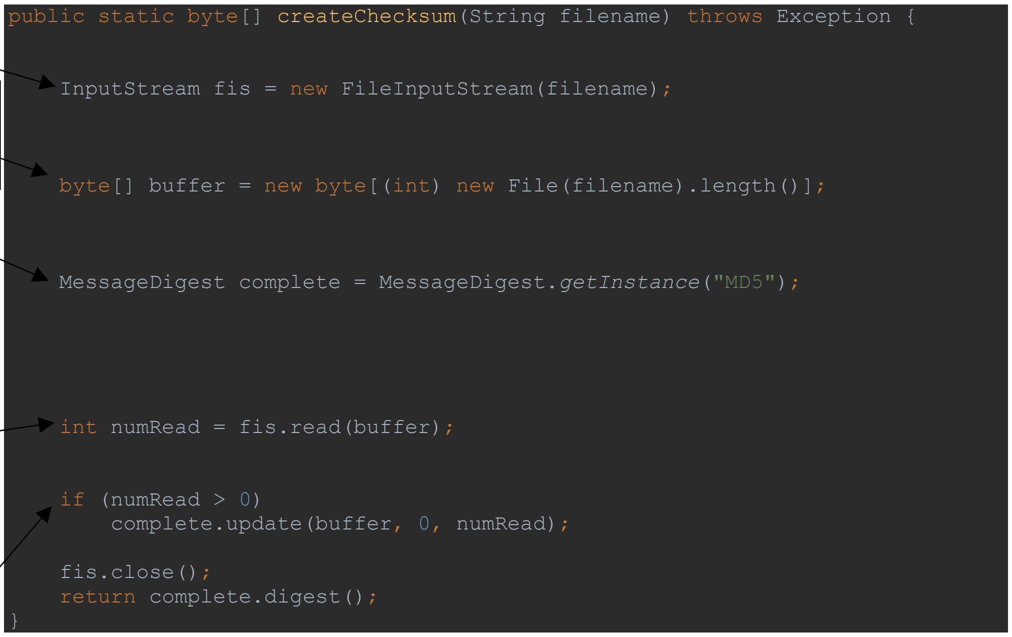
Create an object of class file input stream to help us read from file (we can use scanner instead of input stream)

Create an array of bytes has the same length of the file that we will read

This is an object from class java.security.messageDigest which has the method getInstance(), user provide it with the name of algorithm, and it uses it.

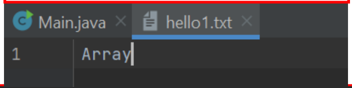
numRead to store the number of characters, fis reads all the chars and store it in array buffer

If number of chars is greater than 0 it uses the array, (0) the index that it will start from, and in the end it'll return an array of 16 bytes just needs to convert to hexadecimal.

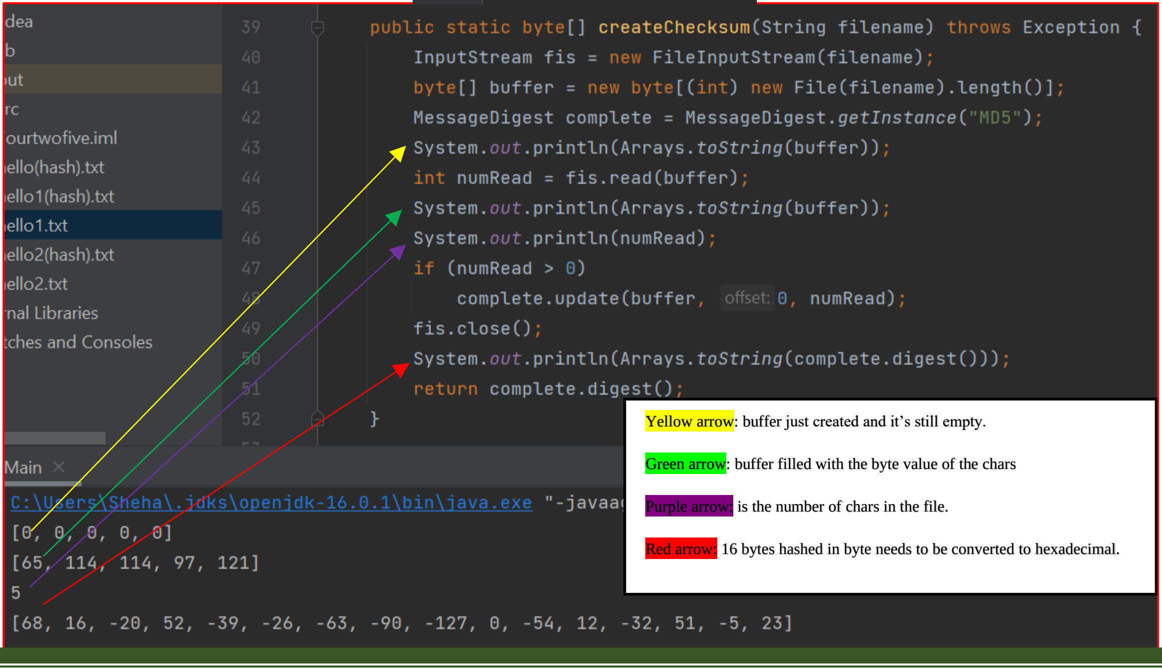


```java
public static byte[] createChecksum(String filename) throws Exception {

    InputStream fis = new FileInputStream(filename);

    byte[] buffer = new byte[(int) new File(filename).length()];

    MessageDigest complete = MessageDigest.getInstance("MD5");

    int numRead = fis.read(buffer);

    if (numRead > 0)
        complete.update(buffer, 0, numRead);

    fis.close();
    return complete.digest();
}
```

Simple example on this txt file.



```
Main.java   hello1.txt
1   Array
```



```java
public static byte[] createChecksum(String filename) throws Exception {
    InputStream fis = new FileInputStream(filename);
    byte[] buffer = new byte[(int) new File(filename).length()];
    MessageDigest complete = MessageDigest.getInstance("MD5");
    System.out.println(Arrays.toString(buffer));
    int numRead = fis.read(buffer);
    System.out.println(Arrays.toString(buffer));
    System.out.println(numRead);
    if (numRead > 0)
        complete.update(buffer, offset: 0, numRead);
    fis.close();
    System.out.println(Arrays.toString(complete.digest()));
    return complete.digest();
}
```

```
C:\Users\Sheha\.jdks\openjdk-16.0.1\bin\java.exe "-javaa
[0, 0, 0, 0, 0]
[65, 114, 114, 97, 121]
5
[68, 16, -20, 52, -39, -26, -63, -90, -127, 0, -54, 12, -32, 51, -5, 23]
```

