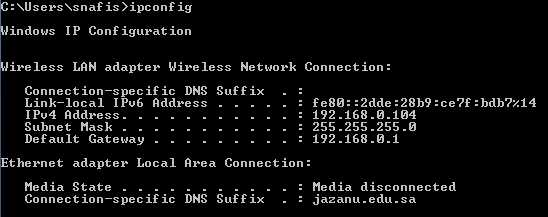
**LAB ACTIVITY NO 1:**

**Some important commands for Data and Network Security**

1. **ipconfig:** Configure IP (Internet Protocol configuration)

It displays all current TCP/IP network configuration values and refreshes Dynamic Host Configuration Protocol (DHCP) and Domain Name System (DNS) settings. Used without parameters, **ipconfig** displays the IP address, subnet mask, and default gateway for all adapters.

* 1. **ipconfig:** Display IP configuration.



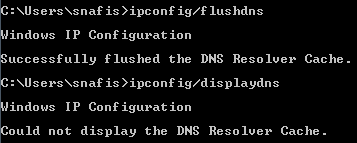
* 1. **ipconfig /all :** Display full configuration information.



* 1. **ipconfig/displaydns:** That command displays your "local" DNS cache that is stored in Windows, this makes browsing faster because it keeps records for any website you have visited before, on your local hard drive, which means the browser does not have to wait for a DNS server out on the internet to resolve the address and pass that information back to your browser.



* 1. **ipconfig/flushdns:** Clean the DNS Resolver cache.



## Ping:

The **ping** command helps to verify IP-level connectivity. When troubleshooting, you can use **ping** to send an ICMP echo request to a target host name or IP address. Use **ping** whenever you need to verify that a host computer can connect to the TCP/IP network and network resources. You can also use **ping** to isolate network hardware problems and incompatible configurations.

Follow this sequence to diagnose network connectivity:

1. Ping the loopback address to verify that TCP/IP is configured correctly on the local computer. **ping 127.0.0.1**
2. Ping the IP address of the local computer to verify that it was added to the network correctly. **ping** *IP\_address\_of\_local\_host*
3. Ping the IP address of the default gateway to verify that the default gateway is functioning and that you can communicate with a local host on the local network.

**ping** *IP\_address\_of\_default\_gateway*

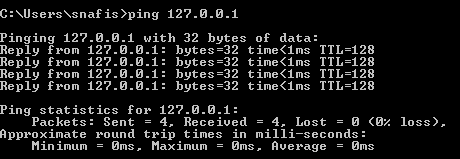
1. Ping the IP address of a remote host to verify that you can communicate through a router.

**ping** *IP\_address\_of\_remote\_host*

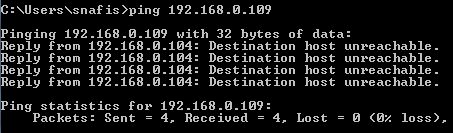
The following table shows some useful **ping** command options.

|  |  |
| --- | --- |
| **Option** | **Use** |
| **-n** *Count* | Determines the number of echo requests to send. The default is 4 requests. |
| **-w** *Timeout* | Enables you to adjust the timeout (in milliseconds). The default is 4,000 (a 4-  second timeout). |
| **-l** *Size* | Enables you to adjust the size of the ping packet. The default size is 32 bytes. |
| **-f** | Sets the Do Not Fragment bit on the ping packet. By default, the ping packet  allows fragmentation. |
| **/?** | Provides command Help. |

**If connected or reachable:**



**If not connected or unreachable:**



## tracert

The tracert command is used to visually see a network packet being sent and received and the amount of hops required for that packet to get to its destination.

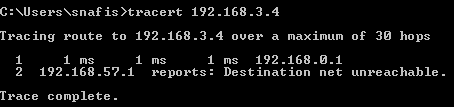
#### Tracert syntax

tracert [-d] [-h maximum\_hops] [-j host-list] [-w timeout] [-R] [-S srcaddr] [-4] [-6] target\_name

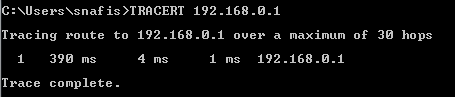
#### Options:

|  |  |
| --- | --- |
| -d | Do not resolve addresses to hostnames. |
| -h maximum\_hops | Maximum number of hops to search for target.  Default is 30 hops. |
| -j host-list | Loose source route along host-list (IPv4-only). |
| -w timeout | Wait timeout milliseconds for each reply. |
| -R | Trace round-trip path (IPv6-only). |
| -S  srcaddr | Source address to use (IPv6-only). |
| -4 | Force using IPv4. |
| -6 | Force using IPv6. |

**If not connected or unreachable:**



**If connected or reachable:**



## nbtstat

MS-DOS utility that displays protocol statistics and current TCP/IP connections using NBT (NetBIOS over TCP/IP), which allow the user to troubleshoot NetBIOS name resolution issues. Normally, name resolution is done when NetBIOS over

TCP/IP is functioning correctly. It does this through local cache lookup, WINS or DNS server query or through LMHOSTS or Hosts lookup.

#### nbtstat syntax

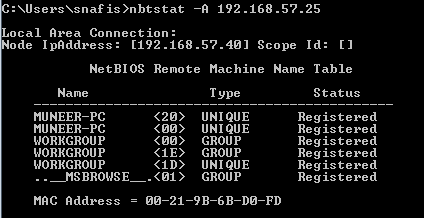
nbtstat [ [-a RemoteName] [-A IP address] [-c] [-n] [-r] [-R] [-RR] [-s] [-S] [interval] ]

|  |  |
| --- | --- |
| -a | (adapter status) Lists the remote machine's name table given its name |
| -A | (Adapter status) Lists the remote machine's name table given its IP address. |
| -c | (cache) Lists NBT's cache of remote [machine] names and their IP addresses |
| -n | (names) Lists local NetBIOS names. |
| -r | (resolved) Lists names resolved by broadcast and via WINS |
| -R | (Reload) Purges and reloads the remote cache name table |
| -S | (Sessions) Lists sessions table with the destination IP addresses |
| -s | (sessions) Lists sessions table converting destination IP addresses to  computer NETBIOS names. |
| -RR | (ReleaseRefresh) Sends Name Release packets to WINs and then, starts  Refresh |
| RemoteName | Remote host machine name. |
| IP address | Dotted decimal representation of the IP address. |
| interval | Redisplays selected statistics, pausing interval seconds between each  display. Press Ctrl+C to stop redisplaying statistics. |

#### nbtstat examples

nbtstat -A 204.224.150.3

The above command would run nbtstat on 204.224.150.3, a remote IP address.



## telnet

It enables a user to telnet to another computer from the command prompt.

#### Telnet syntax

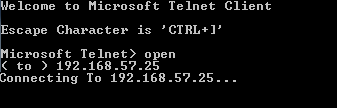
telnet [host [port]]

|  |  |
| --- | --- |
| host | specifies the hostname or IP address of the remote computer. |
| port | Specifies the port number or service name. |

Commands available through the actual telnet program:

|  |  |
| --- | --- |
| close | close current connection |
| display | display operating parameters |
| open | connect to a site |
| quit | exit telnet |
| status | print status information |
| ?/help | print help information |

#### Examples

**telnet 192.168.57.25**

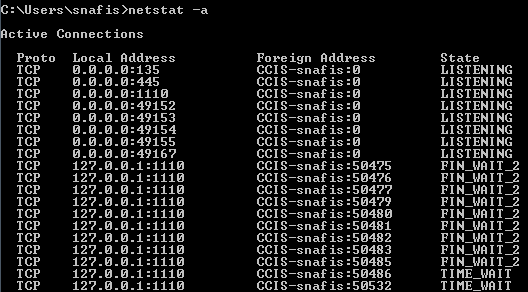
1. **netstat**

Netstat is a useful tool for checking network and Internet connections. Some useful applications for the average PC user are considered, including checking for malware connections.

#### Syntax and switches Netstat syntax

netstat [-a] [-b] [-e] [-f] [-n] [-o] [-p proto] [-r] [-s] [-t] [-v] [interval]

|  |  |
| --- | --- |
| **Switch** | **Description** |
| -a | Displays all connections and listening ports |
| -b | Displays the executable involved in creating each connection or listening  port. (Added in XP SP2.) |
| -e | Displays Ethernet statistics |



## tasklist

This tool displays a list of currently running processes on either a local or remote machine.

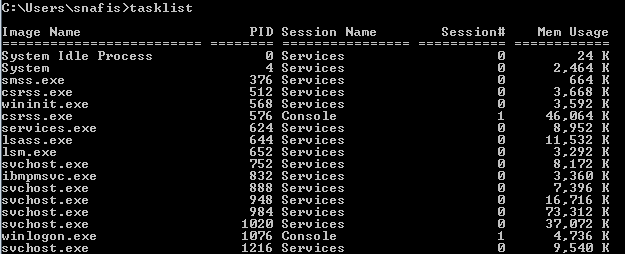
#### Tasklist syntax

TASKLIST [/S system [/U username [/P [password]]]] [/M [module] | /SVC | /V] [/FI filter] [/FO format] [/NH]

#### Filters

|  |  |
| --- | --- |
| /S system | Specifies the remote system to connect to. |
| /U [domain\]user | Specifies the user context under which the command  should execute. |
| /P [password] | Specifies the password for the given user context. Prompts  for input if omitted. |
| /M [module] | Lists all tasks currently using the given exe/dll name. If the  module name is not specified all loaded modules are displayed. |
| /SVC | Displays services hosted in each process. |

**Example**



1. **getmac**

It returns the media access control (MAC) address and list of network protocols associated with each address for all network cards in each computer, either locally or across a network.

Syntax

**getmac**[.**exe**] [**/s** *Computer* [**/u** *Domain\User* [**/p** *Password*]]] [**/fo** {**TABLE**|**LIST**|**CS V**}] [**/nh**] [**/v**]

#### Parameters

**/s *Computer* :** Specifies the name or IP address of a remote computer (do not use backslashes). The default is the local computer.

**/u *Domain* \ *User* :** Runs the command with the account permissions of the user specified by *User* or *Domain*\*User*. The default is the permissions of the current logged on user on the computer issuing the command.

**/p *Password* :** Specifies the password of the user account that is specified in the **/u** parameter.

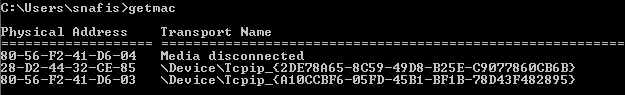
**/fo { TABLE | LIST | CSV } :** Specifies the format to use for the query output. Valid values are **TABLE**, **LIST**, and **CSV**. The default format for output is**TABLE**.

**/nh :** Suppresses column header in output. Valid when the **/fo** parameter is set to **TABLE** or **CSV**.

**/v :** Specifies that the output display verbose information.

**/? :** Displays help at the command prompt.

#### Examples



1. **hostname**

Display the hostname of the machine the command is being run on.



## pathping

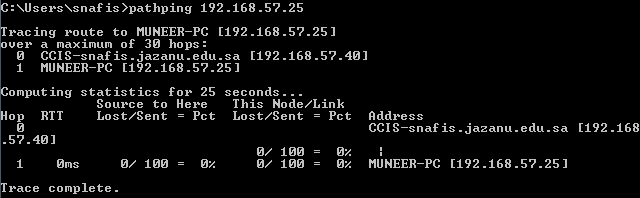
Similar to the tracert command, pathping provides users with the ability of locating spots that have network latency and network loss.

#### Usage:

pathping [-g host-list] [-h maximum\_hops] [-i address] [-n] [-p period] [-q num\_queries] [-w timeout] [-P] [-R] [-T] [-4] [-6] target\_name

#### Options:

|  |  |
| --- | --- |
| -g host-list | Loose source route along host-list. |
| -h maximum\_hops | Maximum number of hops to search for target. |
| -i address | Use the specified source address. |
| -n | Do not resolve addresses to hostnames. |
| -p period | Wait period milliseconds between pings. |
| -q num\_queries | Number of queries per hop. |
| -w timeout | Wait timeout milliseconds for each reply. |
| -P | Test for RSVP PATH connectivity. |
| -R | Test if each hop is RSVP aware. |
| -T | Test connectivity to each hop with Layer-2 priority tags. |
| -4 | Force using IPv4. |
| -6 | Force using IPv6. |



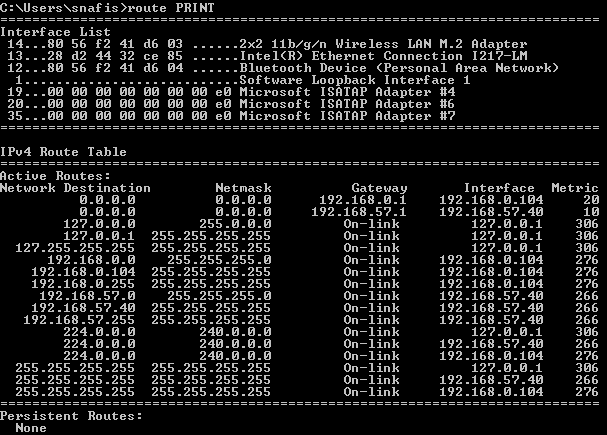
1. **route**

Command to show or manually configure the routes in the routing table.

#### Syntax

ROUTE [-f] [-p] [-4|-6] command [destination] [MASK netmask] [gateway] [METRIC metric] [IF interface]

|  |  |
| --- | --- |
| -f | Clears the routing tables of all gateway entries. If this is used in  conjunction with one of the commands, the tables are cleared prior to running the command. |
| -p | When used with the ADD command, makes a route persistent across boots of the system. By default, routes are not preserved when the system is restarted. When used with the PRINT command, displays the list of registered persistent routes. Ignored for all other commands, which always affect the appropriate persistent routes. This option is not  supported Windows'95. Command |
| -4 | Force using [IPv4.](http://www.computerhope.com/jargon/i/ipv4.htm) |
| -6 | Force using [IPv6.](http://www.computerhope.com/jargon/i/ipv6.htm) |
| command | One of these:  PRINT Prints a route ADD Adds a route DELETE Deletes a route  CHANGE Modifies an existing route destination |
| destination | Specifies the host. |
| MASK | Specifies that the next parameter is the 'netmask' value. |
| netmask | Specifies a subnet mask value for this route entry. If not specified, it  defaults to 255.255.255.255. |
| gateway | Specifies gateway. |
| interface | the interface number for the specified route. |
| METRIC | Specifies the metric, ie. cost for the destination. |



## fc

FC, or file compare, is used to compare two files against each other. Once completed, fc returns lines that differ between the two files. If no lines differ, you will receive a message indicating no differences encountered.

#### fc syntax

Compares two files or sets of files and displays the differences between them.

FC [/A] [/C] [/L] [/LBn] [/N] [/T] [/W] [/nnnn] [drive1:][path1]filename1 [drive2:][path2]filename2

FC /B [drive1:][path1]filename1 [drive2:][path2]filename2

|  |  |
| --- | --- |
| /A | Displays only first and last lines for each set of differences. |
| /B | Performs a binary comparison. |
| /C | Disregards the case of letters. |
| /L | Compares files as ASCII text. |
| /LBn | Sets the maximum consecutive mismatches to the specified  number of lines. |
| /N | Displays the line numbers on an ASCII comparison. |
| /T | Does not expand tabs to spaces. |

|  |  |
| --- | --- |
| /W | Compresses white space (tabs and spaces) for comparison. |
| /nnnn | Specifies the number of consecutive lines that must match  after a mismatch. |
| [drive1:][path1]filename1 | Specifies the first file or set of files to compare. |
| [drive2:][path2]filename2 | Specifies the second file or set of files to compare. |

#### fc examples

fc autoexec.bat config.sys

## sfc

Scan System Files for Problems. Short for **System File Checker, SFC** is a command that scans and replaces any Microsoft Windows file on the computer and replaces any changed file with the correct version. This is a great command to run when you are running into an issue that is difficult to troubleshoot.

#### syntax

SFC [/SCANNOW] [/VERIFYONLY] [/SCANFILE=<file>] [/VERIFYFILE=<file>]

[/OFFWINDIR=<offline windows directory> /OFFBOOTDIR=<offline boot directory> ]

|  |  |
| --- | --- |
| /SCANNOW | Scans integrity of all protected system files and repairs files  with problems when possible. |
| /VERIFYONLY | Scans integrity of all protected system files. No repair operation  is performed. |
| /SCANFILE | Scans integrity of the referenced file, repairs file if problems are  identified. Specify full path <file>. |
| /VERIFYFILE | Verifies the integrity of the file with full path <file>. No re pair  operation is performed. |
| /OFFBOOTDIR | For offline repair specify the location of the offline boot  directory. |
| /OFFWINDIR | For offline repair specify the location of the offline Windows  directory. |

## recimg

Create custom recovery images. It is one of hidden feature of creating custom recovery images. Using this command, you can create your custom recovery images. Using this feature, you can remove default bloatware and also enables you to add your favourite third party programs to recovery images to your PC easily.

## cipher

Displays or alters the encryption of directories [files] on NTFS partitions.

#### Syntax

CIPHER [/E | /D | /C] [/S:directory] [/B] [/H] [pathname [...]] CIPHER /K [/ECC:256|384|521]

CIPHER /R:filename [/SMARTCARD] [/ECC:256|384|521] CIPHER /U [/N]

CIPHER /W:directory CIPHER /X[:efsfile] [filename] CIPHER /Y

CIPHER /ADDUSER [/CERTHASH:hash | /CERTFILE:filename | /USER:username] [/S:directory] [/B] [/H] [pathname [...]]

CIPHER /FLUSHCACHE [/SERVER:servername]

CIPHER /REMOVEUSER /CERTHASH:hash [/S:directory] [/B] [/H] [pathname [...]]

CIPHER /REKEY [pathname [...]]

|  |  |
| --- | --- |
| /B | Abort if an error is encountered. By default, CIPHER continues  executing even if errors are encountered. |
| /C | Displays information on the encrypted file. |
| /D | Decrypts the specified directories. Directories will be marked so  that files added afterward will not be encrypted. |
| /E | /E Encrypts the specified files or directories. Directories will be marked so that files added afterward will be encrypted. The encrypted file could become decrypted when it is modified if the parent directory is not encrypted. It is recommended that you  encrypt the file and the parent directory. |
| /H | Displays files with the hidden or system attributes. These files are  omitted by default. |
| /K | Create new file encryption key for the user running CIPHER. If this option is chosen, all the other options will be ignored.  Note: By default, /K creates a certificate and key that conform to current group policy. If ECC is specified, a self-signed certificate will be created with the supplied key size. |
| /N | This option only works with /U and prevents keys being updated.  This is used to find all the encrypted files on the local drives. |
| /R | /R Generates an EFS recovery key and certificate, then writes them  to a .PFX file (containing certificate and private key) and a .CER |

|  |  |
| --- | --- |
|  | file (containing only the certificate). An administrator may add the contents of the .CER to the EFS recovery policy to create the recovery key for users, and import the .PFX to recover individual files. If SMARTCARD is specified, then writes the recovery key and certificate to a smart card. A .CER file is generated (containing only the certificate). No .PFX file is generated.  Note: By default, /R creates an 2048-bit RSA recovery key and certificate. If ECC is specified, it must be followed by a key size of 256, 384, or 521. |
| /S | Performs the specified operation on directories in the given  directory and all subdirectories. |
| /U | Tries to touch all the encrypted files on local drives. The /U switch update user's file encryption key or recovery keys to the current ones if they are changed. This option does not work with other  options except /N. |
| /W | Removes data from available unused disk space on the entire volume. If this option is chosen, all other options are ignored. The directory specified can be anywhere in a local volume. If it is a mount point or points to a directory in another volume, the data on  that volume will be removed. |
| /X | Backup EFS certificate and keys into file filename. If efsfile is provided, the current user's certificate(s) used to encrypt the file will be backed up. Otherwise, the user's current EFS certificate and keys  will be backed up. |
| /Y | Displays your current EFS certificate thumbprint on the local PC. |
| /ADDUS ER | Adds a user to the specified encrypted file(s). If CERTHASH is provided, cipher will search for a certificate with this SHA1 hash. If CERTFILE is provided, cipher will extract the certificate from the  file. If USER is provided, cipher will try to locate the user's certificate in Active Directory Domain Services. |
| /FLUSHC ACHE | Clears the calling user's EFS key cache on the specified server. If  servername is not provided, cipher clears the user's key cache on the local machine. |
| /REKEY | Updates the specified encrypted file(s) to use the configured EFS  current key. |
| /REMOV  EUSER | Removes a user from the specified file(s). CERTHASH must be the  SHA1 hash of the certificate to remove. |
| directory | A directory path. |
| filename | A filename without extensions. |
| pathname | Specifies a pattern, file or directory. |
| efsfile | An encrypted file path. |

## arp

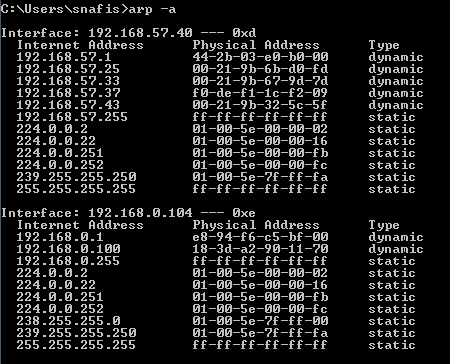
Displays, adds, and removes arp information from network devices.

ARP -s inet\_addr eth\_adr [if\_addr] ARP -d inet\_addr [if\_addr]

ARP -a [inet\_addr] [-N if\_addr]

|  |  |
| --- | --- |
| -a | Displays current ARP entries by interrogating the current protocol data. If inet\_addr is specified, the IP and physical addresses for only the specified computer are displayed. If more than one network interface uses ARP, entries  for each ARP table are displayed. |
| -g | Same as –a |
| inet\_addr | Specifies an Internet address. |
| -N if addr | Displays the ARP entries for the network interface specified by if\_addr. |
| -d | Deletes the host specified by inet\_addr. |
| -s | Adds the host and associates the Internet address inet\_addr with the physical  address eth\_addr. The physical address is given as 6 hexadecimal bytes separated by hyphens. The entry is permanent. |
| eth\_addr | Specifies a physical address |
| if\_addr | If present, this specifies the Internet address of the interface whose address translation table should be modified. If not present, the first applicable  interface will be used. |

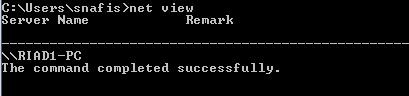
**ARP examples :** arp –a



## net view

It displays a list of computers in a specified workgroup or the shared resources available on a specified computer.