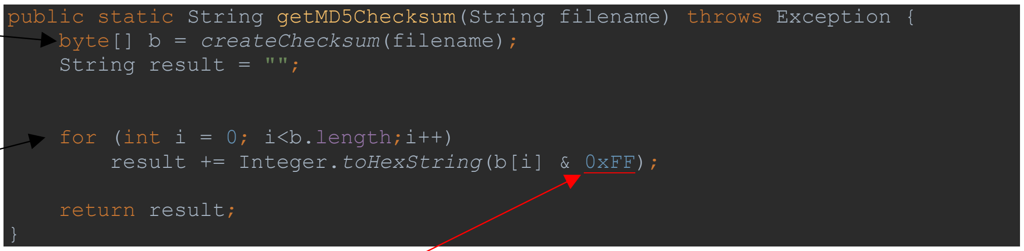
Yellow arrow: buffer just created and it's still empty.

Green arrow: buffer filled with the byte value of the chars

Purple arrow: is the number of chars in the file.

Red arrow: 16 bytes hashed in byte needs to be converted to hexadecimal.

getMD5Checksum(); Method that converts array elements from bytes to hexadecimal and return it as a string.



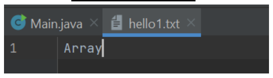
```
public static String getMD5Checksum(String filename) throws Exception {
    byte[] b = createChecksum(filename);
    String result = "";


    for (int i = 0; i<b.length;i++)
        result += Integer.toHexString(b[i] & 0xFF);

    return result;
}
```

Create an array of bytes so we can move through the array using the for loop below, empty string to store results later.

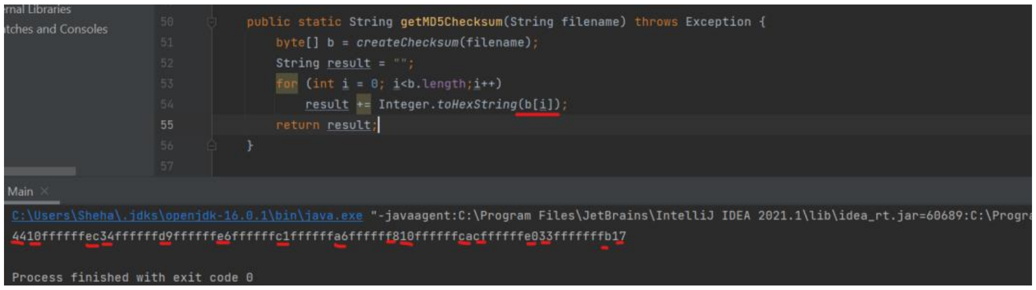
For loop has the same length as the array, now we store the conversion in the result string we created earlier, and in the end we returned 32 character.

This helps us to take the two-right digits because in some cases 'f' do repeat few times

Example

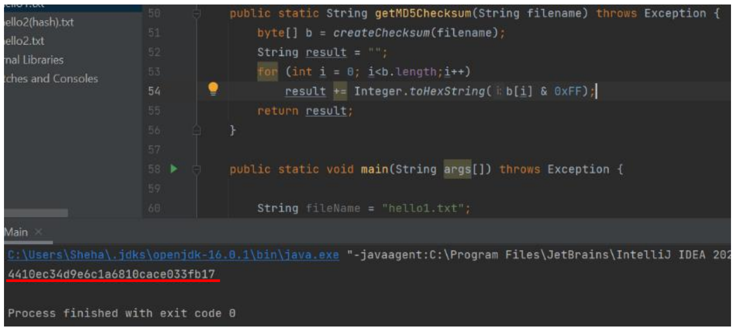


```
Main.java    hello1.txt
1    Array
```



```
public static String getMD5Checksum(String filename) throws Exception {
    byte[] b = createChecksum(filename);
    String result = "";
    for (int i = 0; i<b.length;i++)
        result += Integer.toHexString(b[i]);
    return result;
}
```

```
Main
C:\Users\Sheha\.jdks\openjdk-16.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2021.1\lib\idea_rt.jar=60689:C:\Progra
4410ffffffec34ffffffd9ffffffe6ffffffc1ffffffa6ffffff810ffffffcacffffffe033fffffffb17

Process finished with exit code 0
```



```
public static String getMD5Checksum(String filename) throws Exception {
    byte[] b = createChecksum(filename);
    String result = "";
    for (int i = 0; i<b.length;i++)
        result += Integer.toHexString( b[i] & 0xFF);
    return result;
}

public static void main(String args[]) throws Exception {

    String fileName = "hello1.txt";
```

```
Main
C:\Users\Sheha\.jdks\openjdk-16.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 202
4410ec34d9e6c1a6810cace033fb17

Process finished with exit code 0
```

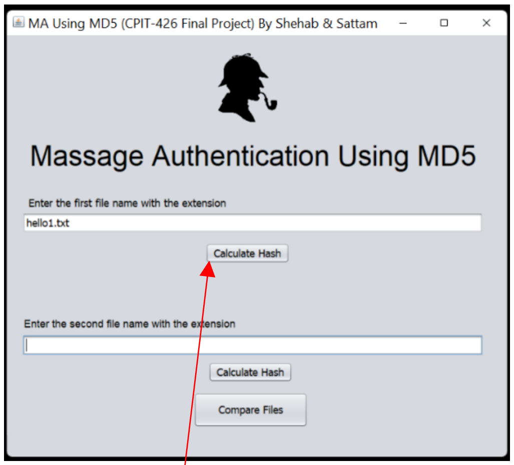
Data Flow

- After the user enters the file name in the first text area with the extension. User will click (Calculate Hash) button.

- The button method will first take what the user typed in the text area then send it to method called hashValue(we used try and catch because the button method is fixed and we couldn't throw exception)

- **hashValue()** takes the file name as a parameter then use other method called **getMD5Checksum()** call the method **creatChecksum()** which return array of byte which represents the hash in bytes, **so getMD5Checksum()** convert it to hexadecimal.

- Same goes for the second Calculate Hash and in comparing button the below code executes.

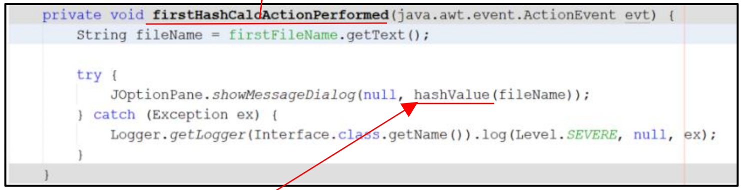


MA Using MD5 (CPIT-426 Final Project) By Shehab & Sattam

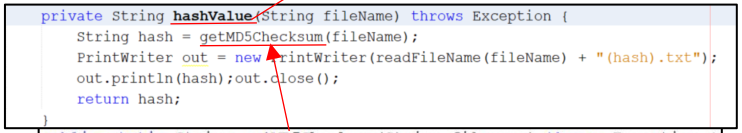
Massage Authentication Using MD5

Enter the first file name with the extension
hello1.txt

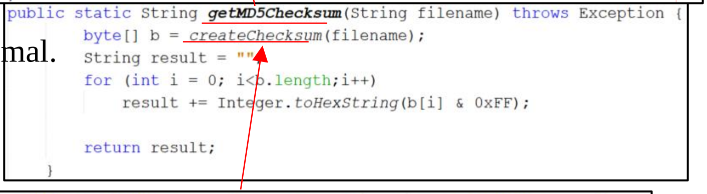
Enter the second file name with the extension



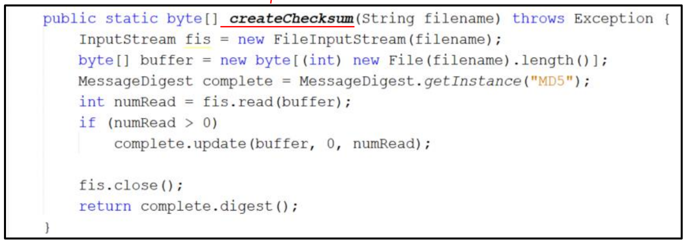
```
private void firstHashCalcActionPerformed(java.awt.event.ActionEvent evt) {
    String fileName = firstFileName.getText();

    try {
        JOptionPane.showMessageDialog(null, hashValue(fileName));
    } catch (Exception ex) {
        Logger.getLogger(Interface.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```



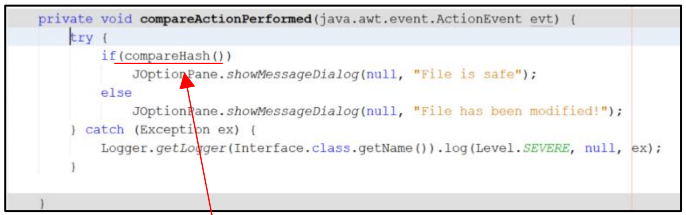
```
private String hashValue(String fileName) throws Exception {
    String hash = getMD5Checksum(fileName);
    PrintWriter out = new PrintWriter(readFileName(fileName) + "(hash).txt");
    out.println(hash);out.close();
    return hash;
}
```



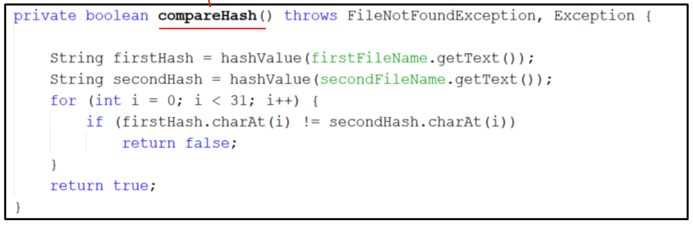
```
public static String getMD5Checksum(String filename) throws Exception {
        byte[] b = createChecksum(filename);
        String result = "";
        for (int i = 0; i<b.length;i++)
            result += Integer.toHexString(b[i] & 0xFF);

        return result;
    }
```



```
public static byte[] createChecksum(String filename) throws Exception {
    InputStream fis = new FileInputStream(filename);
    byte[] buffer = new byte[(int) new File(filename).length()];
    MessageDigest complete = MessageDigest.getInstance("MD5");
    int numRead = fis.read(buffer);
    if (numRead > 0)
        complete.update(buffer, 0, numRead);

    fis.close();
    return complete.digest();
}
```



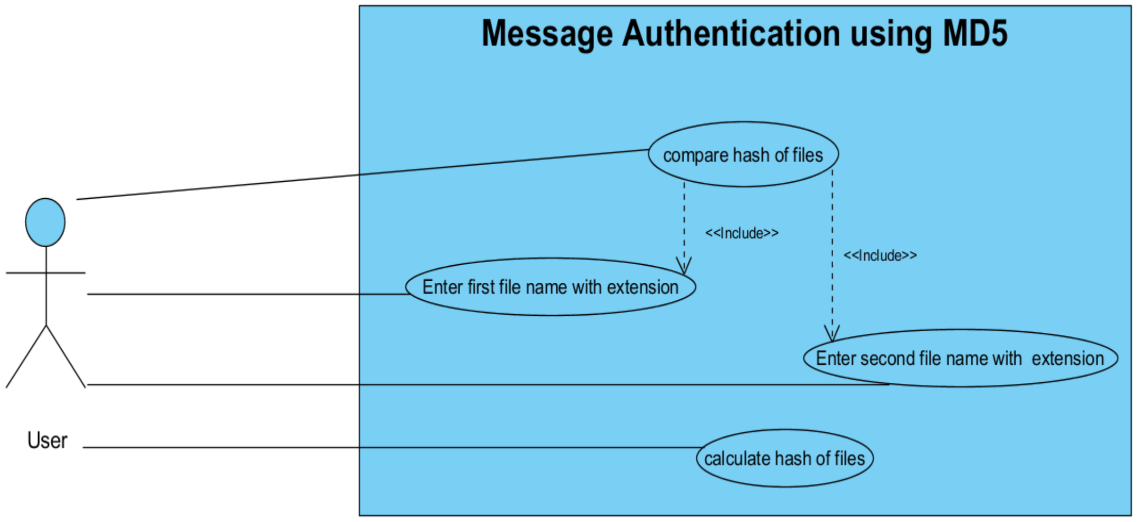
```
private void compareActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        if(compareHash())
            JOptionPane.showMessageDialog(null, "File is safe");
        else
            JOptionPane.showMessageDialog(null, "File has been modified!");
    } catch (Exception ex) {
        Logger.getLogger(Interface.class.getName()).log(Level.SEVERE, null, ex);
    }