Syntax:

[\\computername [/CACHE] | /DOMAIN[:domainname]] NET VIEW /NETWORK:NW [\\computername]

**LAB (TASK) 1**

**ipconfig:**

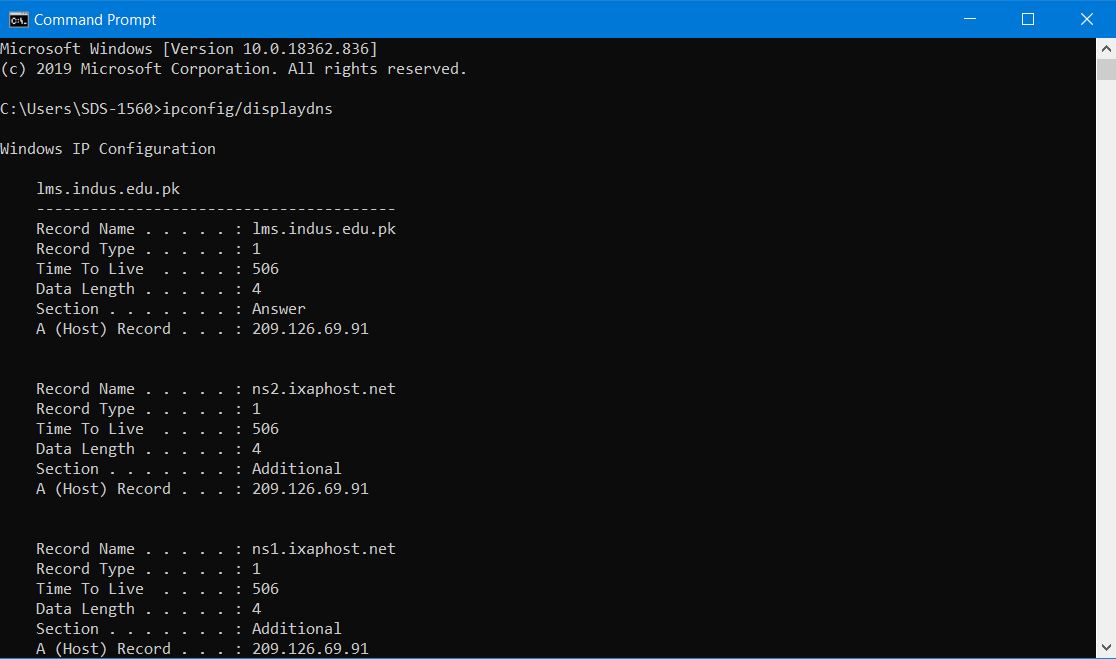
* 1. **ipconfig:** Display IP configuration.



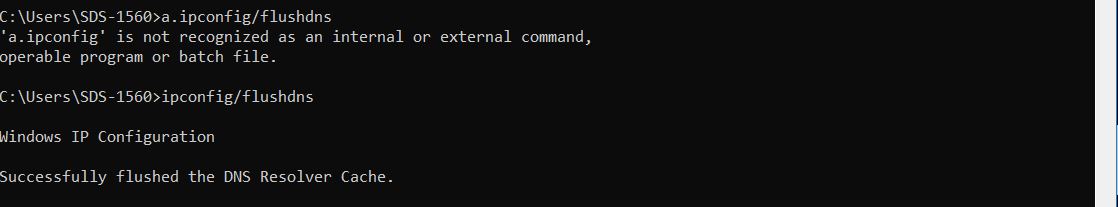
* 1. **ipconfig /all :** Display full configuration information.



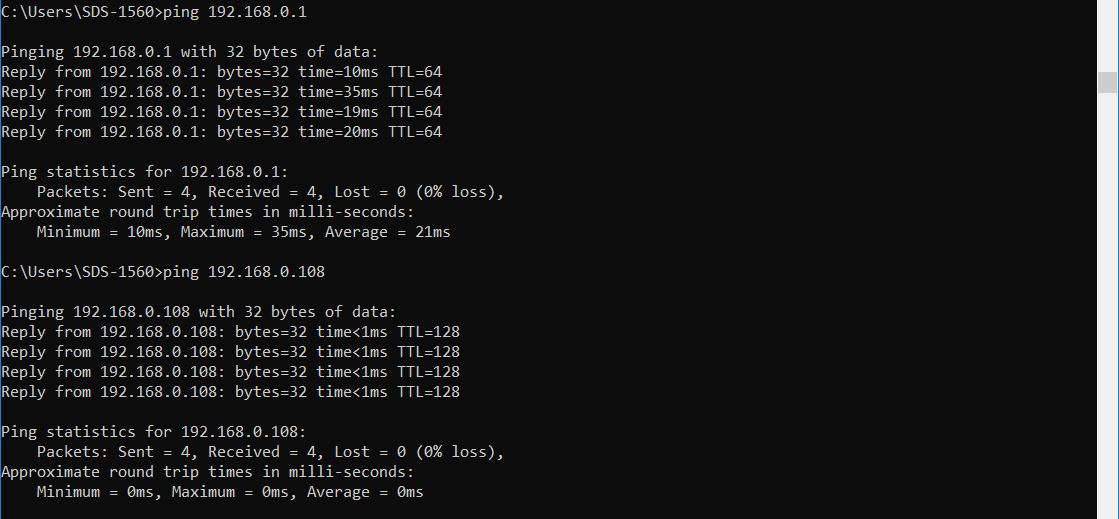
**c. ipconfig/displaydns:**

****

**d. ipconfig/flushdns:** Clean the DNS Resolver cache.



## Ping:



## tracertl1.6.JPG

## nbtstat

## l1.7.JPG

## telnet

## l1.9.JPG

## netstat

## l1.10.JPG

## tasklist

## l1.11.JPG

## getmac , hostname , pathping

## l1.12.JPG

## route

## l1.13.JPG

## fc, sfc

## l1.14.JPG

## recimg, cipher

## l1.15.JPG

## arp

## l1.16.JPG

**LAB ACTIVITY NO 2:**

**OBJECT: LEARN TO INSTALL WINE / VIRTUAL BOX OR ANY OTHER EQUIVALENT SOFTWARE ON THE HOST OS.**

Virtualization is the process of emulating hardware inside a virtual machine.

Virtualization can include the following:

1. Application Virtual Machines.
2. Mainframe Virtual Machines.
3. Parallel Virtual Machines.
4. Operating System Virtual Machines.

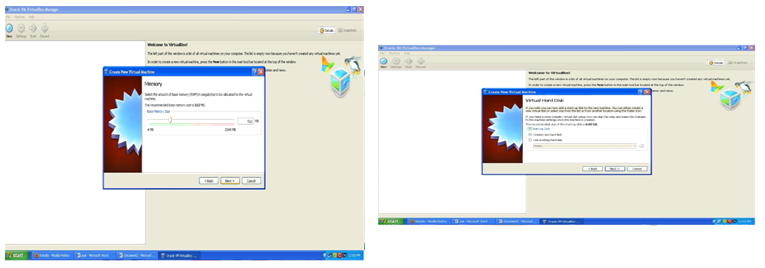
Download the latest version of Sun Virtual Box from the website.

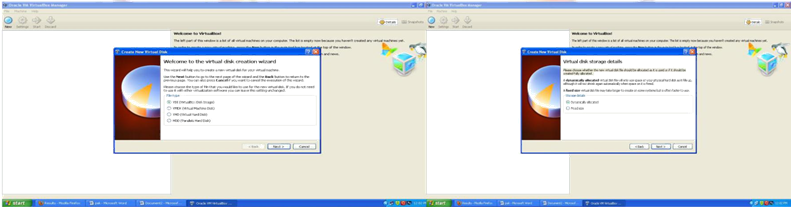
The installation steps are

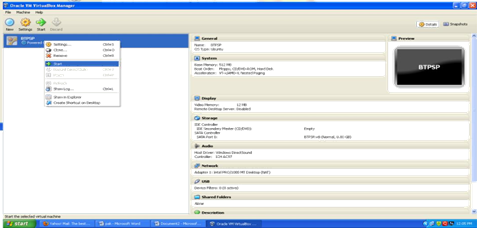
1. Double Click on Virtual Box Executable File.
2. Welcome to the Oracle VM appears [Setup Wizard] Click Next.
3. Custom Setup Screen with all the features of VM appears Click Next.
4. Options to create shortcut on Desktop and Quick launch bar Click Next.
5. Ready to Install Click on Install button.
6. Click Finish Start Oracle VM Virtual Box.

And now install any kind of Operating Systems as Guest Operating System.

The Configuration of Guest OS is shown below.





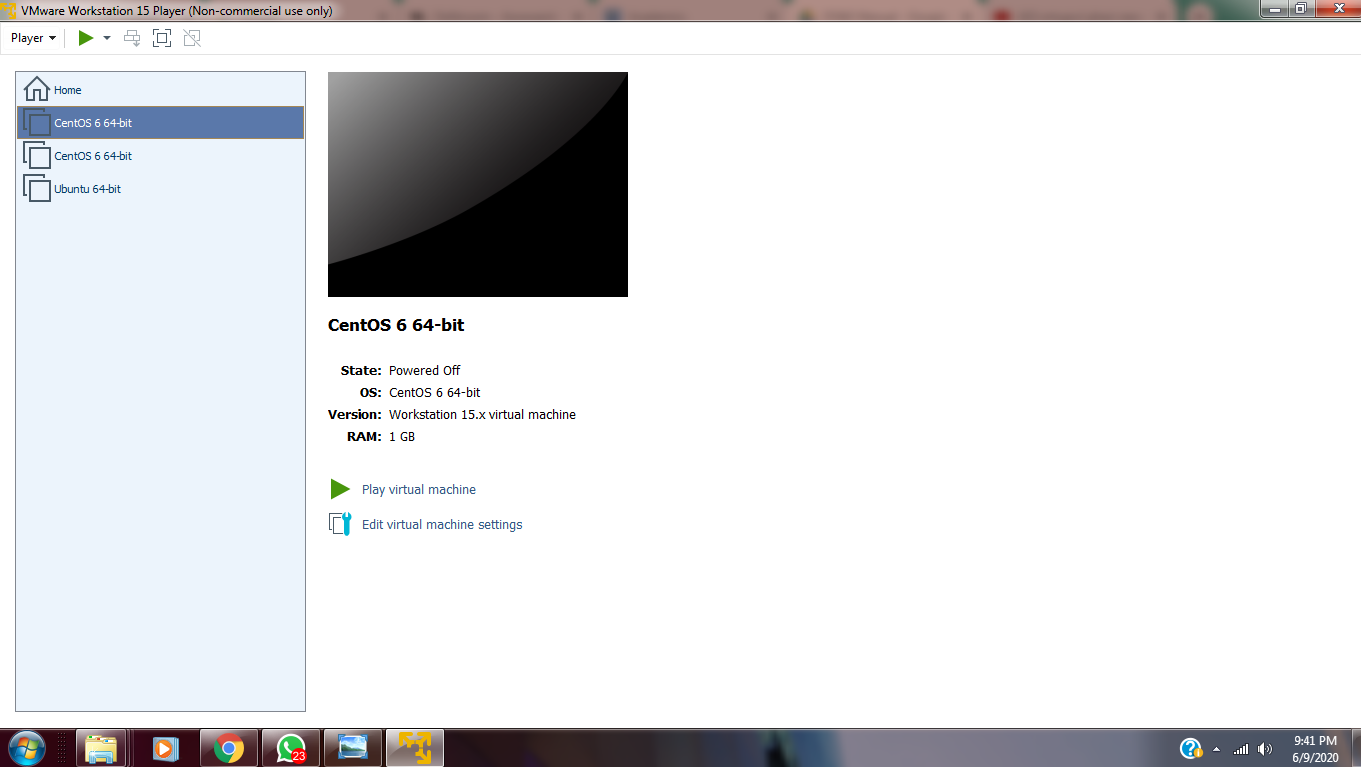


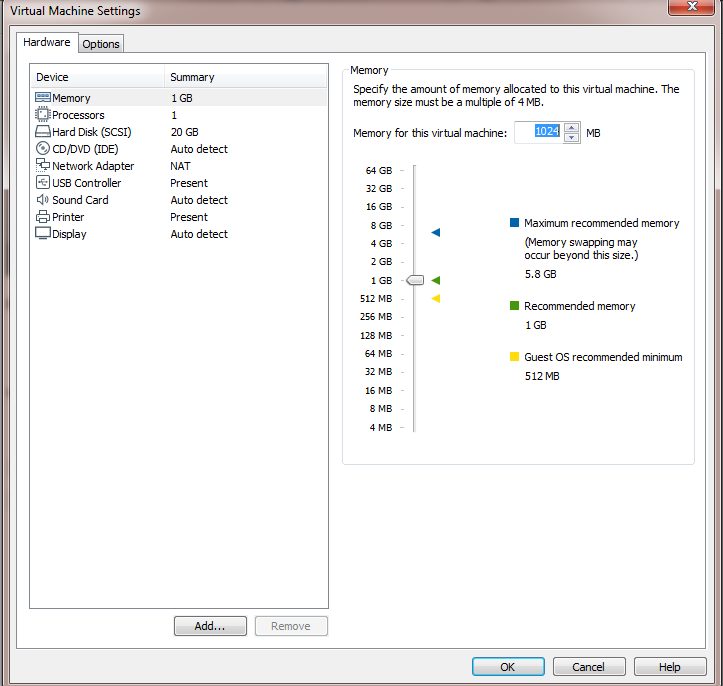
Lab Task:

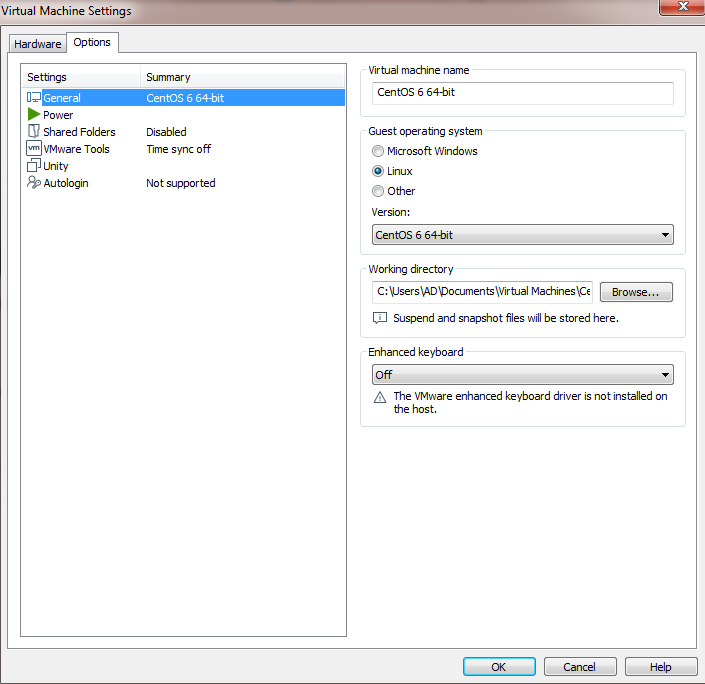
1. Install VM and configure OS.

**LAB (TASK) 2**

**Q Install VM & Configure OS?**

**Output:**

****

****

**LAB ACTIVITY NO 3:**

**OBJECT: PERFORME AN EXPERIMENT TO DEMONSTRATE HOW TO SNIFF FOR ROUTER TRAFFIC BY USING THE TOOL WIRESHARK.**

A packet sniffer, sometimes referred to as a network monitor or network analyzer, can be used by a network or system administrator to monitor and troubleshoot network traffic. Using the information captured by the packet sniffer an administrator can identify erroneous packets and use the data to pinpoint bottlenecks and help maintain efficient network data transmission.

In its simple form a packet sniffer simply captures all of the packets of data that pass through a given network interface. By placing a packet sniffer on a network in promiscuous mode, a

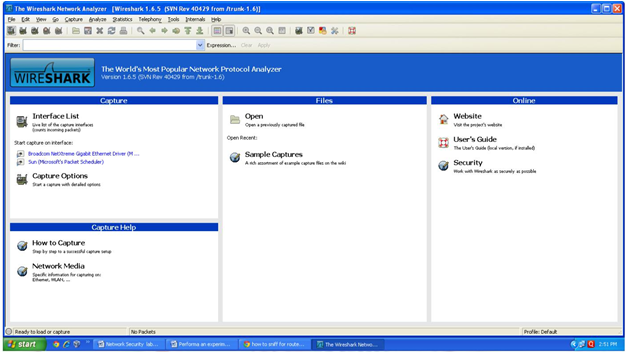
Malicious intruder can capture and analyze all of the network traffic.

Wireshark is a network packet analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible.

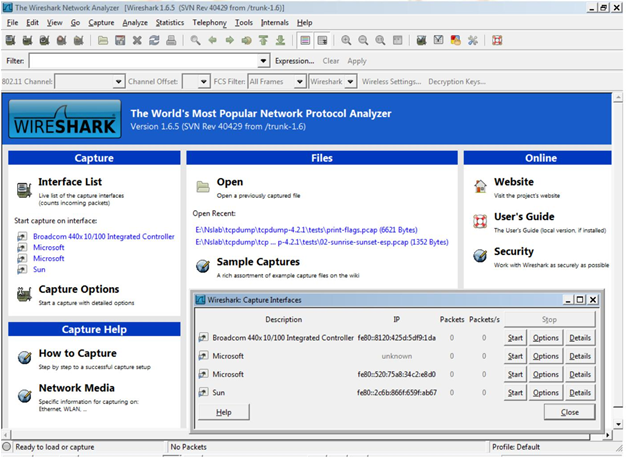
Download and install wireshark network analyzer.

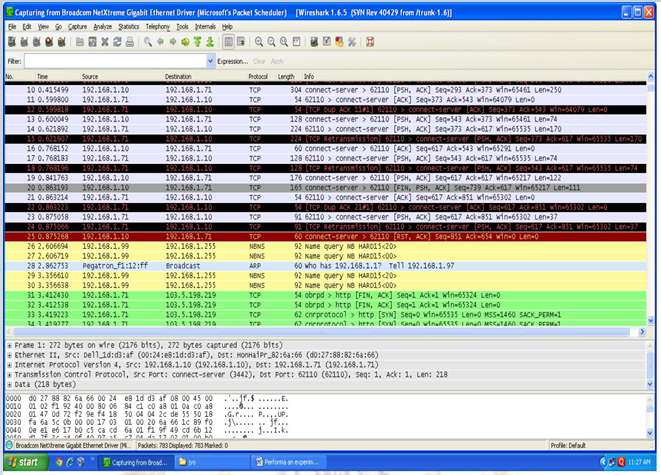
Steps to capture traffic:

1. Open Wireshark network analyzer.



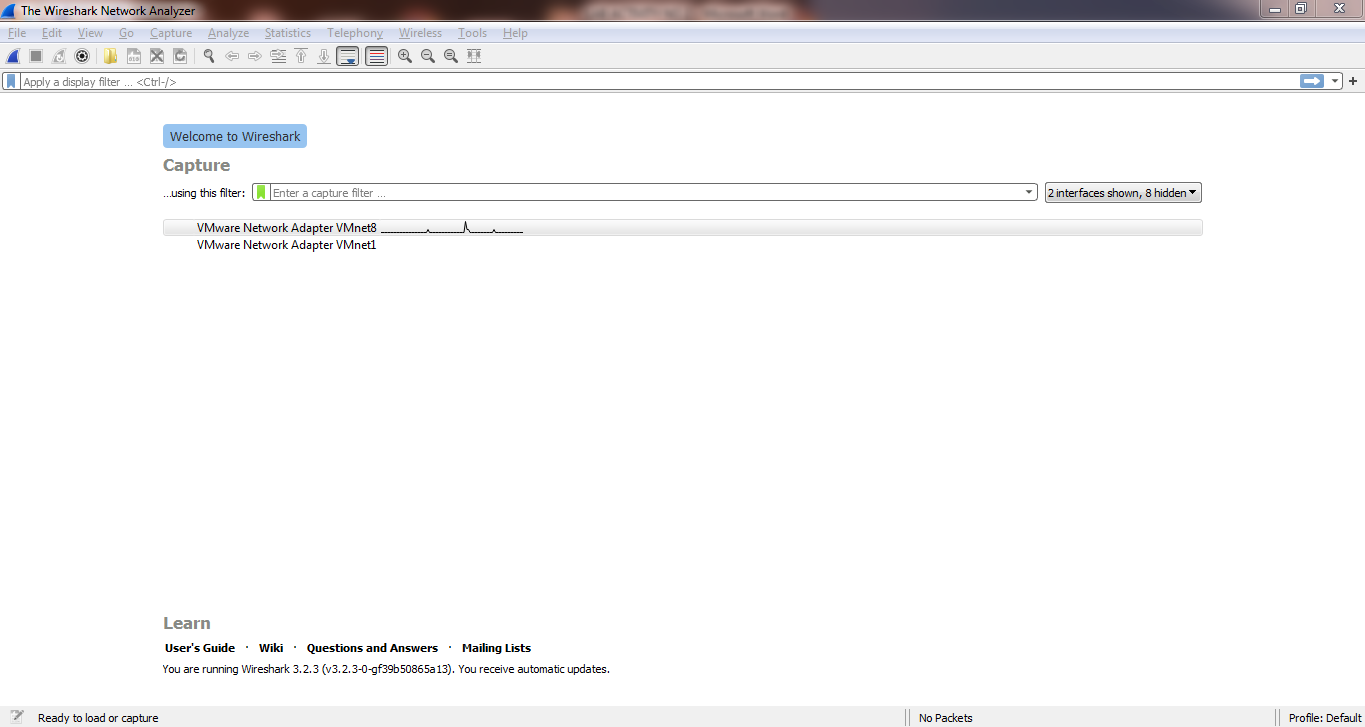
1. Select interface: Goto capture option in menu bar and select interface