}
```



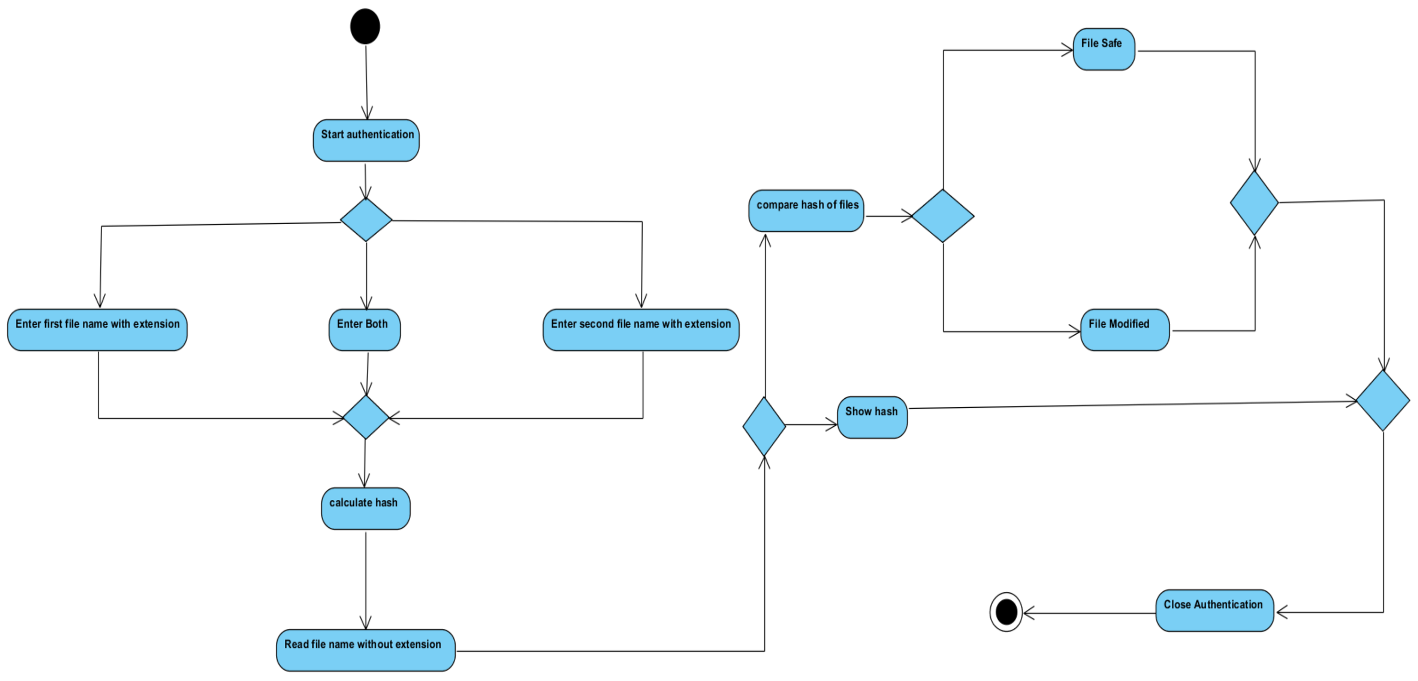
```
private boolean compareHash() throws FileNotFoundException, Exception {

    String firstHash = hashValue(firstFileName.getText());
    String secondHash = hashValue(secondFileName.getText());
    for (int i = 0; i < 31; i++) {
        if (firstHash.charAt(i) != secondHash.charAt(i))
            return false;
    }
    return true;
}
```

Diagrams

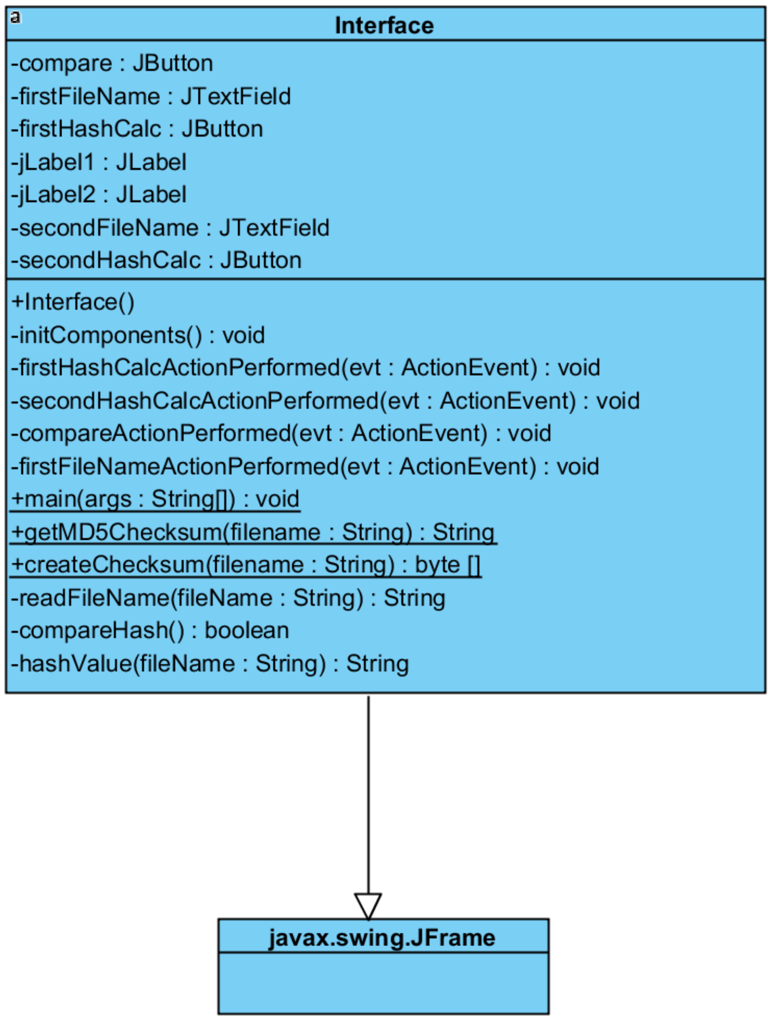
Use Case Diagram


Message Authentication using MD5
compare hash of files
<<Include>>
<<Include>>
Enter first file name with extension
Enter second file name with extension
calculate hash of files
User

Activity diagram



Class Diagram

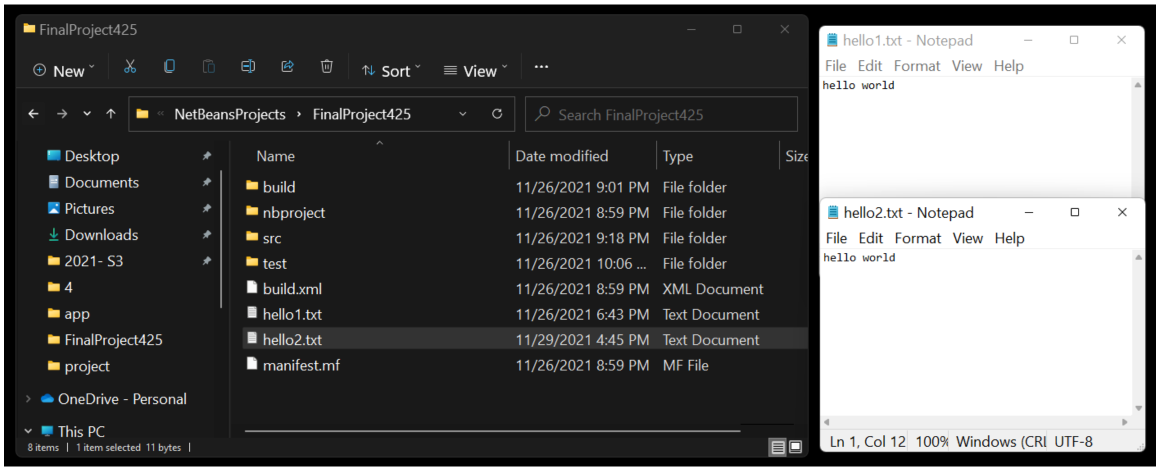

a
Interface
-compare : JButton
-firstFileName : JTextField
-firstHashCalc : JButton
-jLabel1 : JLabel
-jLabel2 : JLabel
-secondFileName : JTextField
-secondHashCalc : JButton
+Interface()
-initComponents() : void
-firstHashCalcActionPerformed(evt : ActionEvent) : void
-secondHashCalcActionPerformed(evt : ActionEvent) : void
-compareActionPerformed(evt : ActionEvent) : void
-firstFileNameActionPerformed(evt : ActionEvent) : void
+main(args : String[]) : void
+getMD5Checksum(filename : String) : String
+createChecksum(filename : String) : byte []
-readFileName(fileName : String) : String
-compareHash() : boolean
-hashValue(fileName : String) : String
javax.swing.JFrame

FACULTY OF COMPUTING & INFORMATION TECHNOLOGY
KING ABDULAZIZ UNIVERSITY

كلية الحاسبات وتقنية المعلومات
جامعة الملك عبدالعزيز

Samples

- Sample(1)



FinalProject425

NetBeansProjects › FinalProject425

Name	Date modified	Type
build	11/26/2021 9:01 PM	File folder
nbproject	11/26/2021 8:59 PM	File folder
src	11/26/2021 9:18 PM	File folder
test	11/26/2021 10:06 ...	File folder
build.xml	11/26/2021 8:59 PM	XML Document
hello1.txt	11/26/2021 6:43 PM	Text Document
hello2.txt	11/29/2021 4:45 PM	Text Document
manifest.mf	11/26/2021 8:59 PM	MF File

8 items | 1 item selected 11 bytes |

hello1.txt - Notepad

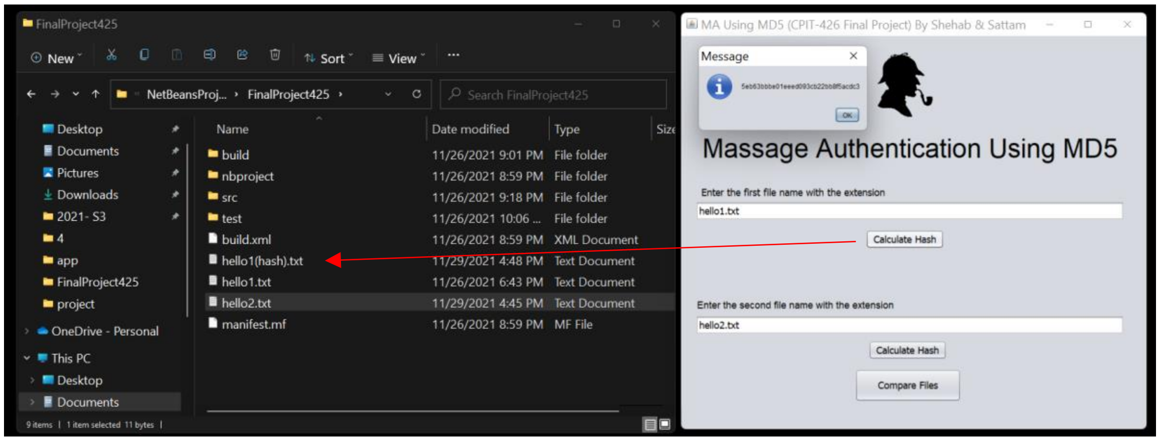
```
hello world
```

hello2.txt - Notepad