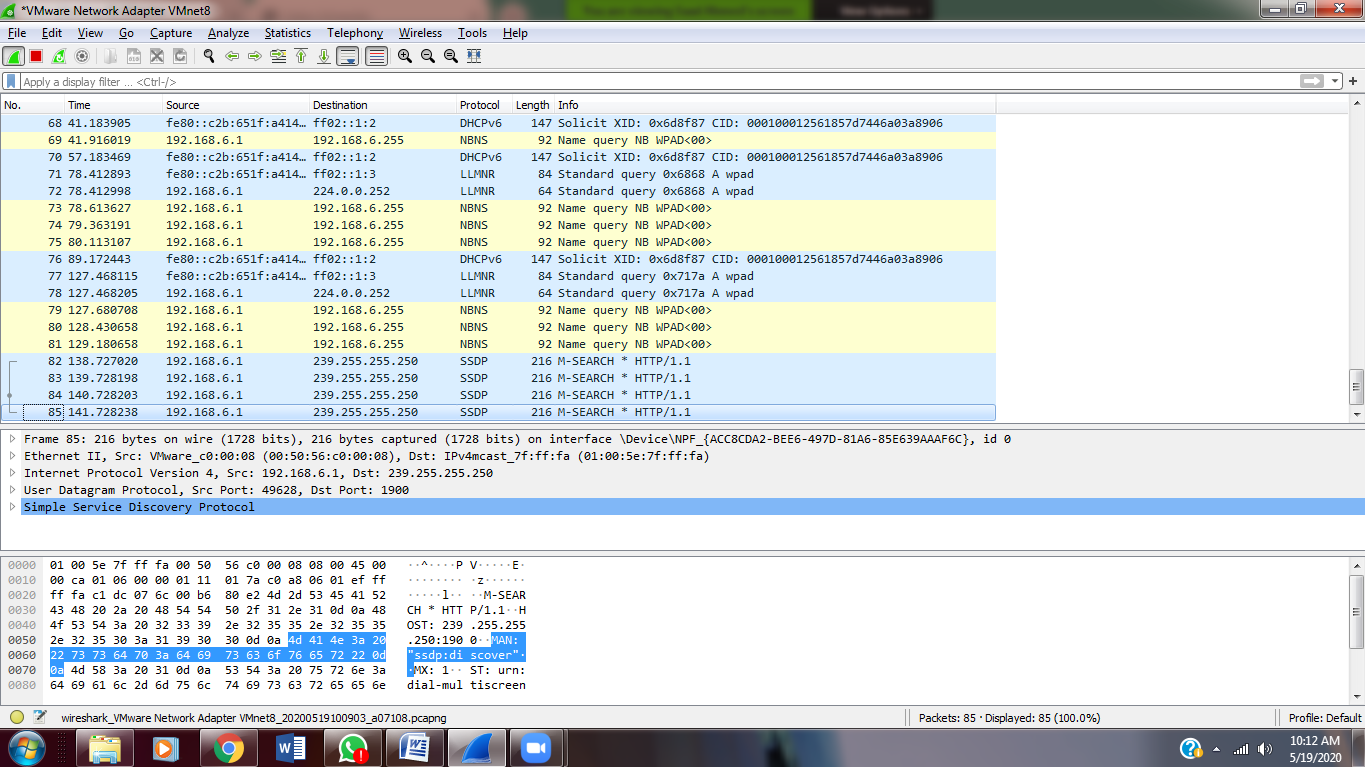




**LAB (TASK) 3**

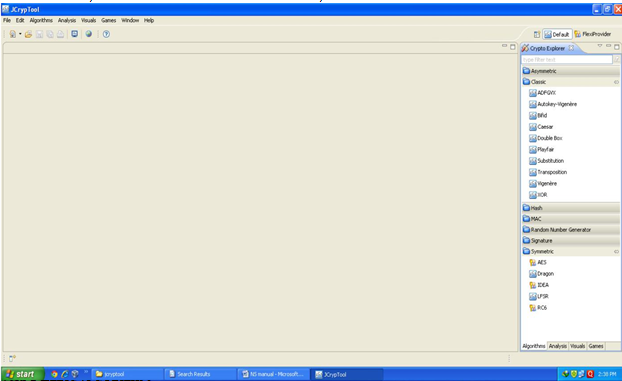
**TASK:**





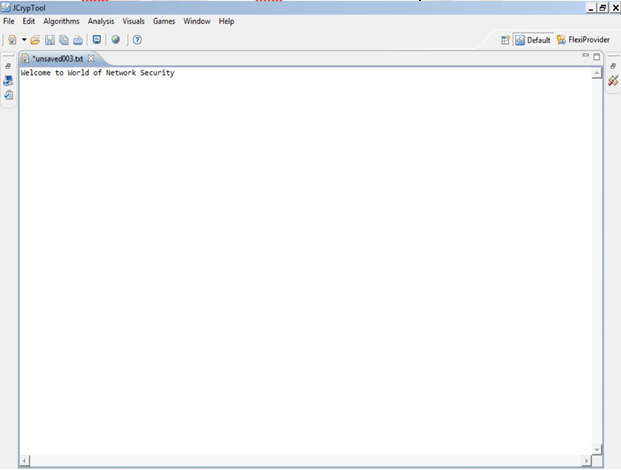
**LAB ACTIVITY NO 4:**

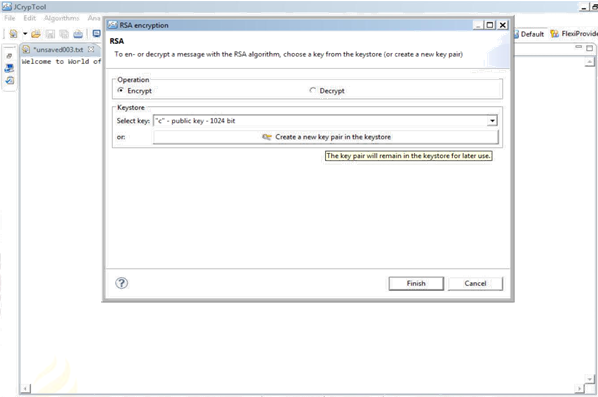
**OBJECT: INSTALL JCRYPT TOOL (OR ANY OTHER EQUIVALENT) AND DEMONSTRATE ASYMMETRIC, SYMMETRIC CRYPTO ALGORITHM, HASH AND DIGITAL/PKI SIGNATURES**



**ASYMMETRIC ALGORITHM**

1. Download Jcrypt tool from Cryptool Website and Install
2. Open Jcrypt Software and Click on NEW text editior, type the text information into it.



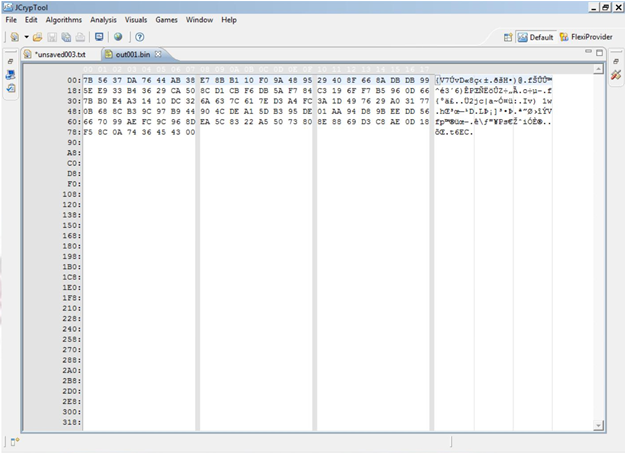


1. Click on the Algorithm menu bar and Select Asymmetric algorithm RSA for encryption. o Click create a New KeyPair and type in the contact name[xxxxx] and enter the password

and confirm password, then Click finish again.



1. Now you can see RSA output bin file is generated.



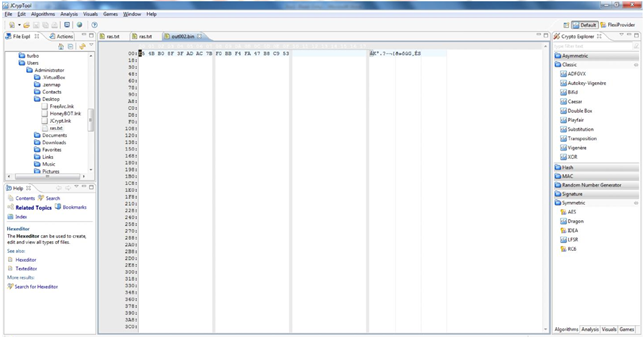
1. The same output bin file to decrypt select RSA Algorithm and Click on Decrypt, Select keyname you have declared earlier and Click Finish.
2. Enter the password to Decrypt and see the output with original Decrypted text on the Screen.

**SYMMETRIC ALGORITHM**

* Click on Algorithm Menu bar Select SymmetricAES and Click on it.
* Click on create a new key, type contact name and enter the password and confirm, Click finish Click finish again.
* Enter the password to open the output file.
* To Decrypt Select AlgorithmsSymmetricSelect the key which you have created and Click Finish.
* Enter the password and see the result in output bin file with hexadecimal values and plain text

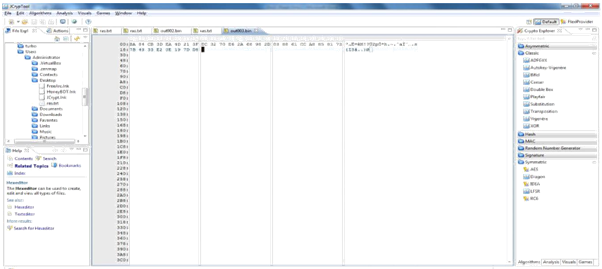
**HASH GENERATION**

* Click on Algorithms, Click on Hash - Select MD5 - Click Finish.
* Now view the output bin file HASH generated.
* Practice using SHA and SHA3 and verify the result on the screen



**DIGITAL SIGNATURE**

* Click on algorithms, Click on Signature, Select DSA and Click on it.
* Select sign operation and Click on create a new key.
* Enter the password and save the file and Click finish.
* To verify Click on Algorithm, Click on Signature and Click DSA.
* Select verify operation, Click open and type the password and Click finish.
* The Signature file is opened and verified.



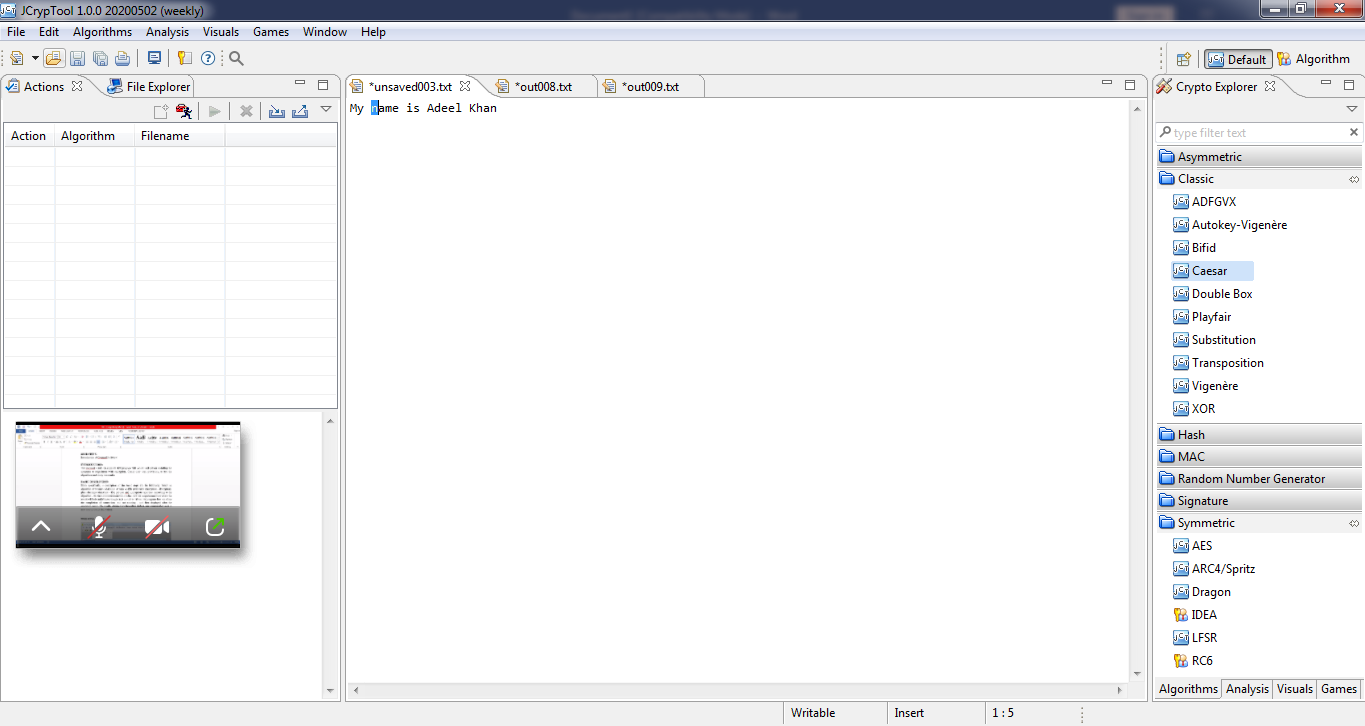
**SHA1 RANDOM PASSWORD GENERATOR**

* Click Algorithm- Click Random Number Generator- Click SHA1
* Type the output size[default 128 bits] or your choice.
* Select filter output binary or output as character or output as numbers and Click Finish.The output bin SHA1 file is generated and displayed on the screen.
* Practice with various size and filter to binary output or output as character or output asnumber

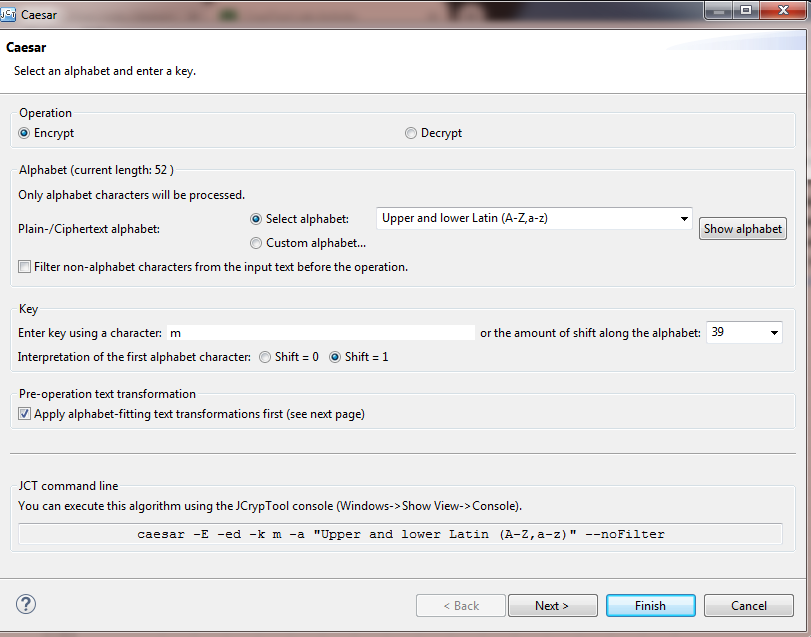
**LAB (TASK) 4**

1. **Caesar Cipher :**

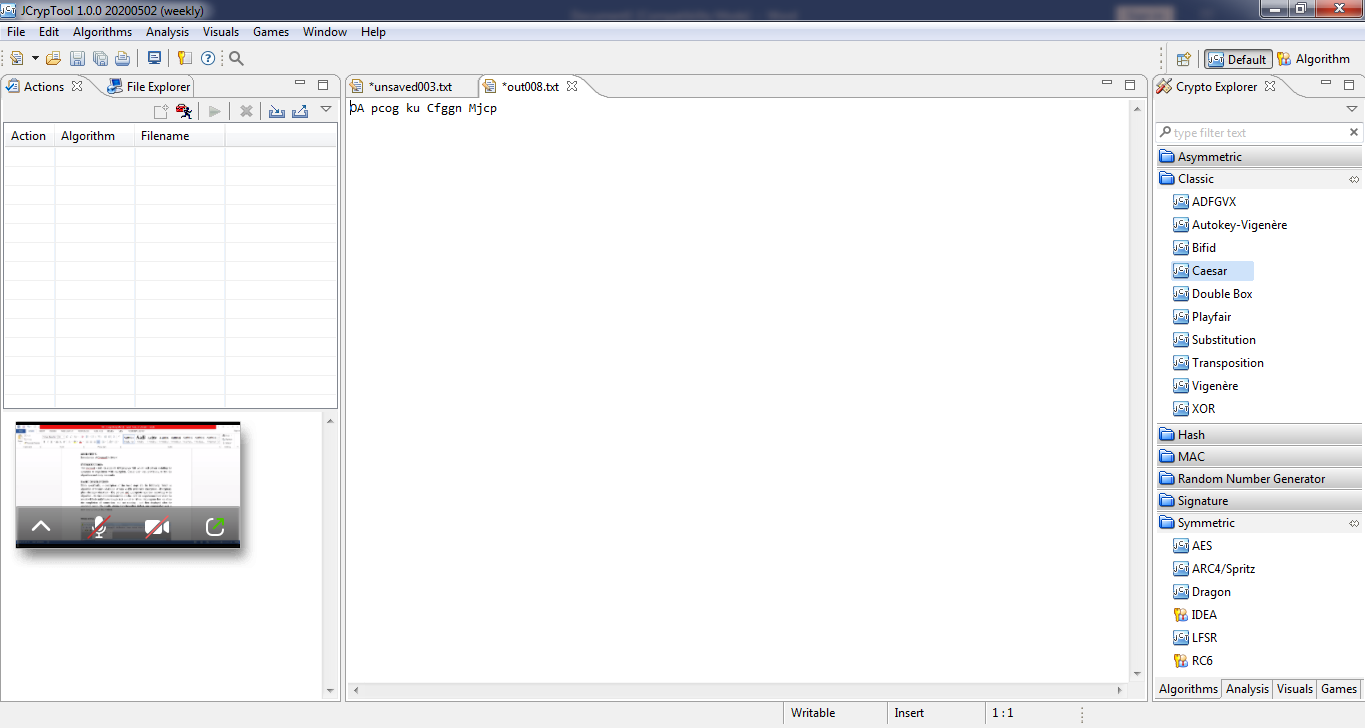
**Text:**

****

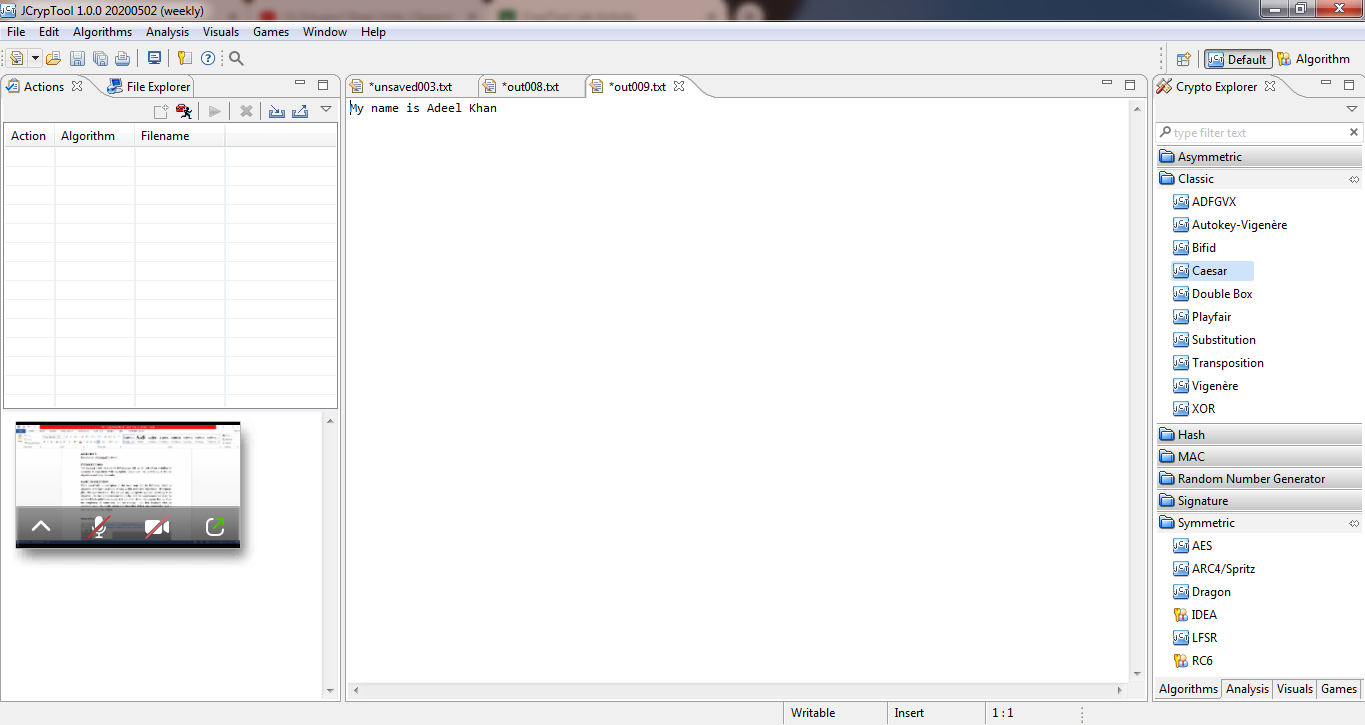
**Key:**

****

**Encryption :**

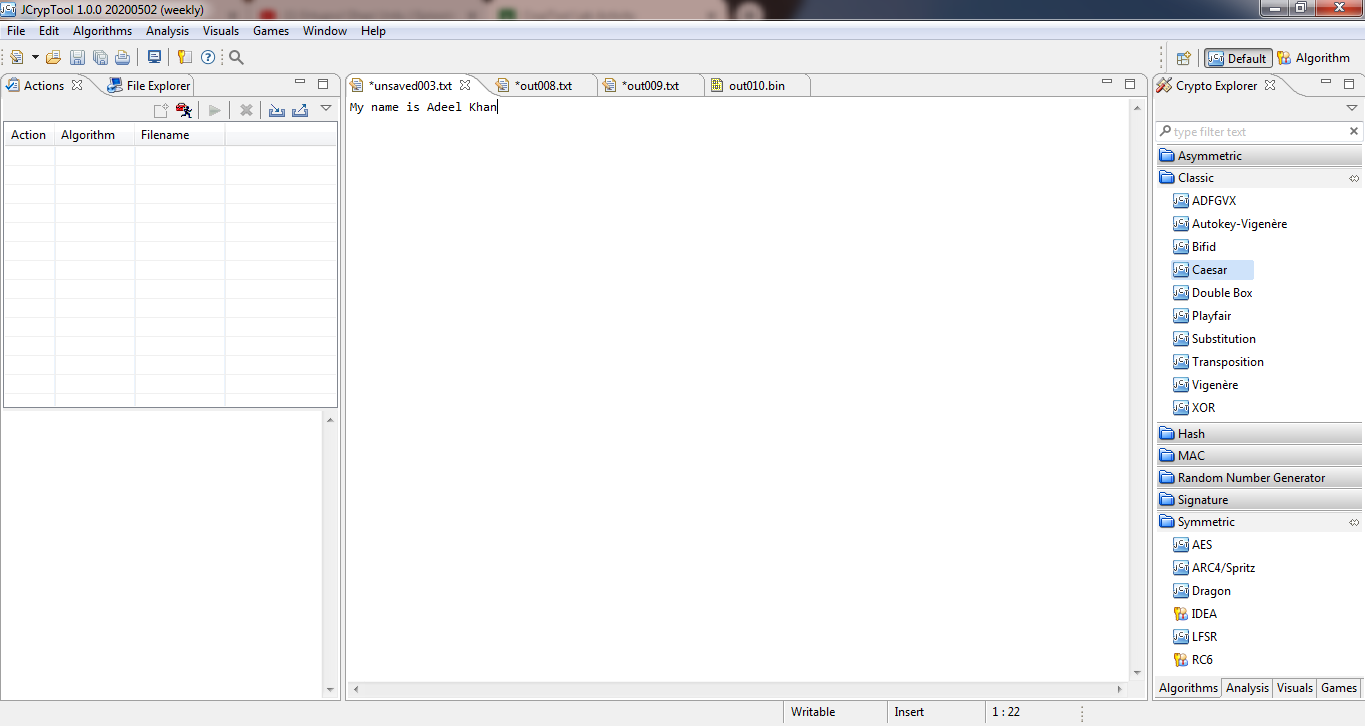
****

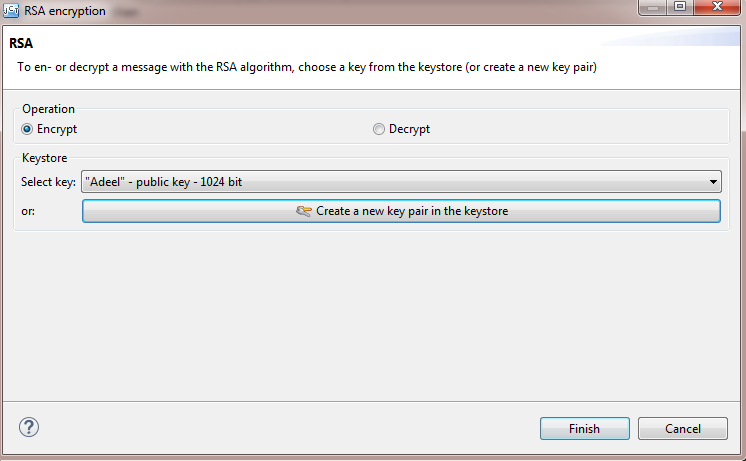
**Decryption:**

****

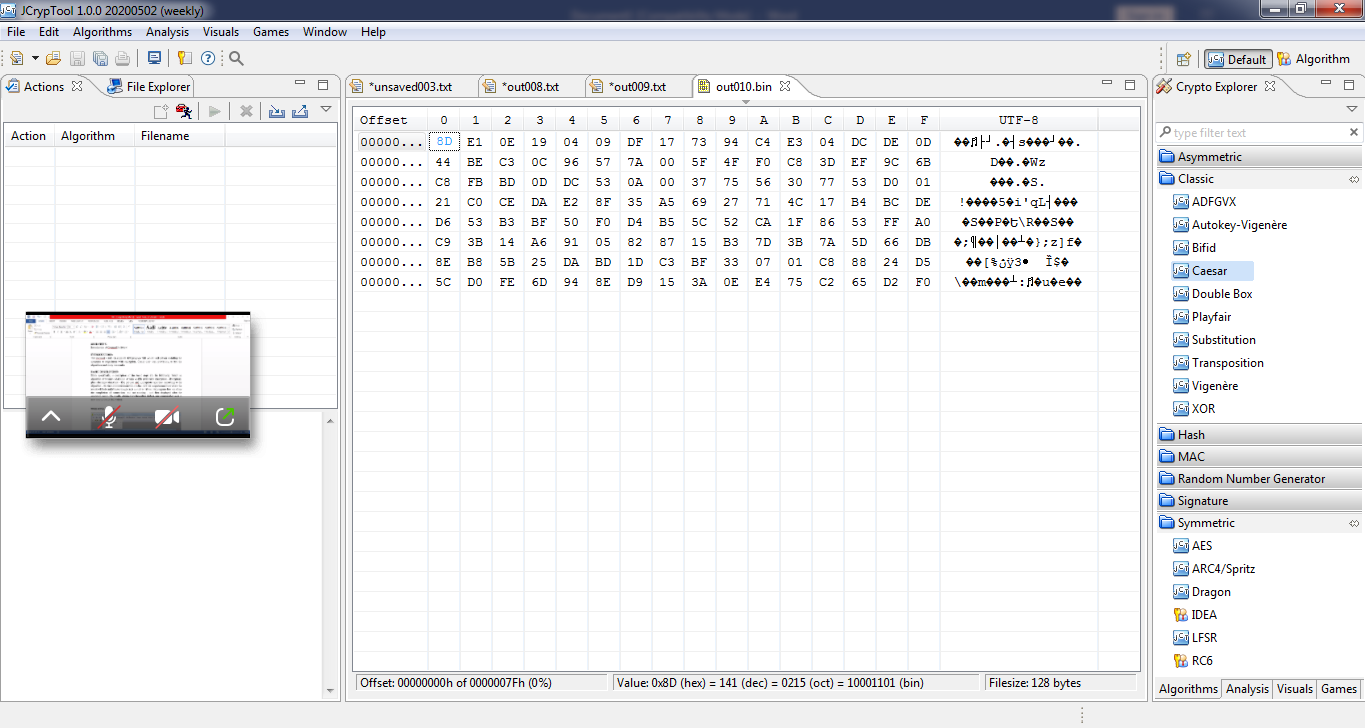
1. **RSA:**

**Plain Text:**

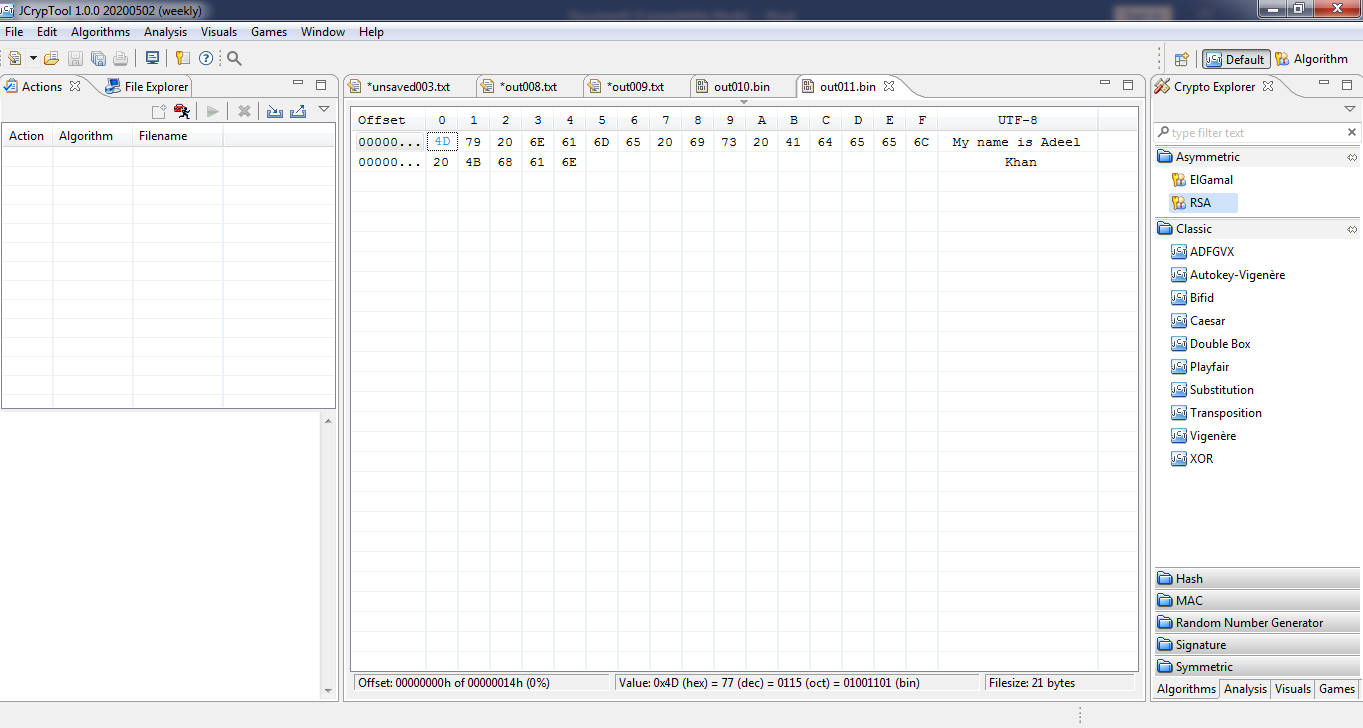
****

**Key: **

**Encryption :**

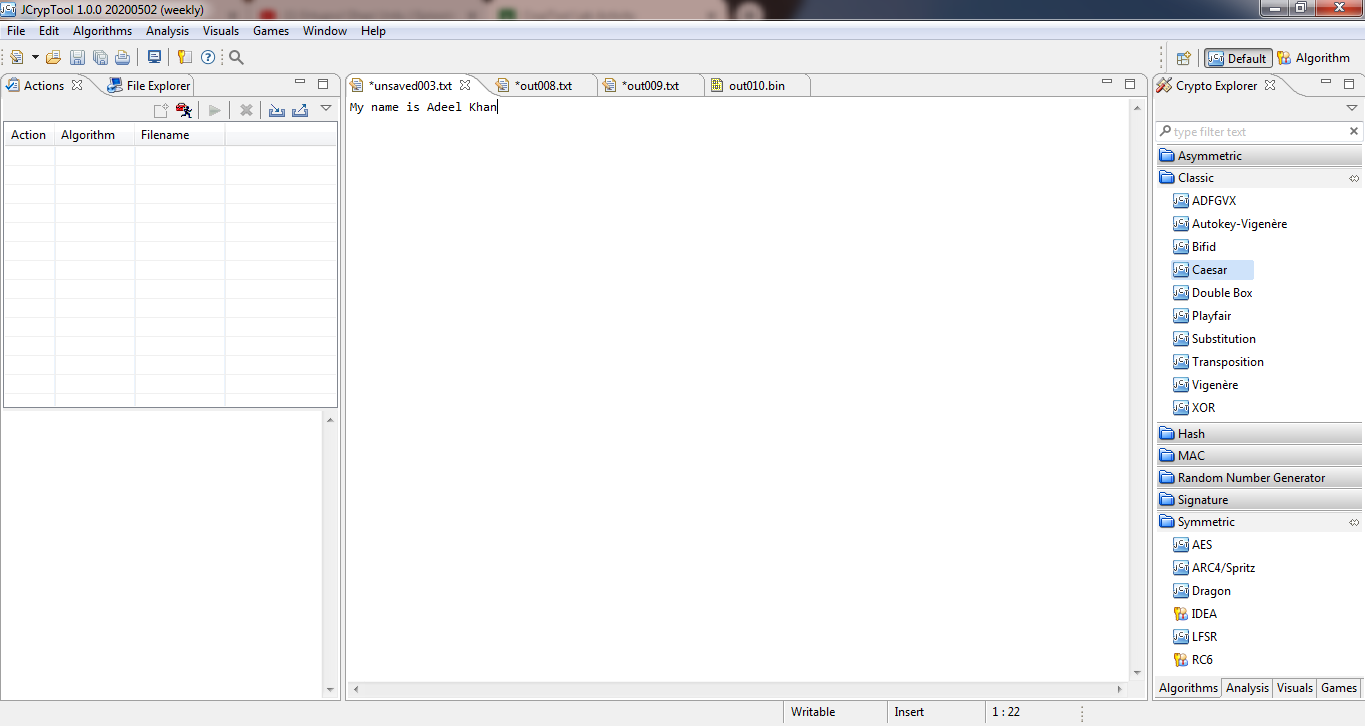
****

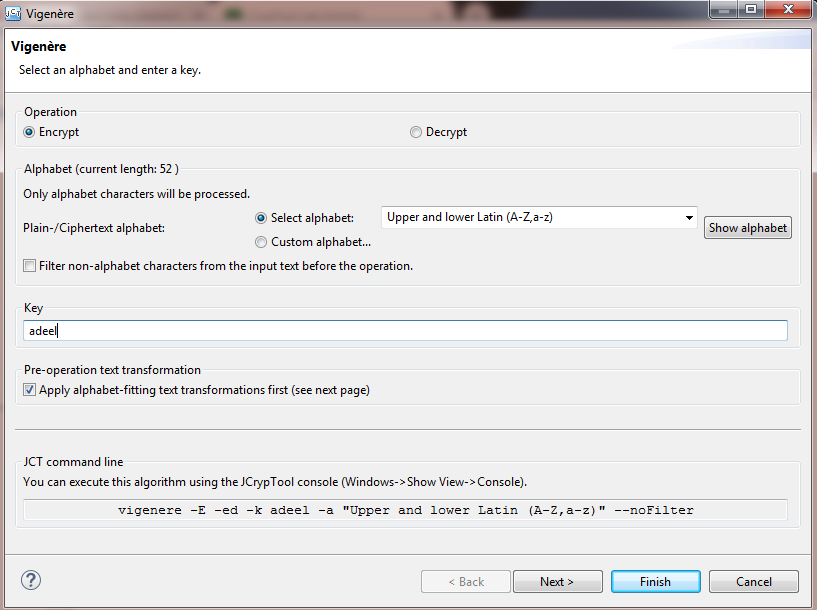
**Decryption:**

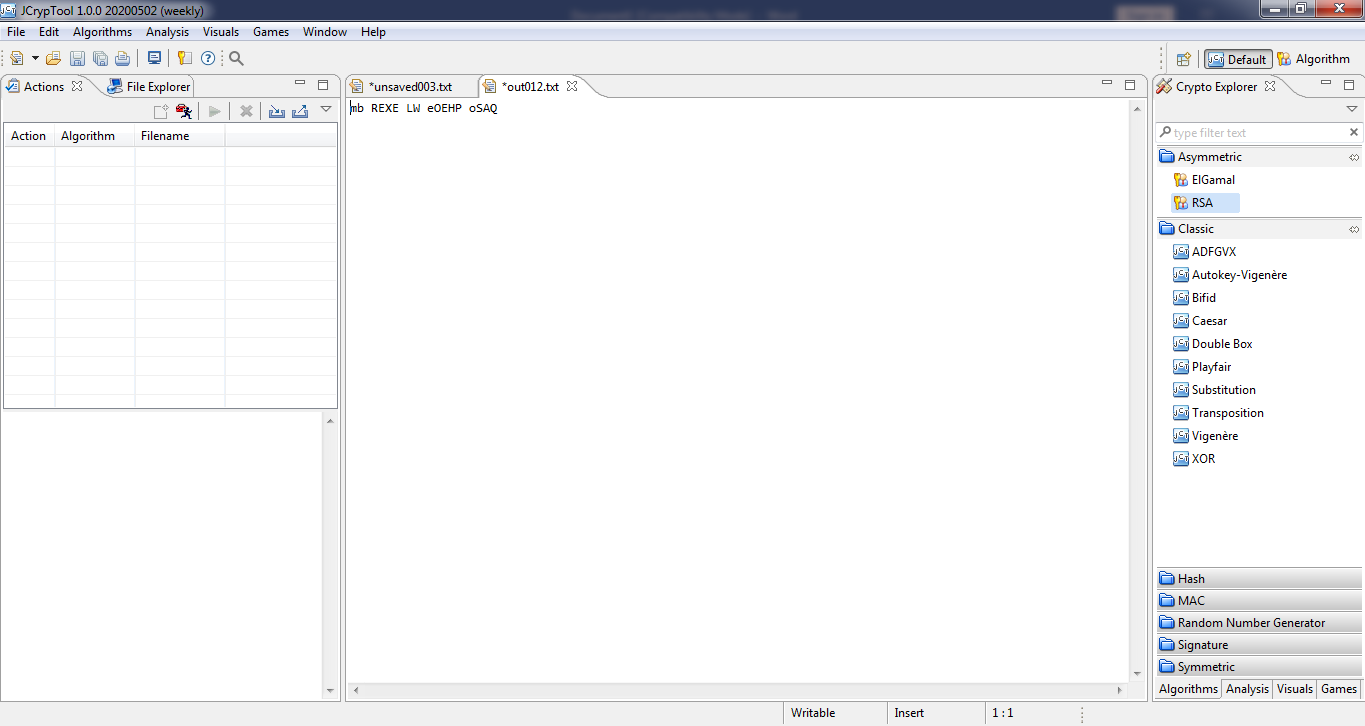
****

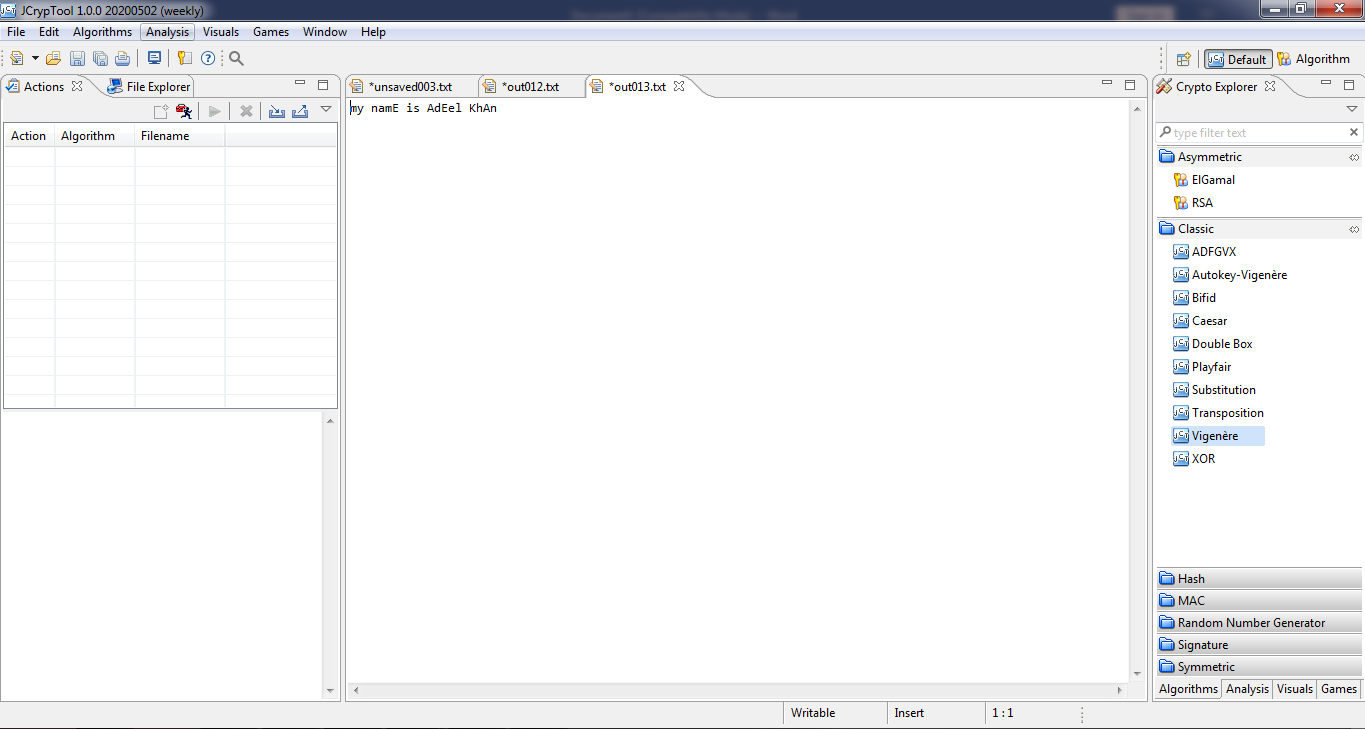
1. **Vigenere :**

**Text:**

****

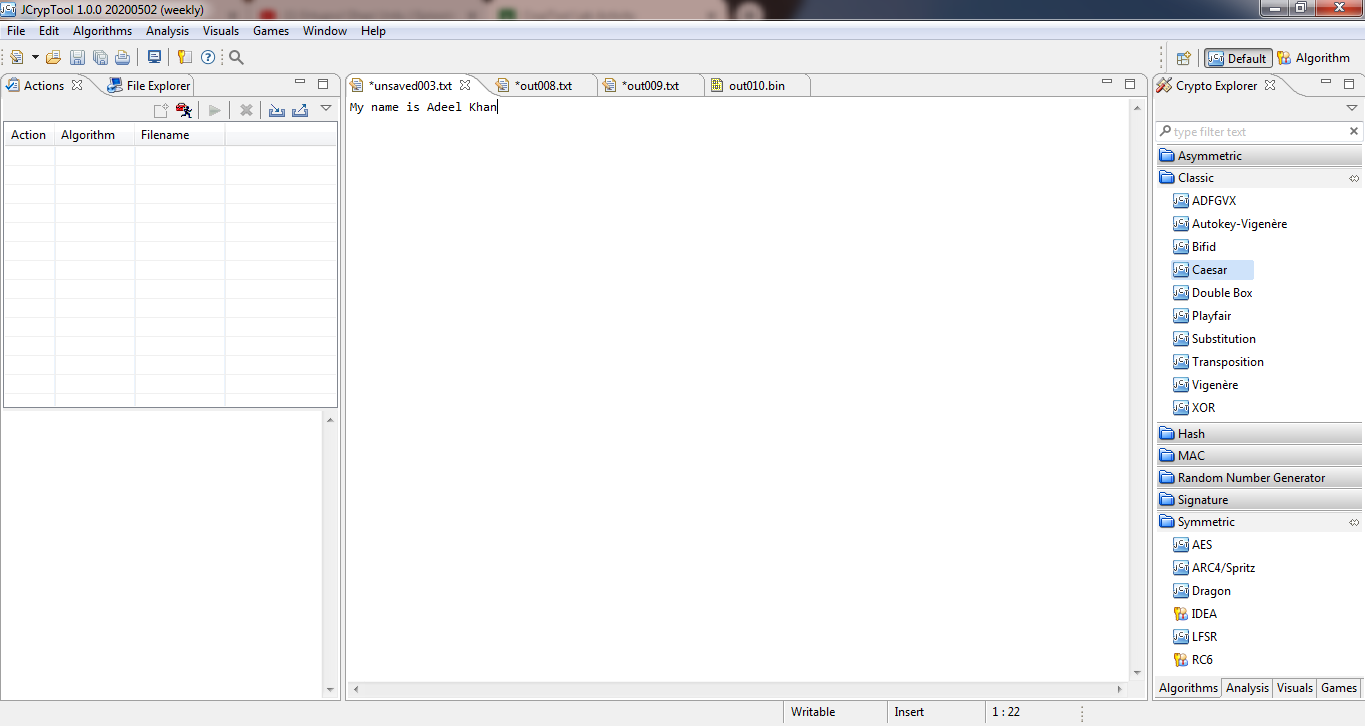
**Key: **

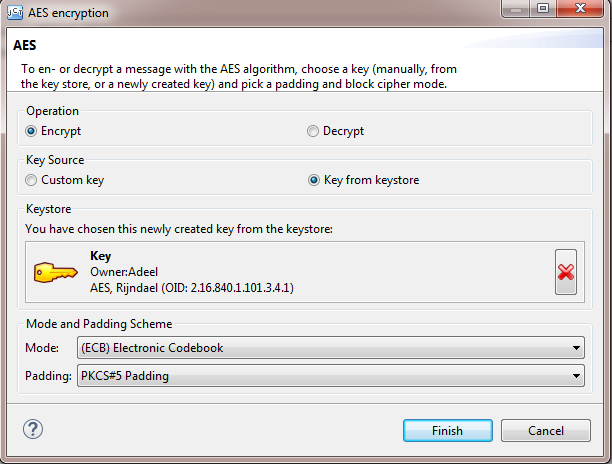