```
hello world
```

Ln 1, Col 12 100% Windows (CRL UTF-8

These are the two files which has the content of “hello world” and the names are **hello1.txt** and **hello2.txt**, the program find each file hash then create a .txt file. Also, it can compare the hash.



FinalProject425

NetBeansProj... › FinalProject425

Name	Date modified	Type
build	11/26/2021 9:01 PM	File folder
nbproject	11/26/2021 8:59 PM	File folder
src	11/26/2021 9:18 PM	File folder
test	11/26/2021 10:06 ...	File folder
build.xml	11/26/2021 8:59 PM	XML Document
hello1(hash).txt	11/29/2021 4:48 PM	Text Document
hello1.txt	11/26/2021 6:43 PM	Text Document
hello2.txt	11/29/2021 4:45 PM	Text Document
manifest.mf	11/26/2021 8:59 PM	MF File

9 items | 1 item selected 11 bytes |

MA Using MD5 (CPIT-426 Final Project) By Shehab & Sattam

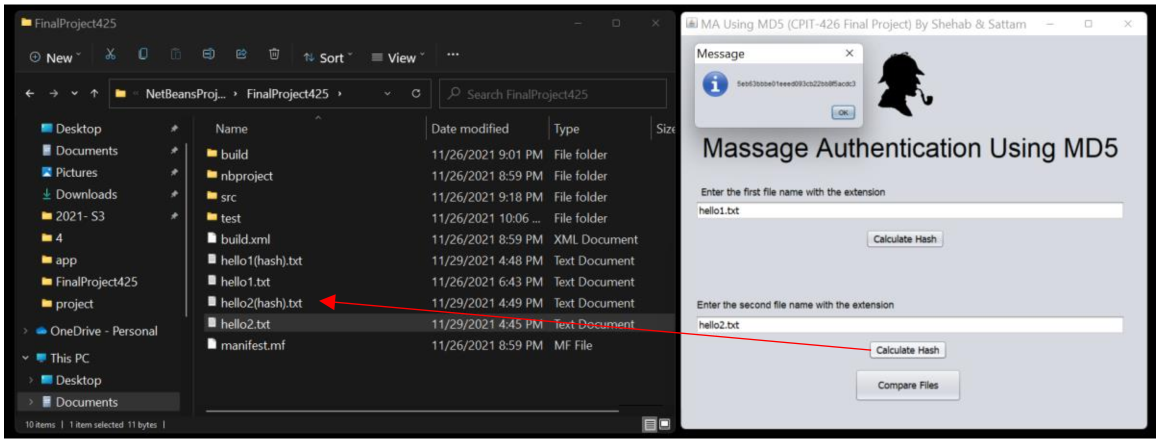
Message

```
5eb63bbbe01eeed093cb22bb8f5acdc3
```

Massage Authentication Using MD5

Enter the first file name with the extension: hello1.txt

Enter the second file name with the extension: hello2.txt

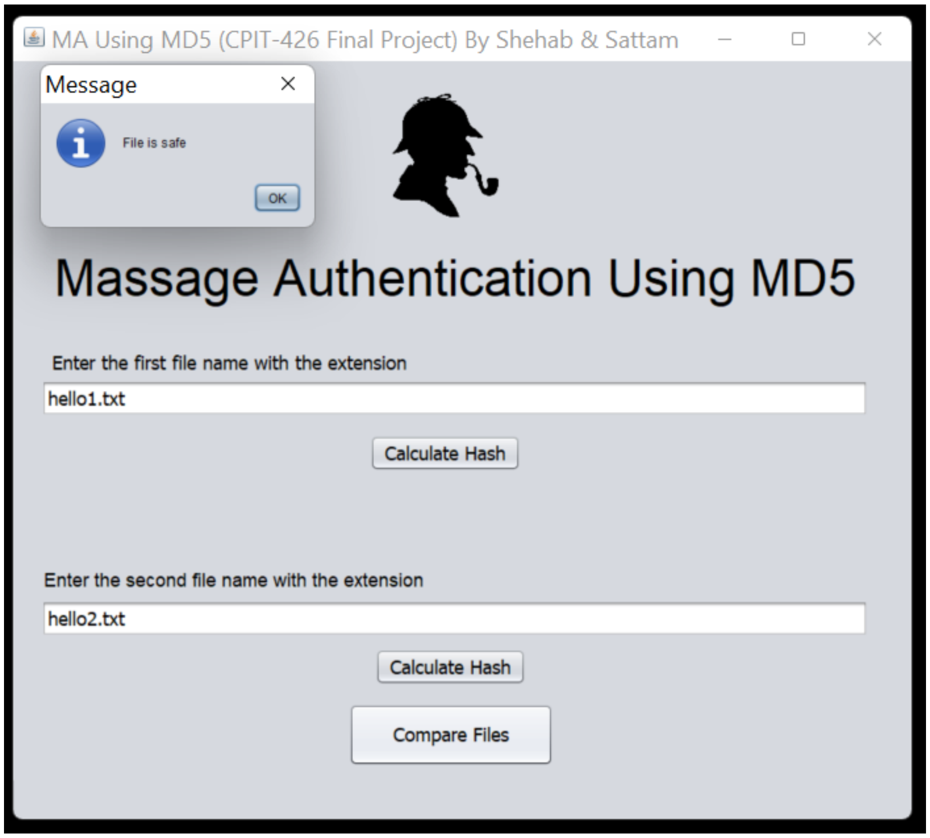

FinalProject425
New
Sort
View
NetBeansProj... › FinalProject425 ›
Search FinalProject425
Desktop
Documents
Pictures
Downloads
2021- S3
4
app
FinalProject425
project
OneDrive - Personal
This PC
Desktop
Documents
10 items | 1 item selected 11 bytes |
MA Using MD5 (CPIT-426 Final Project) By Shehab & Sattam
Message
OK
Massage Authentication Using MD5
Enter the first file name with the extension
Calculate Hash
Enter the second file name with the extension
Calculate Hash
Compare Files

```
Name              Date modified        Type           Size
build             11/26/2021 9:01 PM   File folder
nbproject         11/26/2021 8:59 PM   File folder
src               11/26/2021 9:18 PM   File folder
test              11/26/2021 10:06 ... File folder
build.xml         11/26/2021 8:59 PM   XML Document
hello1(hash).txt  11/29/2021 4:48 PM   Text Document
hello1.txt        11/26/2021 6:43 PM   Text Document
hello2(hash).txt  11/29/2021 4:49 PM   Text Document
hello2.txt        11/29/2021 4:45 PM   Text Document
manifest.mf       11/26/2021 8:59 PM   MF File
```