**Encryption: **

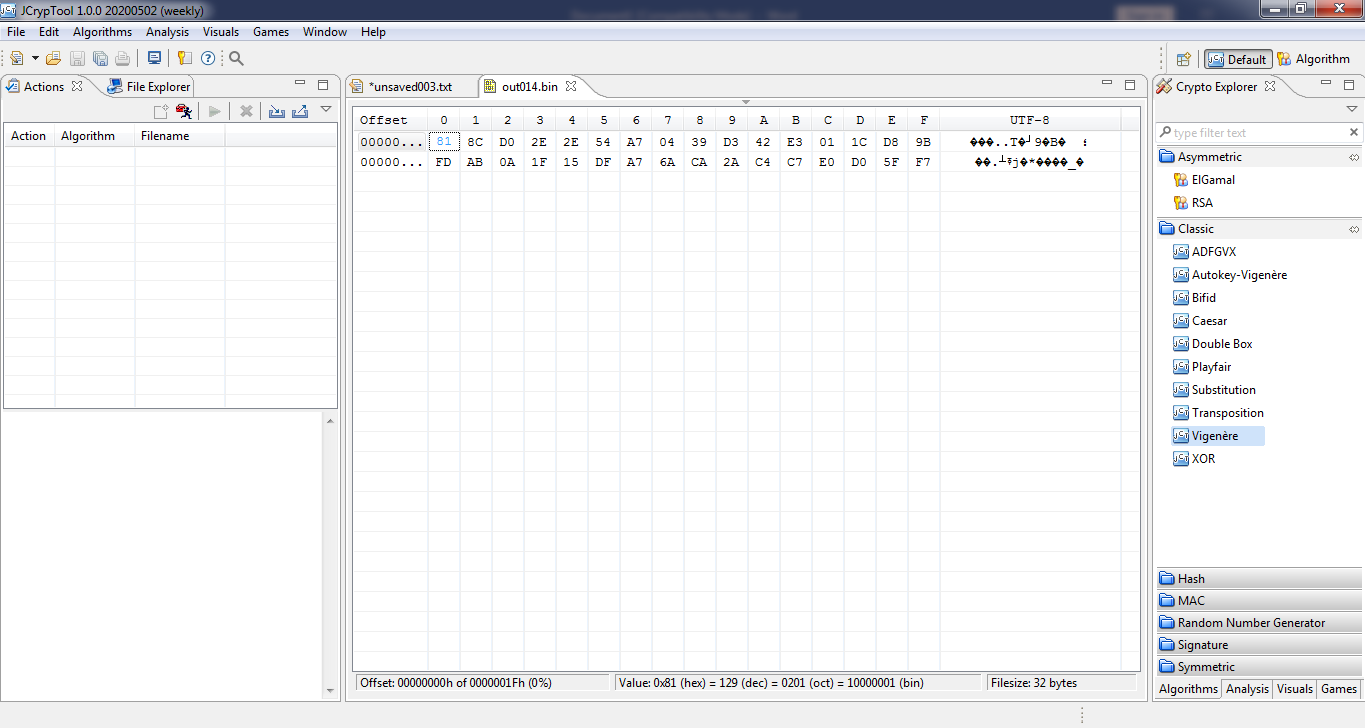
**Decryption: **

1. **AES :**

**Text:**

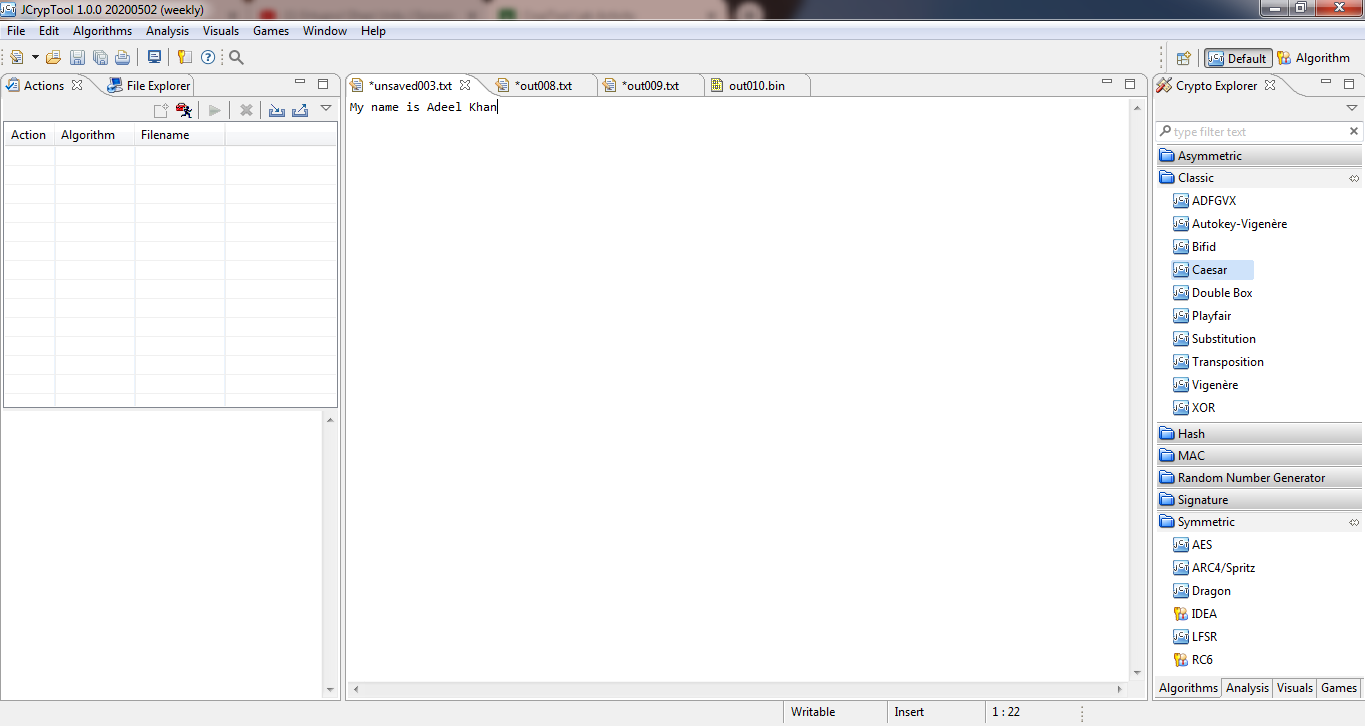
****

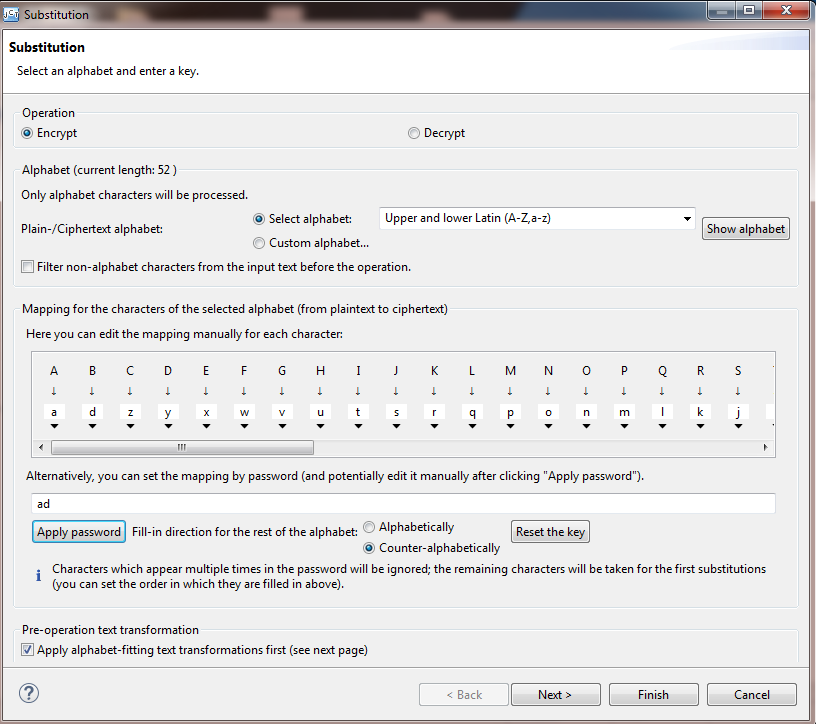
**Key: **

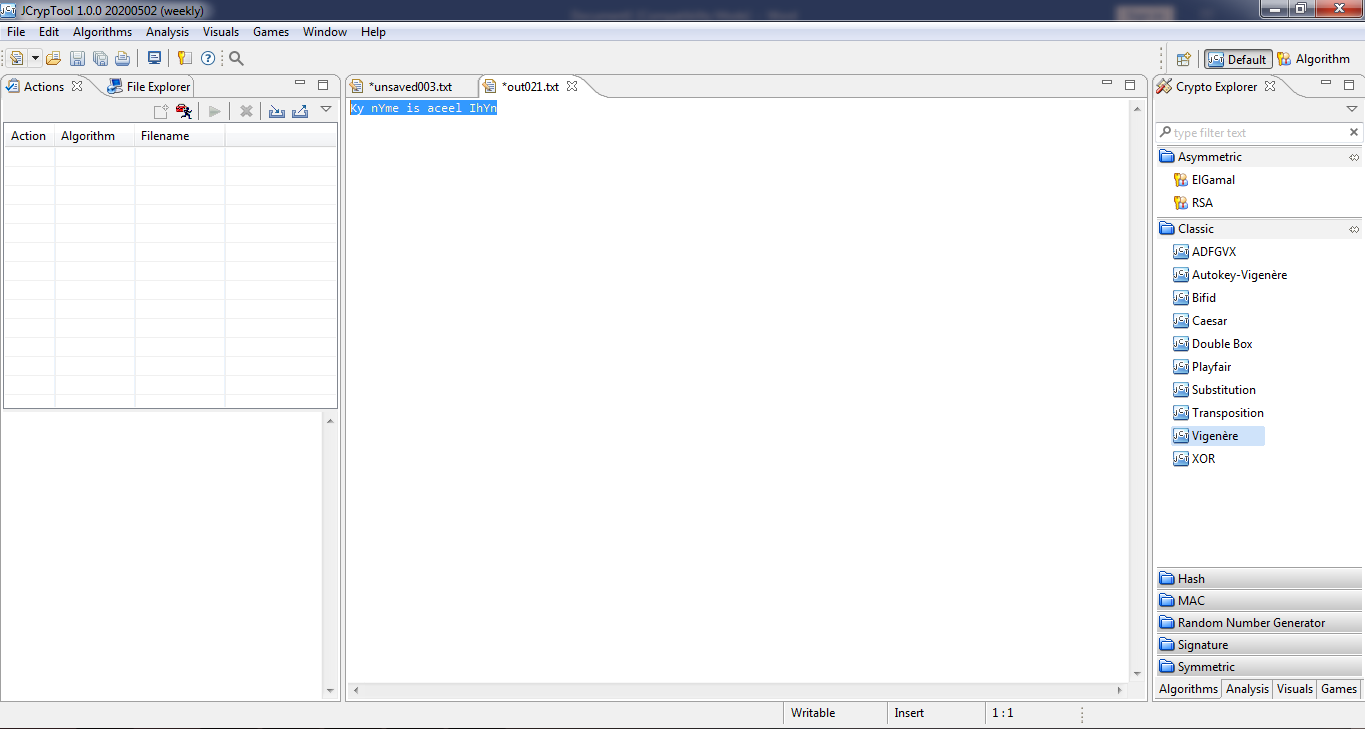
**Encryption: **

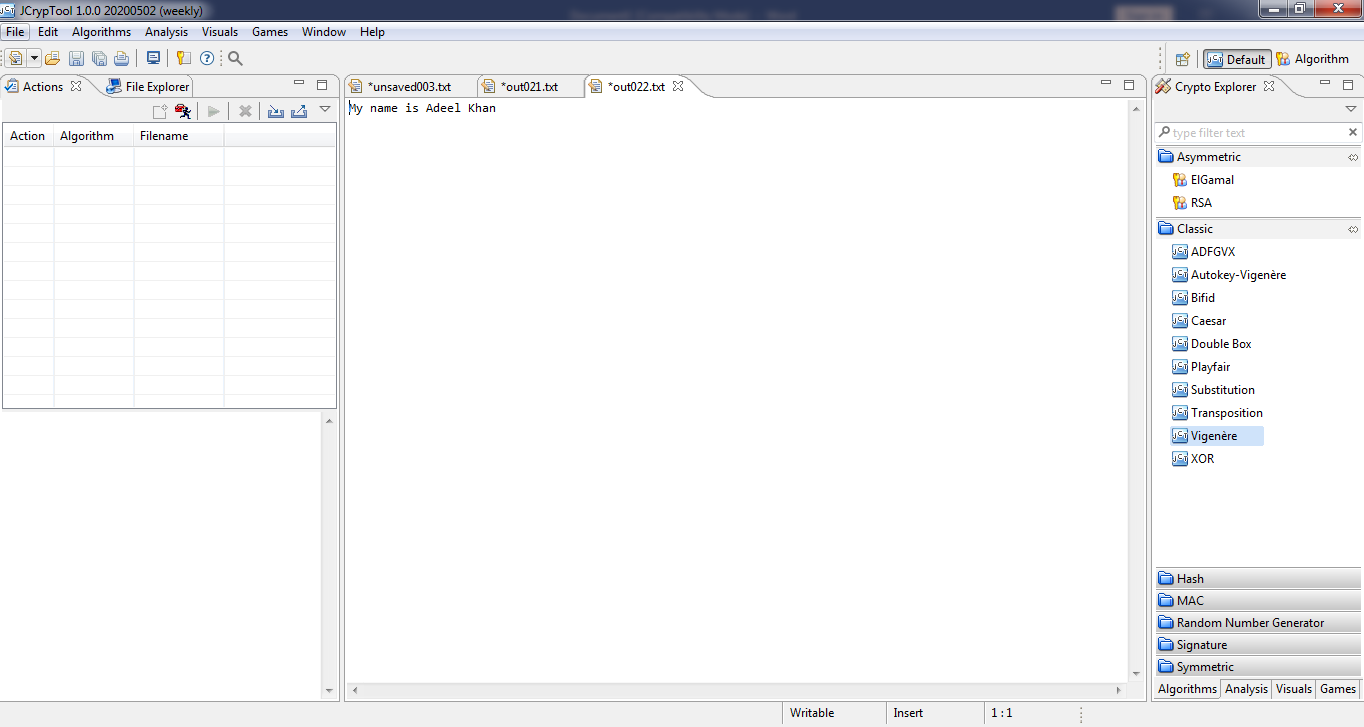
**Decryption: **

1. **Substitution:**

**Text:**

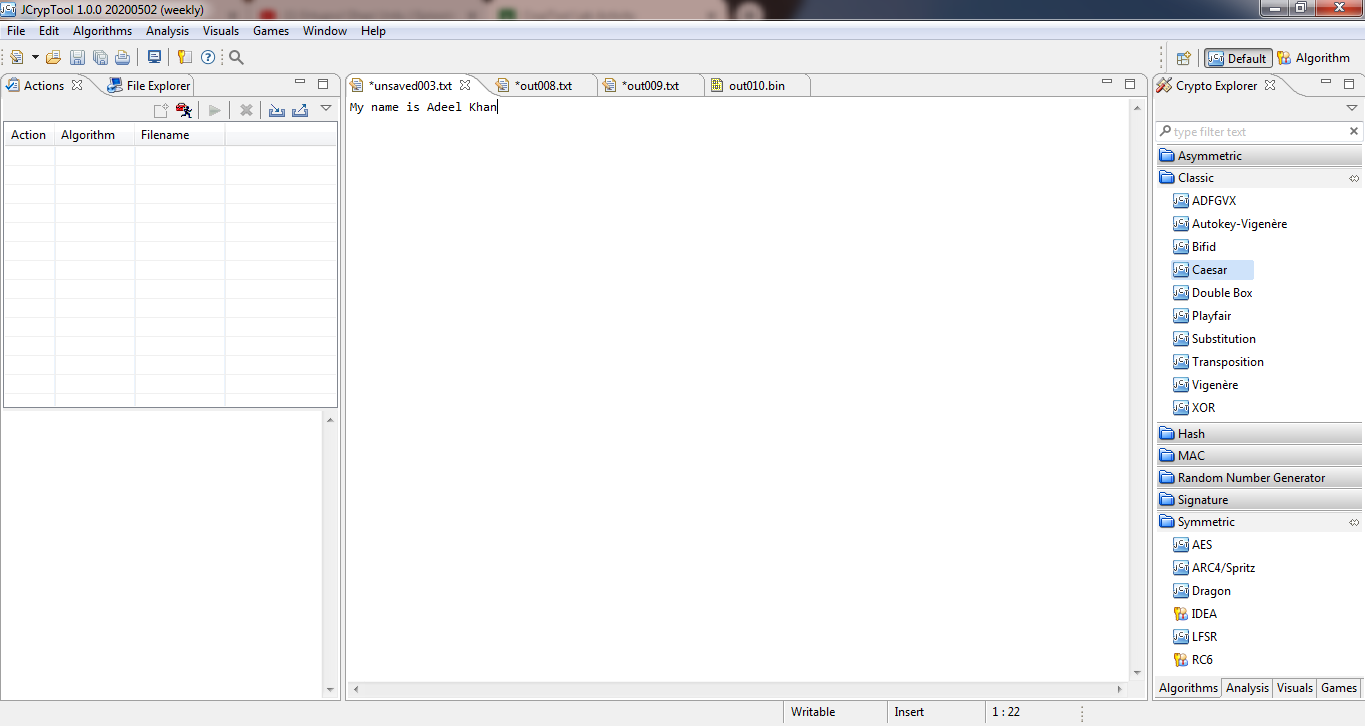
**Key: **

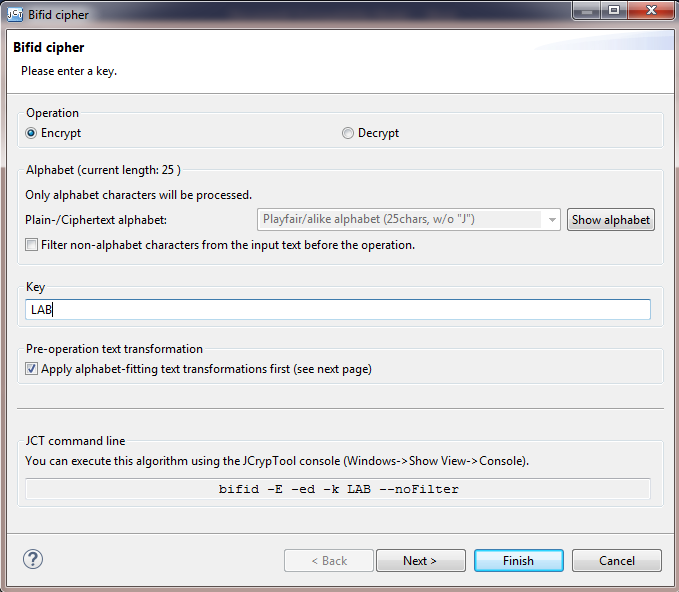
**Encryption: **

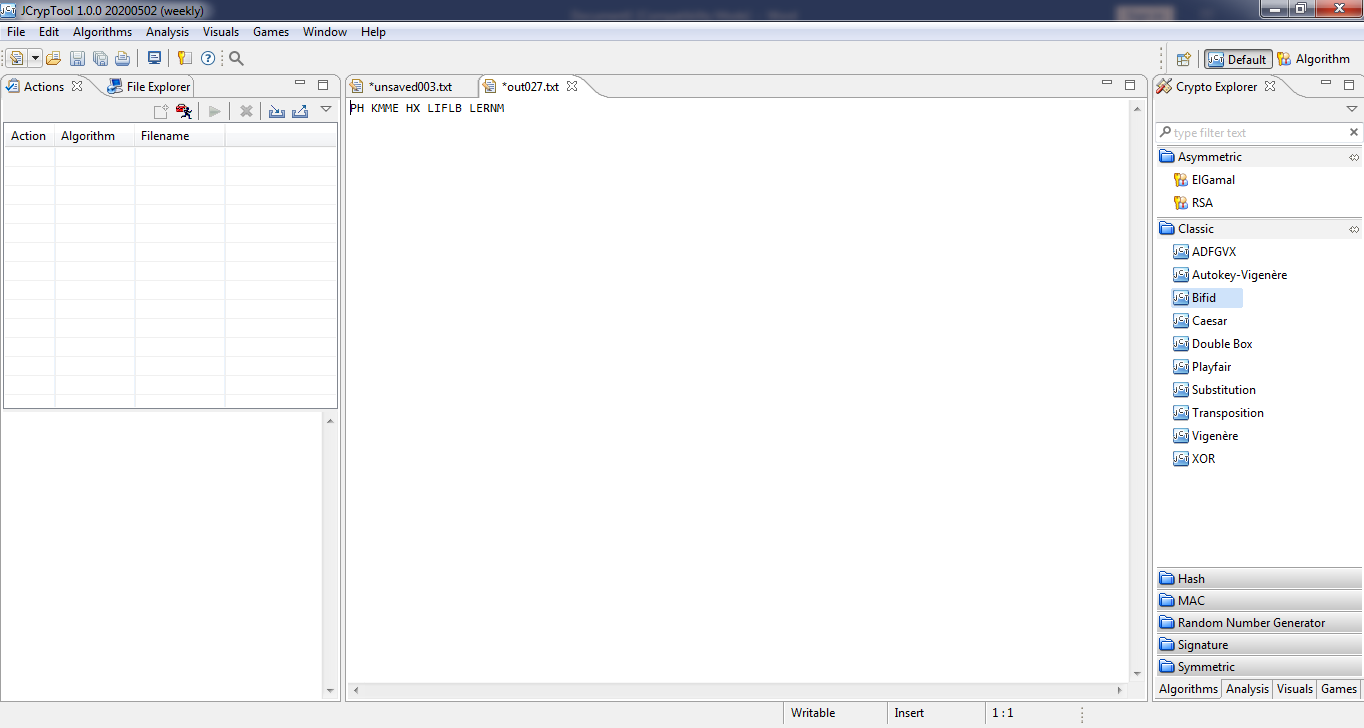
**decryption: **

1. **Bifid cipher:**

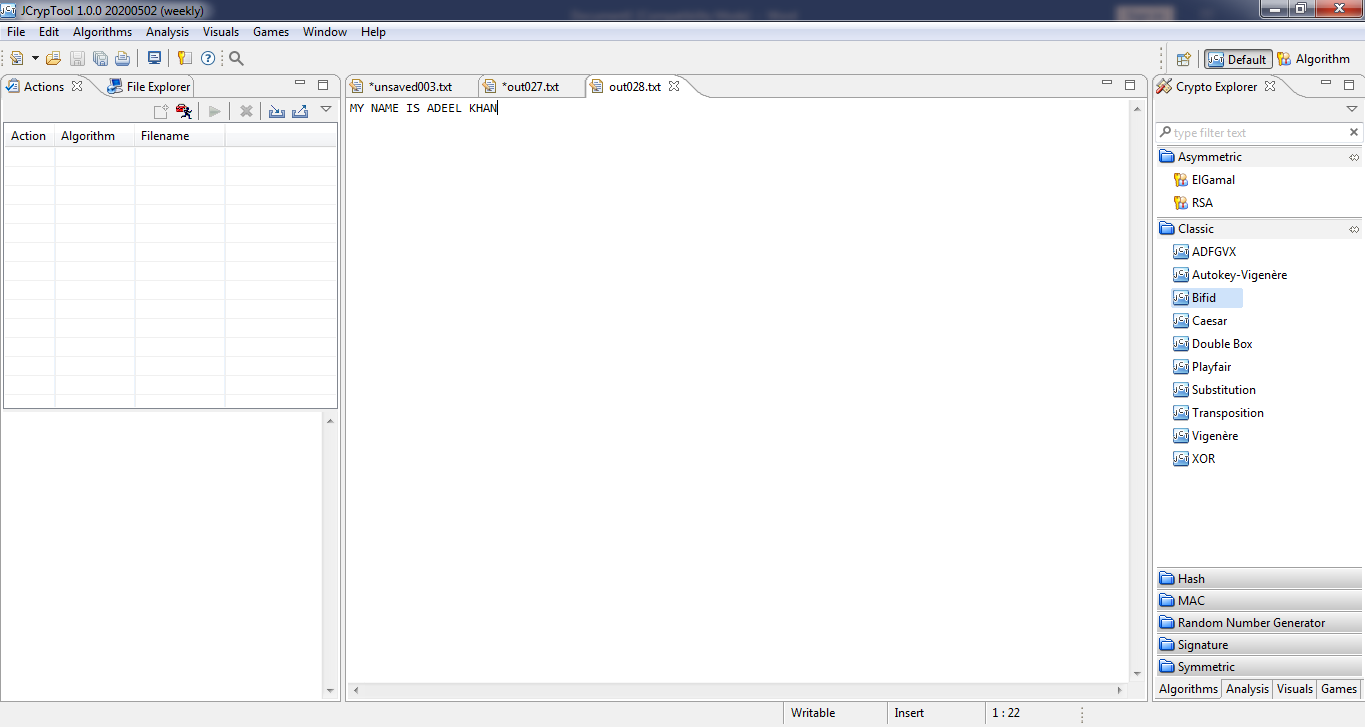
**Text:**

****

**Key: **

**Encryption: **

**Decryption:**

****

**LAB ACTIVITY NO 5:**

**CAESAR CIPHER**

**AIM:**

ToimplementaprogramforencryptingaplaintextanddecryptingaciphertextusingCaesarCipher(shiftcipher)substitutiontechnique

**ALGORITHM DESCRIPTION:**

* + Itisatypeof[substitutioncipher](https://en.wikipedia.org/wiki/Substitution_cipher)inwhicheachletterinthe[plaintext](https://en.wikipedia.org/wiki/Plaintext)isreplacedbyaletter somefixed number of positions downthe[alphabet](https://en.wikipedia.org/wiki/Alphabet).Forexample,witha leftshiftof3,DwouldbereplacedbyA,E wouldbecome B,andsoon.
  + Themethodisnamedafter[JuliusCaesar](https://en.wikipedia.org/wiki/Julius_Caesar),whouseditinhisprivatecorrespondence.
  + Thetransformationcanberepresentedbyaligningtwoalphabets;thecipheralphabetistheplainalphabetrotatedleftorrightbysomenumberofpositions.
  + Theencryptioncanalsoberepresentedusing[modulararithmetic](https://en.wikipedia.org/wiki/Modular_arithmetic)byfirsttransformingthelettersintonumbers,accordingtothescheme,A=0,B=1,Z=25.
  + Encryptionofaletterxbyashiftn canbedescribedmathematicallyas,En(x)=(x+n)mod26
  + Decryptionisperformedsimilarly,Dn(x)=(x-n)mod26

**PROGRAM:**

importjava.util.\*;

classcaesarCipher{

publicstaticStringencode(Stringenc,intoffset){

offset= offset %26+26;

StringBuilderencoded=newStringBuilder();

for(chari:enc.toCharArray()){

if(Character.isLetter(i)){

if(Character.isUpperCase(i)){

encoded.append((char)('A'+(i- 'A'+offset)%26));

}

else{

encoded.append((char)('a'+(i-'a'+offset)%26 ));

}

}

else{

encoded.append(i);

}

}

returnencoded.toString();

}

public staticString decode(String enc,intoffset){

return encode(enc, 26-offset);

}

public staticvoidmain (String[]args)throws java.lang.Exception{

String msg ="Hello welcome toSecurity Laboratory";

System.out.println("simulation of Caesar Cipher");

System.out.println("input message : " +msg);

System.out.printf( "encoded message :");

System.out.println(caesarCipher.encode(msg, 12));

System.out.printf( "decoded message :");

System.out.println(caesarCipher.decode(caesarCipher.encode(msg, 12), 12));

}

}

**stdin:**

Standardinputisempty

**stdout:**

simulation of Caesar Cipher

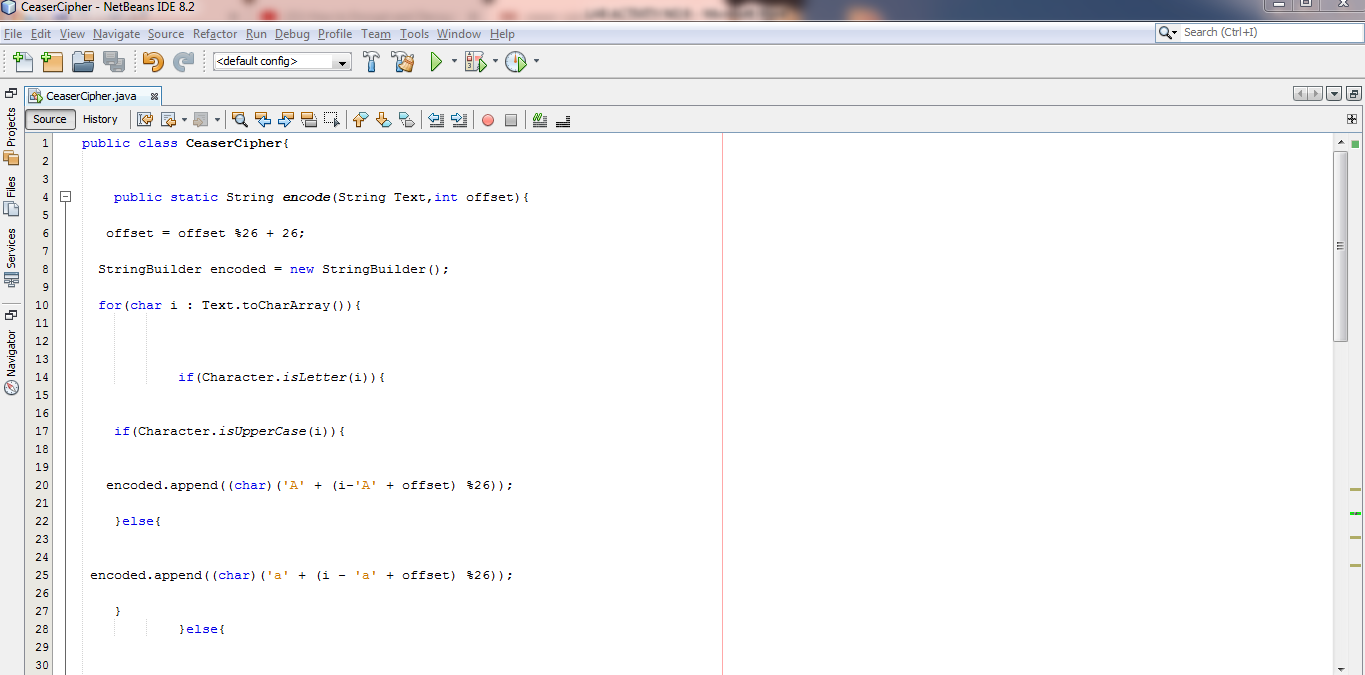
input message : Hello welcome to Security Laboratory

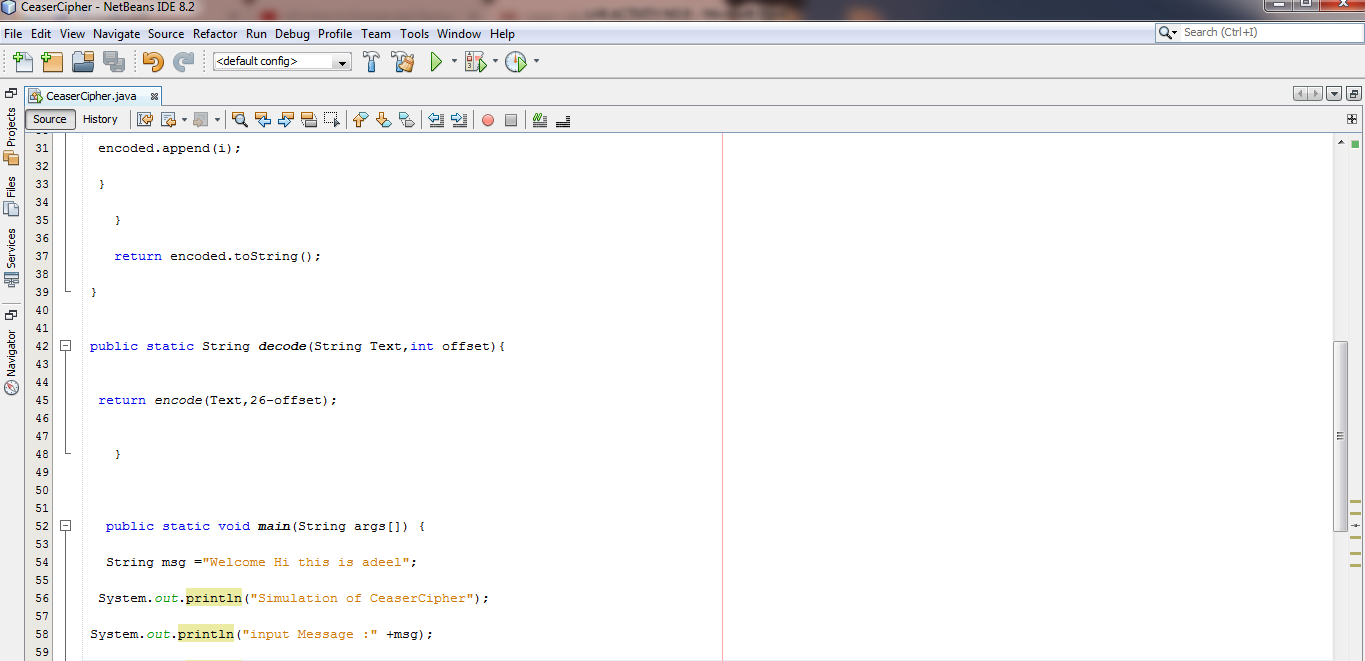
encodedmessage: TqxxaiqxoayqfaEqogdufkXmnadmfadk

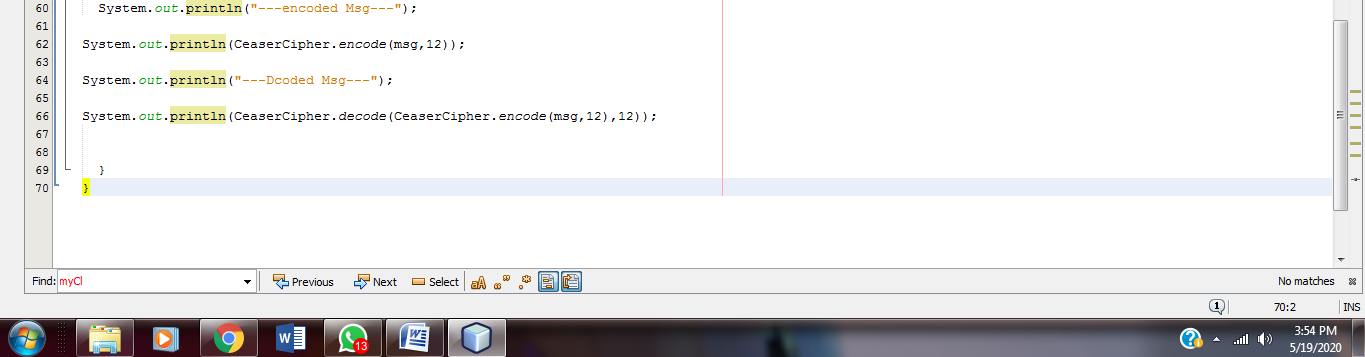
decoded message: Hello welcometo Security Laboratory

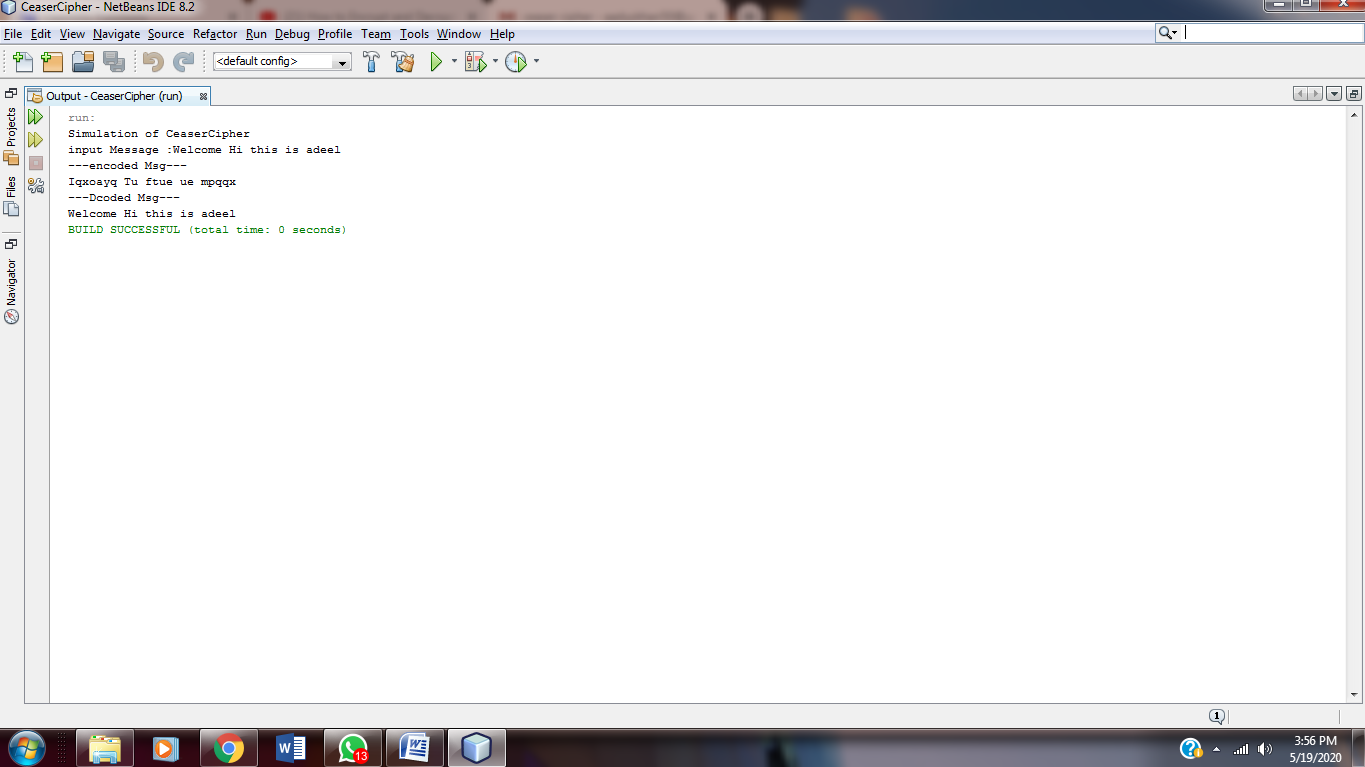
**LAB (TASK) 5**

**Code:**







**Output: **

**LAB ACTIVITY NO 6**

**VIGENERE CIPHER**

**AIM:**

To implement a program for encryption and decryption using vigenere cipher substitution technique

**ALGORITHM DESCRIPTION:**

* The Vigenerecipher is a method of encrypting alphabetic text by using a series of differentCaesarciphers based on the letters of a keyword.
* It is a simpleform of polyalphabeticsubstitution.
* To encrypt, a table of alphabetscan be used,termed a Vigeneresquare, orVigenere table.
* It consists of the alphabetwritten out 26 times indifferentrows,eachalphabetshifted cyclically to the leftcompared to the previous alphabet,corresponding to the 26 possible Caesarciphers.
* Atdifferent points in the encryptionprocess, the cipheruses a differentalphabetfrom one of the rows used.
* The alphabetateach point depends on a repeating keyword.

**PROGRAM:**

importjava.util.\*;

classvigenereCipher{

staticStringencode(Stringtext, final String key){

String res = "";

text=text.toUpperCase();

for(inti=0,j=0;i<text.length(); i++){

charc=text.charAt(i);

if (c <'A' || c>'Z'){

continue;

}

res += (char)((c +key.charAt(j) -2\* 'A') %26+ 'A'); j= ++j %key.length();

}

return res;

}

staticStringdecode(Stringtext, final String key){

String res = "";

text=text.toUpperCase();

for(inti=0,j=0;i<text.length(); i++) {

charc=text.charAt(i);

if (c <'A' || c>'Z')

{

continue;

}

res += (char)((c -key.charAt(j)+26)%26+ 'A');

j= ++j %key.length();

}

return res;

}

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

String key = "VIGENERECIPHER";

Stringmsg= "SecurityLaboratory";

System.out.println("simulation ofVigenere Cipher");

System.out.println("input message:"+msg); Stringenc= encode(msg, key); System.out.println("encoded message :"+enc);

System.out.println("decoded message :"+ decode(enc, key));

}

}

**stdin:**

Standard inputisempty

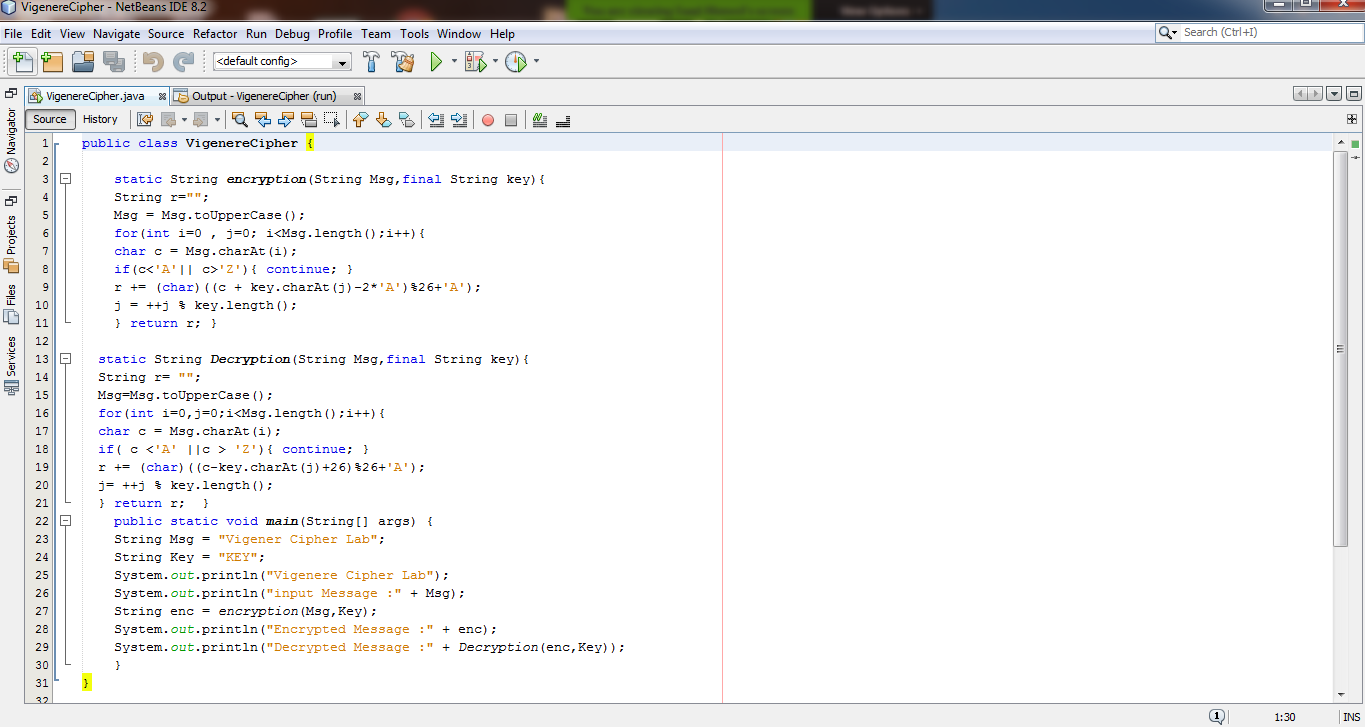
**stdout:**

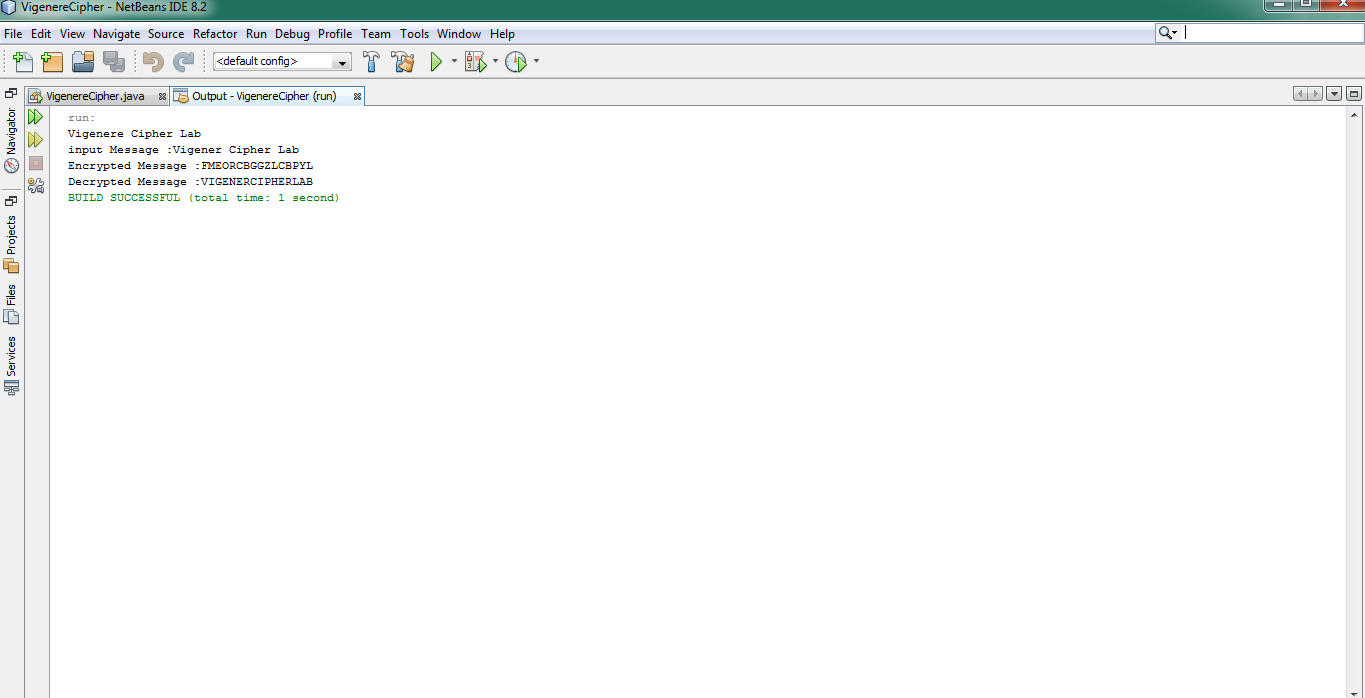
simulationofVigenereCipher

input message:SecurityLaboratory

encodedmessage: NMIYEMKCNIQVVROWXC decoded message: SECURITYLABORATORY

**LAB (TASK) 6**

**Source:**

Output: 

**LAB ACTIVITY NO 7:**

**RAIL FENCE CIPHER**

**AIM:**

Toimplementaprogramforencryptionanddecryptionusingrailfencetranspositiontechnique.