```
5eb63bbbe01eeed093cb22bb8f5acdc3
```

```
hello1.txt
hello2.txt
```

After clicking (Calculate Hash) a massage will pop up containing the hash and a txt file will be created.

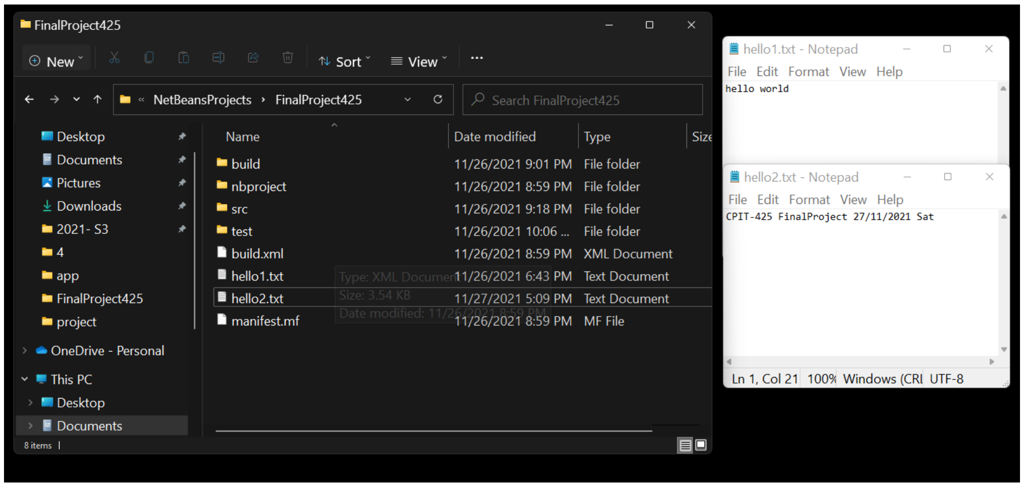

MA Using MD5 (CPIT-426 Final Project) By Shehab & Sattam
Message
OK
Massage Authentication Using MD5
Enter the first file name with the extension
Calculate Hash
Enter the second file name with the extension
Calculate Hash
Compare Files

```
File is safe
```

```
hello1.txt
hello2.txt
```

Since the files have the same hash, means the file have not been modified. So, it safe.

- Sample(2)



FinalProject425 › NetBeansProjects › FinalProject425

Name	Date modified	Type
build	11/26/2021 9:01 PM	File folder
nbproject	11/26/2021 8:59 PM	File folder
src	11/26/2021 9:18 PM	File folder
test	11/26/2021 10:06 ...	File folder
build.xml	11/26/2021 8:59 PM	XML Document
hello1.txt	11/26/2021 6:43 PM	Text Document
hello2.txt	11/27/2021 5:09 PM	Text Document
manifest.mf	11/26/2021 8:59 PM	MF File

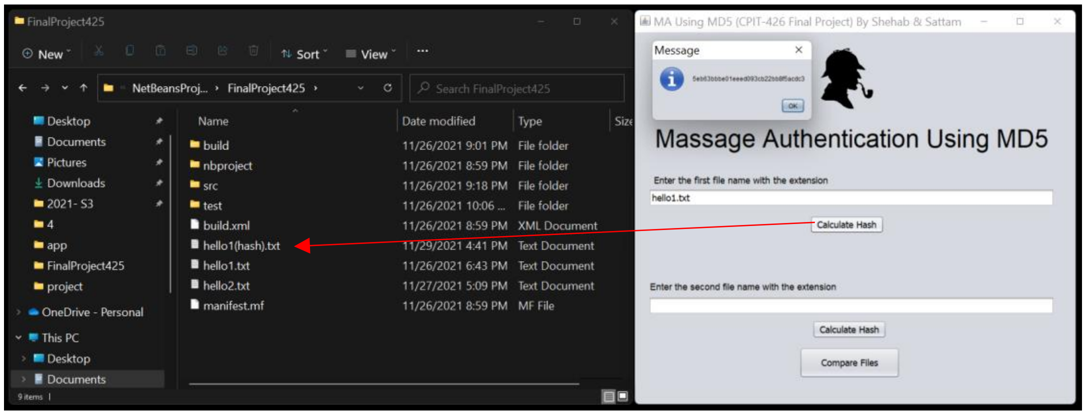
hello1.txt - Notepad

```
hello world
```

hello2.txt - Notepad

```
CPIT-425 FinalProject 27/11/2021 Sat
```

hello1.txt has the content of "<u>hello world</u>" and **hello2.txt** has the content "<u>CPIT-425 FinalProject 27/11/2021 Sat</u>".



MA Using MD5 (CPIT-426 Final Project) By Shehab & Sattam

Message

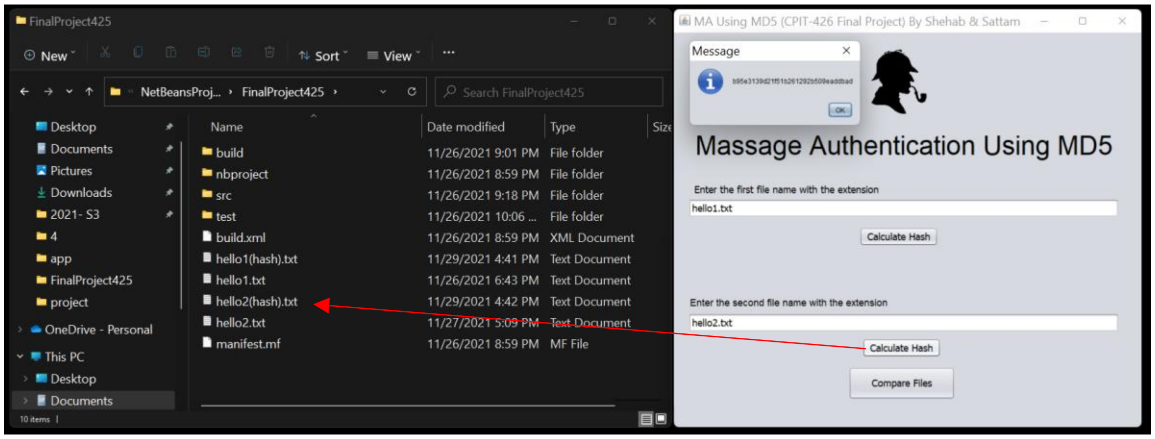
```
5eb63bbbe01eeed093cb22bb8f5acdc3
```

Massage Authentication Using MD5

Enter the first file name with the extension: hello1.txt

Enter the second file name with the extension:

Name	Date modified	Type
build	11/26/2021 9:01 PM	File folder
nbproject	11/26/2021 8:59 PM	File folder
src	11/26/2021 9:18 PM	File folder
test	11/26/2021 10:06 ...	File folder
build.xml	11/26/2021 8:59 PM	XML Document
hello1(hash).txt	11/29/2021 4:41 PM	Text Document
hello1.txt	11/26/2021 6:43 PM	Text Document
hello2.txt	11/27/2021 5:09 PM	Text Document
manifest.mf	11/26/2021 8:59 PM	MF File



MA Using MD5 (CPIT-426 Final Project) By Shehab & Sattam

Message