**ALGORITHM DESCRIPTION:**

* + Intherailfencecipher,theplaintextiswrittendownwardsanddiagonallyonsuccessive"rails"ofanimaginaryfence,thenmovingupwhenwereachthebottomrail.
  + Whenwereachthetoprail,themessageiswrittendownwardsagainuntilthewholeplaintextiswrittenout.

**PROGRAM:**

importjava.util.\*;

classrailfenceCipherHelper{

int depth;

Stringencode(String msg,int depth) throws Exception{

intr= depth;

intl=msg.length(); intc= l/depth;

intk=0;

charmat[][]= new char[r][c];

Stringenc= "";

for(int i=0; i<c; i++){

for(int j=0; j<r; j++){

if (k !=l){

mat[j][i]=msg.charAt(k++);

}

else{

mat[j][i]= 'X';

}

}

}

for(int i=0; i<r; i++){

for(int j=0; j<c; j++){

enc+=mat[i][j];

}

}

returnenc;

}

Stringdecode(String encmsg, int depth) throws Exception{

intr= depth;

intl=encmsg.length(); intc= l/depth;

intk=0;

charmat[][]= new char[r][c];

Stringdec= "";

for(int i=0; i<r; i++){

for(int j=0; j< c; j++)

{

mat[i][j]=encmsg.charAt(k++);

}

}

for(int i=0; i<c; i++){

for(int j=0; j<r;j++){

dec+=mat[j][i];

}

}

returndec;

}

}

classrailfenceCipher{

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

railfenceCipherHelperrf= new railfenceCipherHelper();

Stringmsg, enc, dec;

msg="hellorailfencecipher"; \

int depth =2;

enc=rf.encode(msg, depth);

dec=rf.decode(enc, depth);

System.out.println("simulation ofRailfenceCipher");

System.out.println("input message:"+msg);

System.out.println("encoded message :"+enc);

System.out.printf( "decoded message :"+dec);

}

}

**stdin:**

Standard inputisempty

**stdout:**

simulationofRailfenceCipher

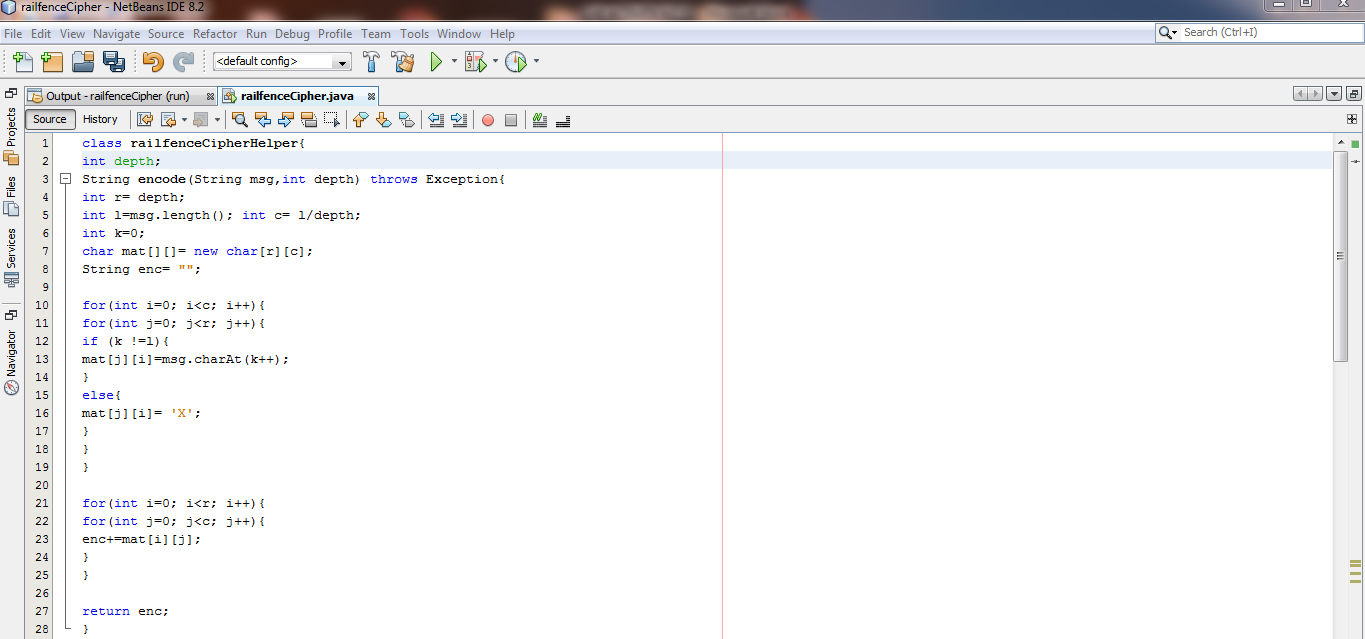
input message:hellorailfencecipher

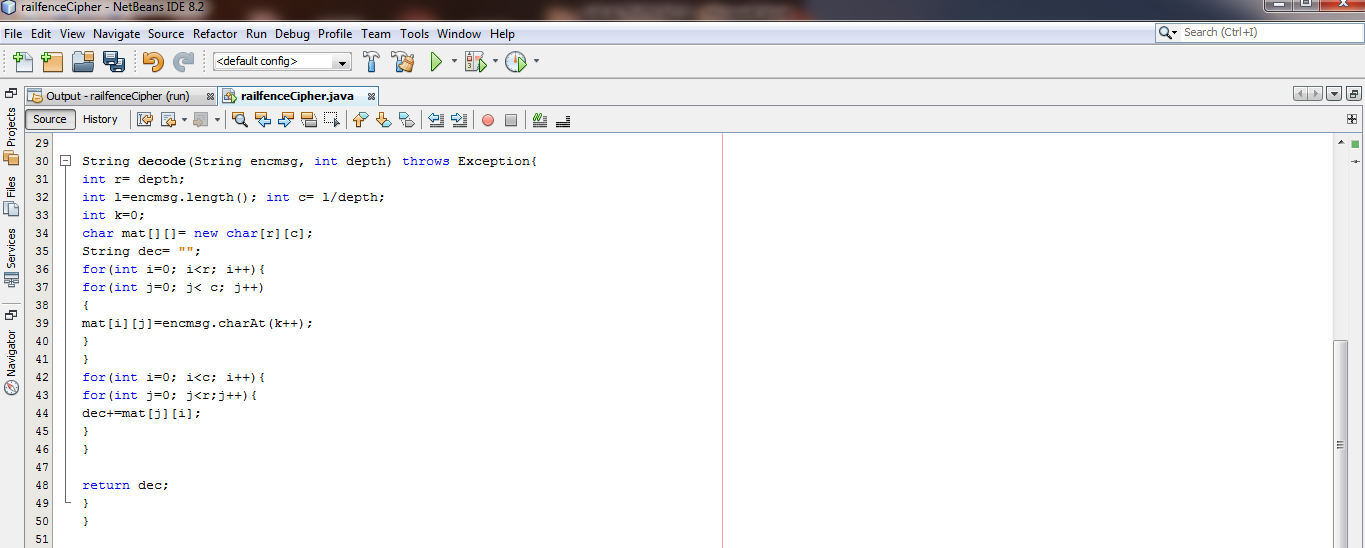
encodedmessage:hloaleccpeelrifneihr

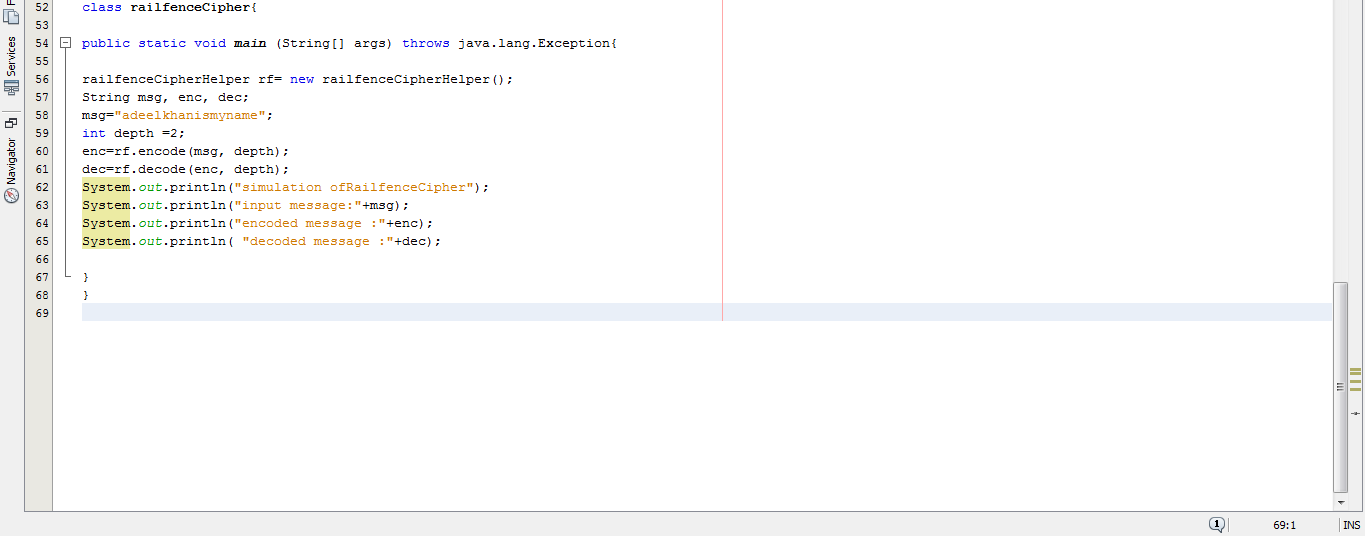
decoded message:hellorailfencecipher

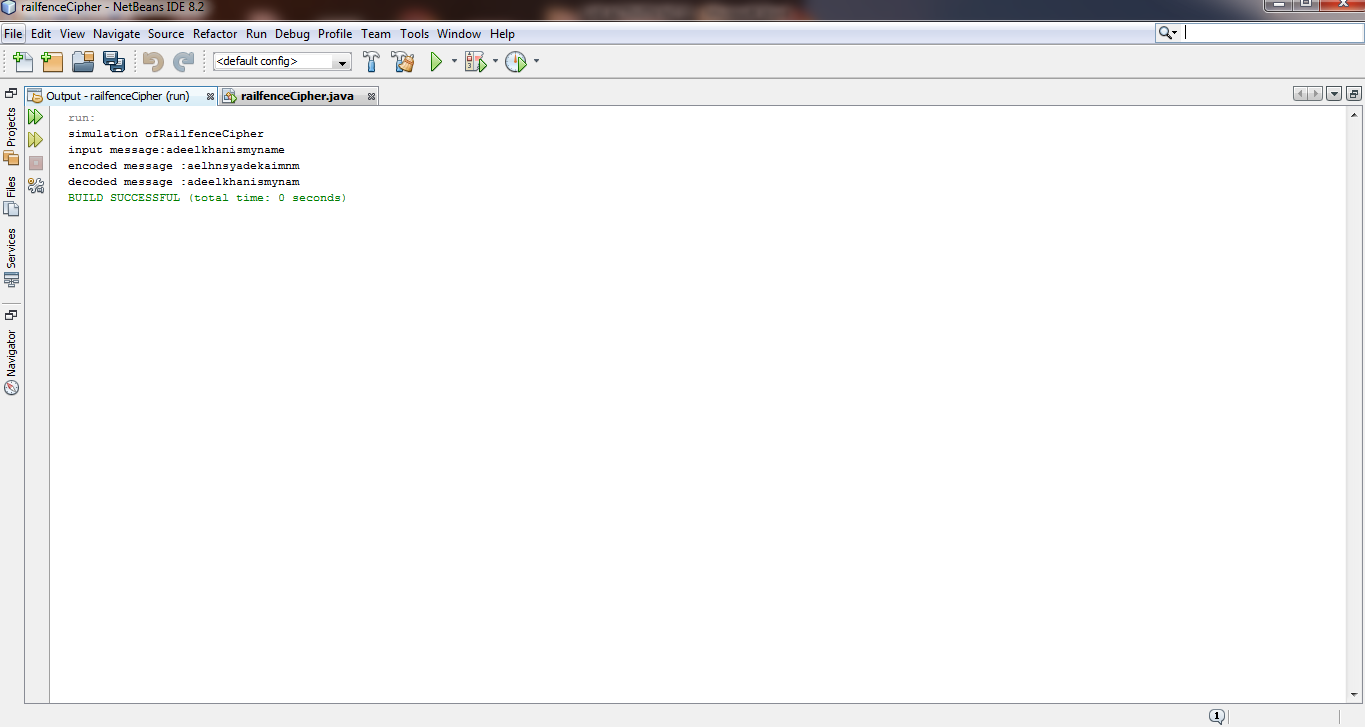
**LAB (TASK) 7**

Code:







Output: 

**LAB ACTIVITY NO 8:**

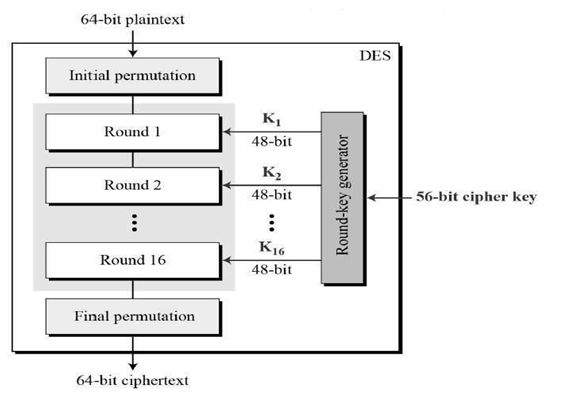
**DATA ENCRYPTION STANDARD (DES)**

**AIM:**

TodevelopaprogramtoimplementDataEncryptionStandardforencryptionanddecryption.

**ALGORITHM DESCRIPTION:**

* + TheDataEncryptionStandard(DES)isasymmetric-keyblockcipherpublishedbytheNationalInstituteofStandardsandTechnology(NIST).
  + DESisanimplementationofaFeistelCipher. Ituses16roundFeistelstructure.Theblocksizeis64-bit.
  + Though,keylengthis64-bit,DEShasaneffectivekeylengthof56bits,since8ofthe64bitsofthekeyarenotusedbytheencryptionalgorithm(functionascheck bitsonly).
  + GeneralStructureofDESisdepictedinthefollowingillustration



**PROGRAM:**

importjavax.swing.\*;

importjava.security.SecureRandom;

importjavax.crypto.Cipher;

importjavax.crypto.KeyGenerator;

importjavax.crypto.SecretKey;

importjavax.crypto.spec.SecretKeySpec;

importjava.util.Random;

class DES {

byte[] skey= new byte[1000];

StringskeyString;

static byte[] raw;

StringinputMessage,encryptedData,decryptedMessage;

public DES() {

try{

generateSymmetricKey();

inputMessage=JOptionPane.showInputDialog(null,"Enter message to encrypt"); byte[] ibyte=inputMessage.getBytes();

byte[] ebyte=encrypt(raw, ibyte);

StringencryptedData=newString(ebyte); S

ystem.out.println("Encrypted message "+encryptedData);

JOptionPane.showMessageDialog(null,"Encrypted Data "+"\n"+encryptedData); byte[] dbyte= decrypt(raw,ebyte);

StringdecryptedMessage= new String(dbyte);

System.out.println("Decrypted message "+decryptedMessage);

JOptionPane.showMessageDialog(null,"Decrypted Data "+"\n"+decryptedMessage);

}

catch(Exception e) {

System.out.println(e);

}

}

voidgenerateSymmetricKey() {

try{

Random r= new Random();

intnum=r.nextInt(10000);

Stringknum=String.valueOf(num);

byte[] knumb=knum.getBytes();

skey=getRawKey(knumb);

skeyString=new String(skey);

System.out.println("DES Symmetric key = "+skeyString);

}

catch(Exception e) {

System.out.println(e);

}

}

private static byte[] getRawKey(byte[] seed) throws Exception {

KeyGeneratorkgen=KeyGenerator.getInstance("DES");

SecureRandomsr=SecureRandom.getInstance("SHA1PRNG");

sr.setSeed(seed);

kgen.init(56, sr);

SecretKeyskey=kgen.generateKey();

raw=skey.getEncoded();

return raw;

}

private static byte[] encrypt(byte[] raw, byte[] clear) throws Exception {

SecretKeySpecskeySpec= new SecretKeySpec(raw, "DES");

Ciphercipher=Cipher.getInstance("DES"); cipher.init(Cipher.ENCRYPT\_MODE, skeySpec);

byte[] encrypted =cipher.doFinal(clear);

return encrypted;

}

private static byte[] decrypt(byte[] raw, byte[] encrypted) throws Exception{

SecretKeySpecskeySpec= new SecretKeySpec(raw, "DES");

Ciphercipher=Cipher.getInstance("DES"); cipher.init(Cipher.DECRYPT\_MODE, skeySpec);

byte[] decrypted =cipher.doFinal(encrypted);

returndecrypted;

}

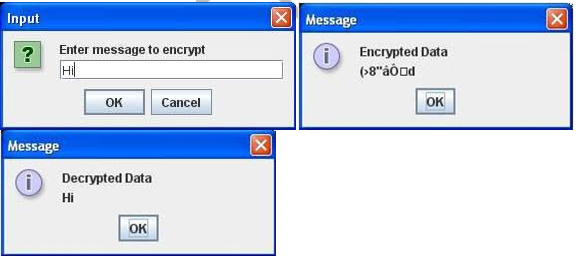
public static void main(String args[]){

DES des= new DES();

}

}

**Output:**

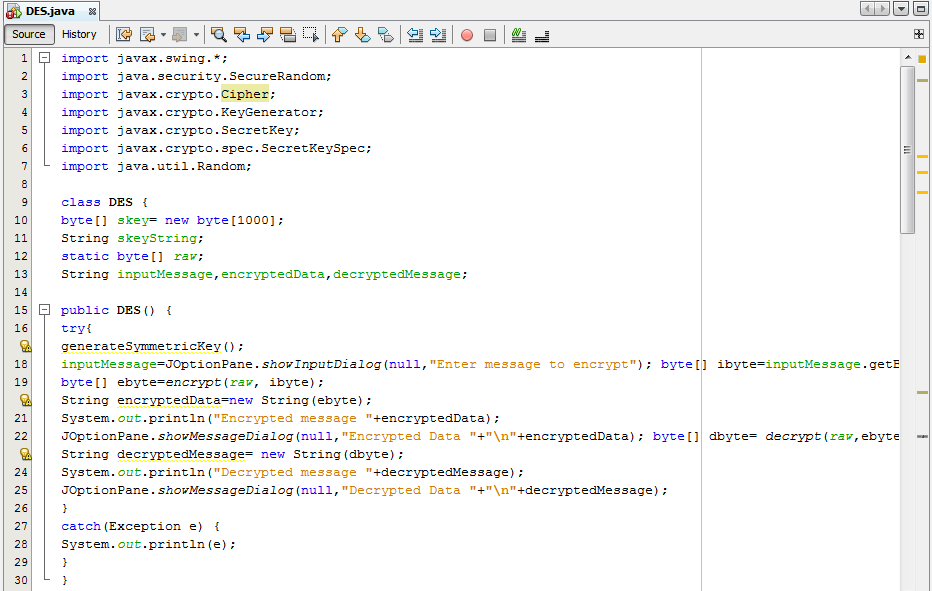


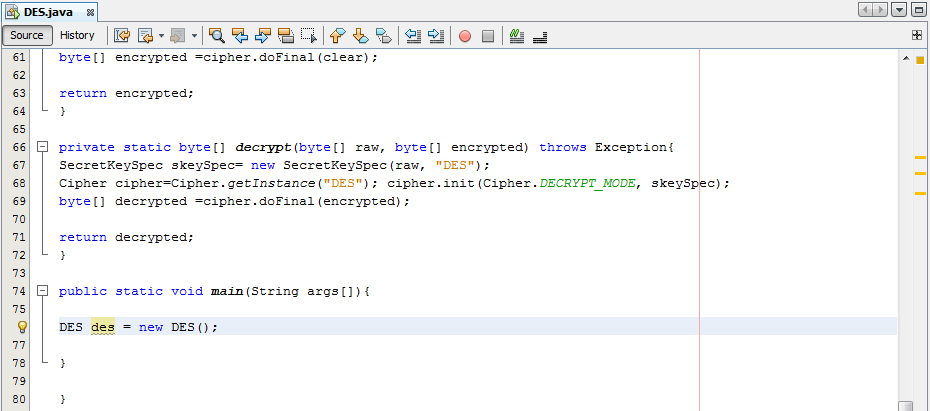
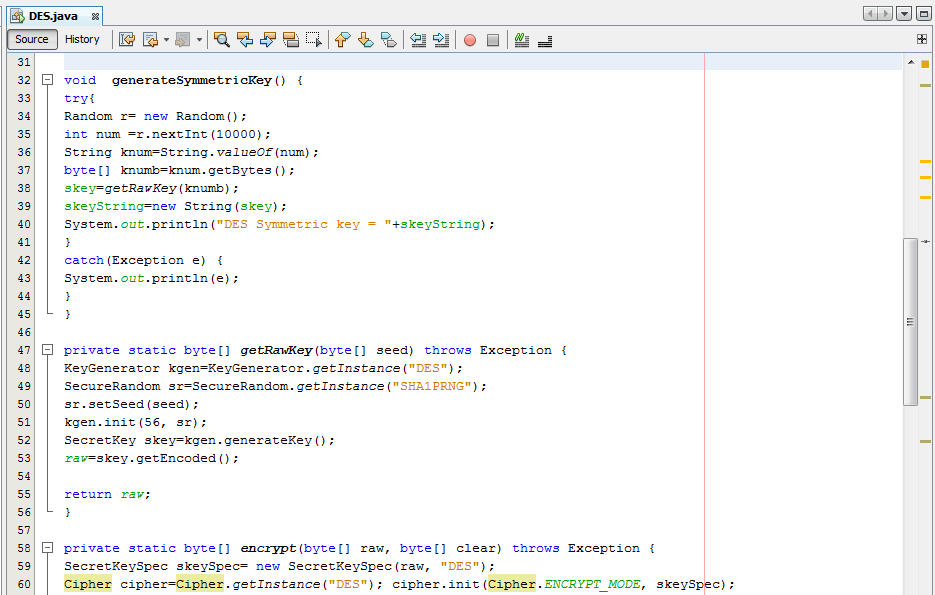
**LAB (TASK 8)**

**DATA ENCRYPTION STANDARD (DES)**

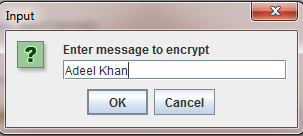
TodevelopaprogramtoimplementDataEncryptionStandardforencryptionanddecryption.