```
b95e3139d21f51b261292b509eaddbad
```

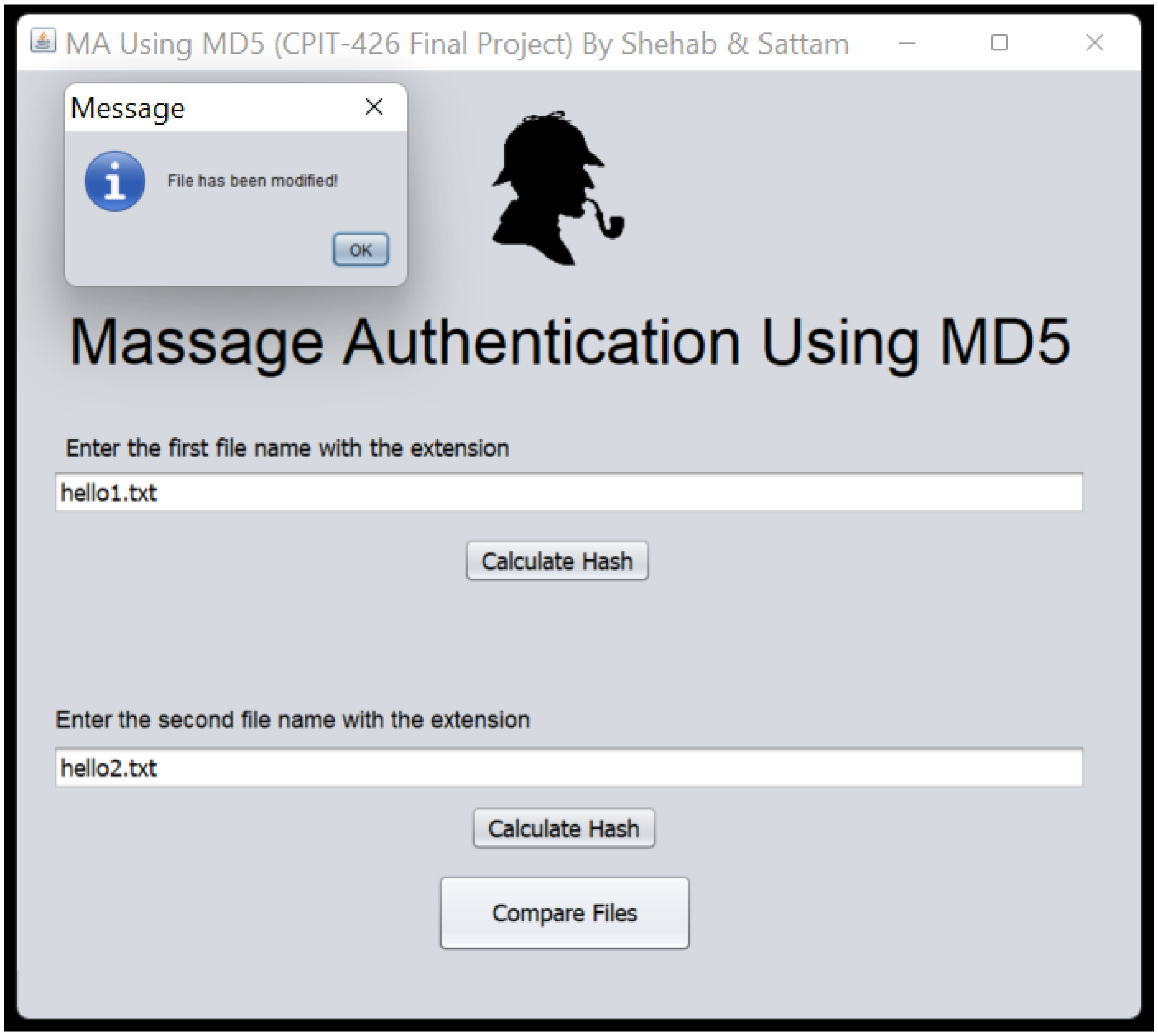
Massage Authentication Using MD5

Enter the first file name with the extension: hello1.txt

Enter the second file name with the extension: hello2.txt

Name	Date modified	Type
build	11/26/2021 9:01 PM	File folder
nbproject	11/26/2021 8:59 PM	File folder
src	11/26/2021 9:18 PM	File folder
test	11/26/2021 10:06 ...	File folder
build.xml	11/26/2021 8:59 PM	XML Document
hello1(hash).txt	11/29/2021 4:41 PM	Text Document
hello1.txt	11/26/2021 6:43 PM	Text Document
hello2(hash).txt	11/29/2021 4:42 PM	Text Document
hello2.txt	11/27/2021 5:09 PM	Text Document
manifest.mf	11/26/2021 8:59 PM	MF File

After clicking (Calculate Hash) a massage will pop up containing the hash and a txt file will be created.


MA Using MD5 (CPIT-426 Final Project) By Shehab & Sattam

Message

File has been modified!

OK

Massage Authentication Using MD5

Enter the first file name with the extension

hello1.txt

Calculate Hash

Enter the second file name with the extension

hello2.txt

Calculate Hash

Compare Files

Since the file have different hash, means the file have been modified.

References

[1] StackOverFlow. https://stackoverflow.com/questions/304268/getting-a-files-md5-checksum-in-java . **Accessed Date:** Thursday - 25/11/2021 **(The site which we took the code from)**

[2] MD5 Hash Generator. https://www.md5hashgenerator.com/ **(The site which helped us to test our resaults)**

[3] YouTube . https://www.youtube.com/watch?v=hy96HssvdtA&t=751s **.Accessed Date:** Thursday - 25/11/2021

[4] StackExchange . https://crypto.stackexchange.com/questions/41827/md5-algorithm-constant . **Accessed Date:** Thursday - 7/12/2021 **(The site which we took the constants from)**

[5] Crypto and Net Security STALLINGS 5ed. Page 343

[6] OnlineInteRviewQuestions . https://www.onlineinterviewquestions.com/what-is-difference-between-md5-and-sha256/ s