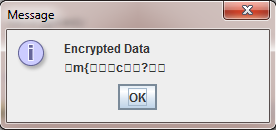
**Source:**

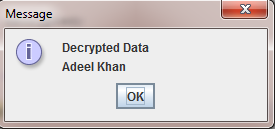
****

****

**Output**

****

****

****

**LAB ACTIVITY NO 9:**

**Random Number**

Random numbers are useful for a variety of purposes, such as generating data encryption keys, simulating and modelling complex phenomena and for selecting random samples from larger data sets. They have also been used aesthetically, for example in literature and music, and are of course ever popular for games and gambling. When discussing single numbers, a random number is one that is drawn from a set of possible values, each of which is equally probable, i.e., a uniform distribution. When discussing a sequence of random numbers, each number drawn must be statistically independent of the others.

### Pseudo Random Number Generator (PRNG)

As the word ‗pseudo‘ suggests, pseudo-random numbers are not random in the way you might expect, at least not if you're used to dice rolls or lottery tickets. Essentially, PRNGs are algorithms that use mathematical formulae or simply pre-calculated tables to produce sequences of numbers that appear random. A good example of a PRNG is the linear congruential method. A good deal of research has gone into pseudo-random number theory, and modern algorithms for generating pseudo-random numbers are so good that the numbers look exactly like they were really random.

The basic difference between PRNGs and TRNGs is easy to understand if you compare computer-generated random numbers to rolls of a die. Because PRNGs generate random numbers by using mathematical formulae or pre-calculated lists, using one corresponds to someone rolling a die many times and writing down the results. Whenever you ask for a die roll, you get the next on the list. Effectively, the numbers appear random, but they are really predetermined. TRNGs work by getting a computer to actually roll the die — or, more commonly, use some other physical phenomenon that is easier to connect to a computer than a die is.

### True Random Number Generators (TRNGs)

In comparison with PRNGs, TRNGs extract randomness from physical phenomena and introduce it into a computer. You can imagine this as a die connected to a computer, but typically people use a physical phenomenon that is easier to connect to a computer than a die is. The physical phenomenon can be very simple, like the little variations in somebody's mouse movements or in the amount of time between keystrokes. In practice, however, you have to be careful about which source you choose. For example, it can be tricky to use keystrokes in this fashion, because keystrokes are often buffered by the computer's operating system, meaning that several keystrokes are collected before they are sent to the program waiting forthem.

#### Practical No. 6: Program in C++ to generate Pseudo Random numbers in a range

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Coding \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

#include <iostream>

#include<cstdlib> // for rand(), srand() #include<ctime> // fortime()

using namespace std;

intmain() {

intnum, max, min,nt;

// rand() generate a random number in [0, RAND\_MAX]

cout<<"Input the Minimum value of Random Number " <<endl; cin>>min;

cout<<"Input the Maximum value of Random Number " <<endl; cin>> max;

cout<<"Input how much Random Number to generate " <<endl; cin>>nt;

srand(time(0));

for(inti = 0; i<nt; ++i) {

num= rand() % (max + 1 - min) + min; // need <cstdlib> header

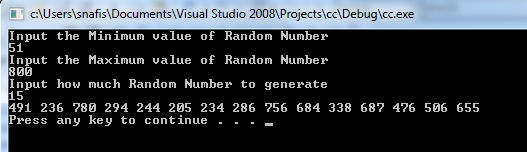
cout<<num<<" ";

}

cout<<endl; system ("pause"); return 0;

}

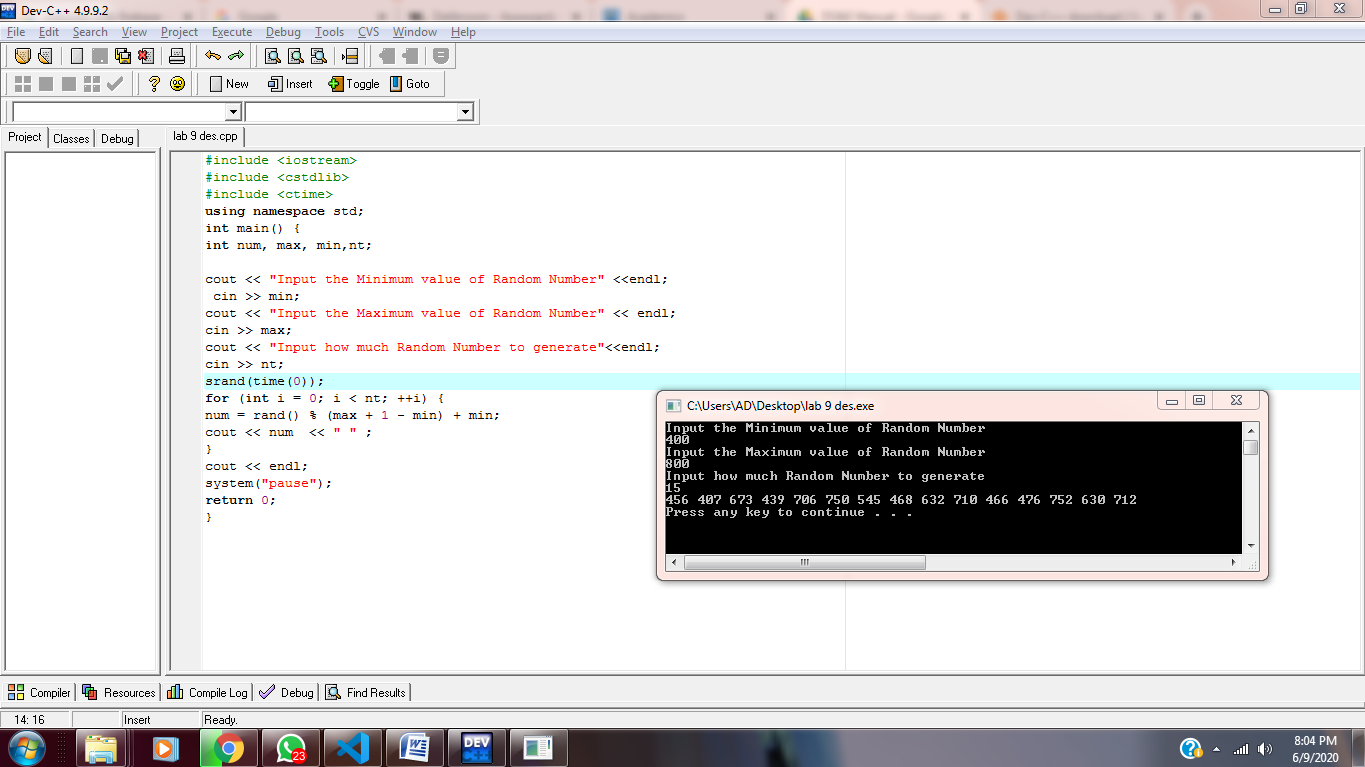
#### /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* OUTPUT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/



**Lab Task**

Run the above program and generate the desired output.

**LAB (TASK) 9**

**Output & Code: **

**LAB ACTIVITY NO 10:**

**Diffie - Hellman Key exchange Algorithm:**

Diffie - Helman is a way of ***generating*** a shared secret between two people in such a way that the secret can't be seen by observing the communication. Here we are not sharing information during the key exchange but creating a key together.

This is particularly useful because you can use this technique to create an encryption key with someone, and then start encrypting your traffic with that key. And even if the traffic is recorded and later analyzed, there's absolutely no way to figure out what the key was, even though the exchanges that created it may have been visible. This is where [**perfect forward secrecy**](http://en.wikipedia.org/wiki/Perfect_forward_secrecy) comes from. Nobody analyzing the traffic at a later date can break in because the key was never saved, never transmitted, and never made visible anywhere.

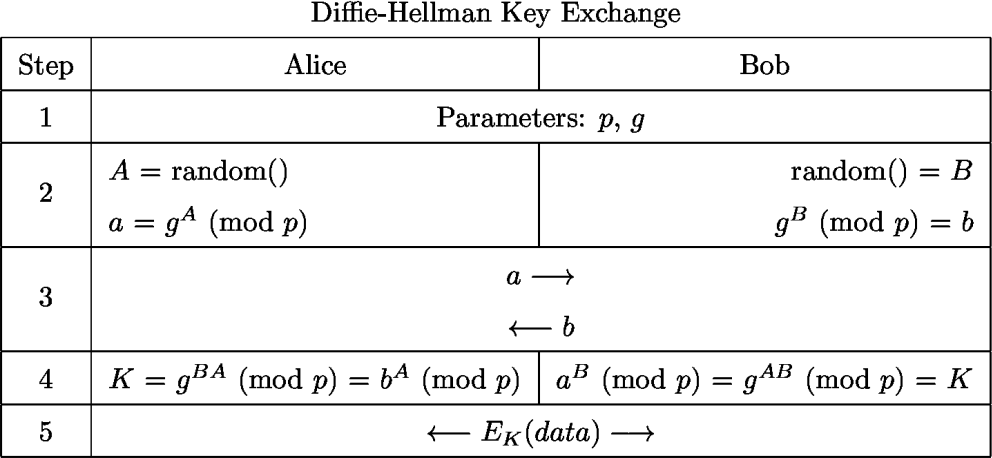
It uses the same underlying principles as public key cryptography but this is not asymmetric cryptography because nothing is ever encrypted or decrypted during the exchange. It is, however, an essential building-block, and was in fact the base upon which asymmetric crypto was later built.

The basic idea works like this:

1. I come up with two prime numbers **g** and **p** and tell you what they are.
2. You then pick a secret number (**a**), but you don't tell anyone. Instead you compute **ga** *mod* **p** and send that result back to me. (We'll call that **A** since it came from **a**).
3. I do the same thing, but we'll call my secret number **b** and the computed number **B**. So I compute**gb** *mod* **p** and send you the result (called "**B**")
4. Now, you take the number I sent you and do the exact same operation with *it*. So that's **Ba** *mod***p**.
5. I do the same operation with the result you sent me, so: **Ab** *mod* **p**.

The "magic" here is that the answer I get at step 5 is *the same number* you got at step

4. Now it's not really magic, it's just math, and it comes down to a fancy property of modulo exponents.



#### Practical No. 11: Program in C++ for Diffie-Hellman Key exchange

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Coding \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

/\* this program calculates the Key for two persons using the Diffie Hellman Key exchange algorithm \*/

#include<stdio.h> #include<iostream>

long long int power(int a,int b,int mod)

{

long long int t; if(b==1)

return a; t=power(a,b/2,mod); if(b%2==0)

return (t\*t)%mod; else

return (((t\*t)%mod)\*a)%mod;

}

long long int calculateKey(int a,int x,int n)

{

return power(a,x,n);

}

int main()

{

int n,g,x,a,y,b;

// both the persons will be agreed upon the common n and g printf("Enter the value of n and g : "); scanf("%d%d",&n,&g);

// first person will choose the x

printf("Enter the value of x for the first person : "); scanf("%d",&x);

a=power(g,x,n);

// second person will choose the y

printf("Enter the value of y for the second person : "); scanf("%d",&y);

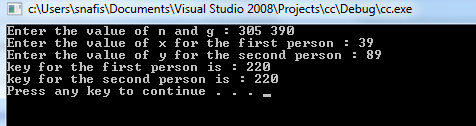
b=power(g,y,n);

printf("key for the first person is : %lld\n",power(b,x,n)); printf("key for the second person is : %lld\n",power(a,y,n));

system ("pause"); return 0;

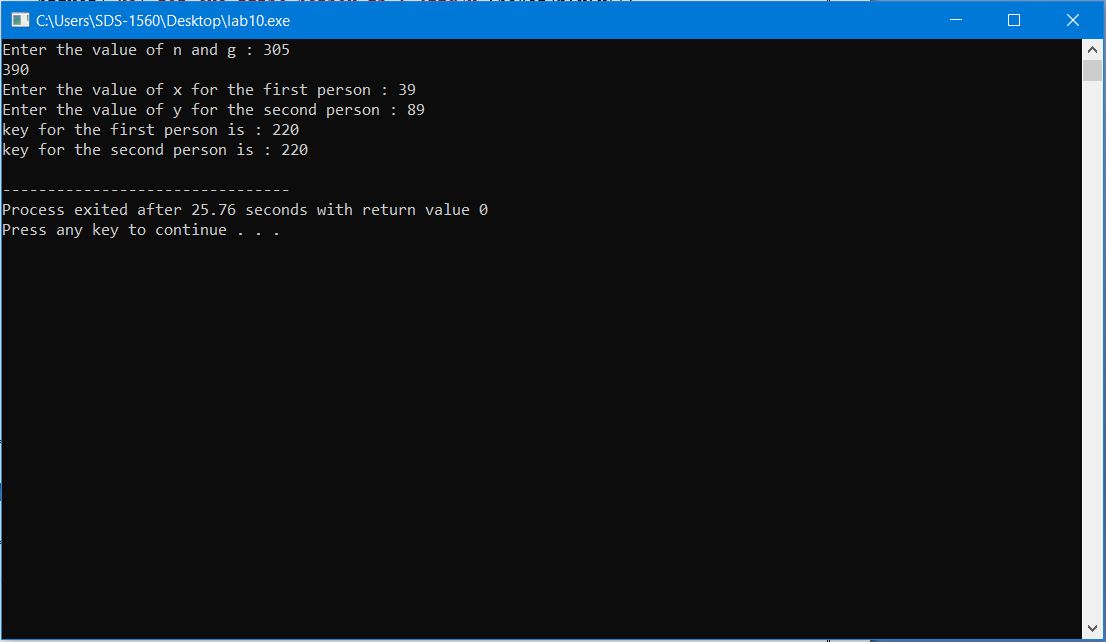
}

#### /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* OUTPUT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/



**LAB (TASK) 10**

Q) Run the above program and generate the desired output.



### LAB ACTIVITY NO 11:

### Vernam Cipher

**Gilbert Sandford Vernam** (3 April 1890 – 7 February 1960) was a [WorcesterPolytechnic Institute](https://en.wikipedia.org/wiki/Worcester_Polytechnic_Institute) 1914 graduate and [AT&T](https://en.wikipedia.org/wiki/AT%26T_Corporation) [Bell Labs](https://en.wikipedia.org/wiki/Bell_Labs) engineer who, in 1917, invented an additive [polyalphabetic](https://en.wikipedia.org/wiki/Polyalphabetic_cipher) [stream cipher](https://en.wikipedia.org/wiki/Stream_cipher) and later co-invented an automated [one-time pad](https://en.wikipedia.org/wiki/One-time_pad) [cipher.](https://en.wikipedia.org/wiki/Cipher) Vernam proposed a teleprinter cipher in which a previously prepared [key,](https://en.wikipedia.org/wiki/Key_(cryptography)) kept on[paper tape,](https://en.wikipedia.org/wiki/Paper_tape) is combined character by character with the [plaintext](https://en.wikipedia.org/wiki/Plaintext) message to produce the [ciphertext](https://en.wikipedia.org/wiki/Ciphertext). To decipher the ciphertext, the same key would be again combined character by character, producing the [plaintext](https://en.wikipedia.org/wiki/Plaintext). Vernam later worked for the [Postal Telegraph Company,](https://en.wikipedia.org/wiki/Postal_Telegraph_Company) and became an employee of [WesternUnion](https://en.wikipedia.org/wiki/Western_Union) when that company acquired Postal in 1943. His later work was largely with automatic switching systems for [telegraph](https://en.wikipedia.org/wiki/Telegraphy)networks.

In modern terminology, a **Vernam cipher** is a symmetrical [stream cipher](https://en.wikipedia.org/wiki/Stream_cipher) in which the plaintext is combined with a random or [pseudorandom](https://en.wikipedia.org/wiki/Pseudorandom) stream of data (the "keystream") of the same length, to generate the ciphertext, using the [Boolean](https://en.wikipedia.org/wiki/Boolean_algebra_(logic)) ["exclusive or" (XOR)](https://en.wikipedia.org/wiki/Exclusive_or) function. This is symbolised by ⊕ and is represented by the following "[truth table](https://en.wikipedia.org/wiki/Truth_table)", where + represents "true" and − represents "false".

Other names for this function are: Not equal (NEQ), [modulo](https://en.wikipedia.org/wiki/Modular_arithmetic) 2 addition (without 'carry') and modulo 2 subtraction (without 'borrow').

The cipher is reciprocal in that the identical keystream is used both to encipher plaintext to ciphertext and to decipher ciphertext to yield the original plaintext:

Plaintext ⊕ Key = Ciphertext and:

Ciphertext ⊕ Key = Plaintext

If the keystream is truly random and used only once, this is effectively a [one-timepad.](https://en.wikipedia.org/wiki/One-time_pad) Substituting pseudorandom data generated by a [cryptographically secure pseudo-random number generator](https://en.wikipedia.org/wiki/Cryptographically_secure_pseudo-random_number_generator) is a common and effective construction for a streamcipher.

**One-time pad** (**OTP**)

The **one-time pad** (**OTP**) is a[n encryption](https://en.wikipedia.org/wiki/Encryption) technique that cannot be [cracked](https://en.wikipedia.org/wiki/Cryptanalysis) if used correctly. It is a special implementation of Vernam cipher, in this technique, a [plaintext](https://en.wikipedia.org/wiki/Plaintext)is paired with a random secret [key](https://en.wikipedia.org/wiki/Key_(cryptography)) (also referred to as *a one-time pad*). Then, each bit or character of the plaintext is encrypted by combining it with the corresponding bit or character from the pad using [modular addition](https://en.wikipedia.org/wiki/Modular_addition). If the key is truly [random,](https://en.wikipedia.org/wiki/Random) is at least as long as the plaintext, is never reused in whole or in part, and is kept completely [secret,](https://en.wikipedia.org/wiki/Secret) then the resulting [ciphertext](https://en.wikipedia.org/wiki/Ciphertext) will be impossible to decrypt orbreak.

#### Practical No. 8: Program in C++ for Vernam Cipher

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Coding \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

#include<iostream>#include <string>

using namespace std; class vernam

{

public:

string s,k;

charenc[1000],dec[1000]; vernam(string a, string b)

{

s = a; k = b;

}

voidencrypt()

{

inti,j=0; for(i=0;i<s.size();i++)

{

enc[i] = s[i]^k[j]; j++;

if(j>=k.size())

{

j =0;

}

}

}

voiddecrypt()

{

inti,j=0; for(i=0;i<s.size();i++)

{

dec[i] = enc[i]^k[j]; j++;

if(j>=k.size())

{

j =0;

}

}

}

voidprintenc()

{

inti; char c;

for(i=0;i<s.size();i++)

{

c = enc[i]%74 + 48; cout<<c;

}

cout<<endl;

}

voidprintdec()

{

inti; for(i=0;i<s.size();i++)

{

cout<<dec[i];

}

cout<<endl;

}

};

intmain()

{

string s,k;

cout<<"Enter the Plain Text Message"<<endl; getline(cin,s);

cout<<"Enter the Key"<<endl; getline(cin,k);

vernam v(s,k);

v.encrypt();

cout<<"Encrypted Message : "; v.printenc();

cout<<endl;

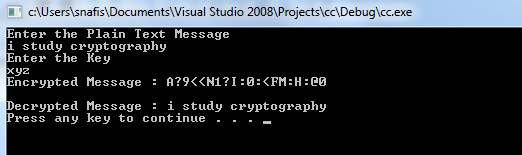
v.decrypt();

cout<<"Decrypted Message : "; v.printdec();

system ("pause"); return 0;

}

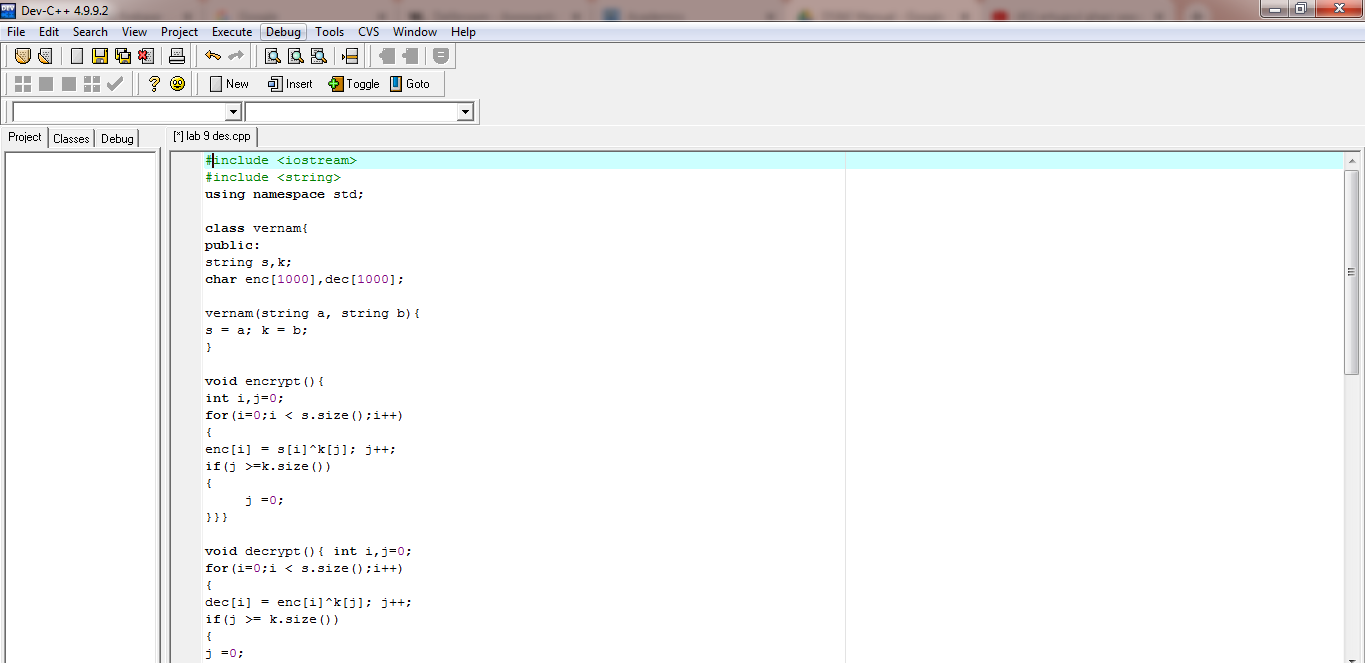
#### /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* OUTPUT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

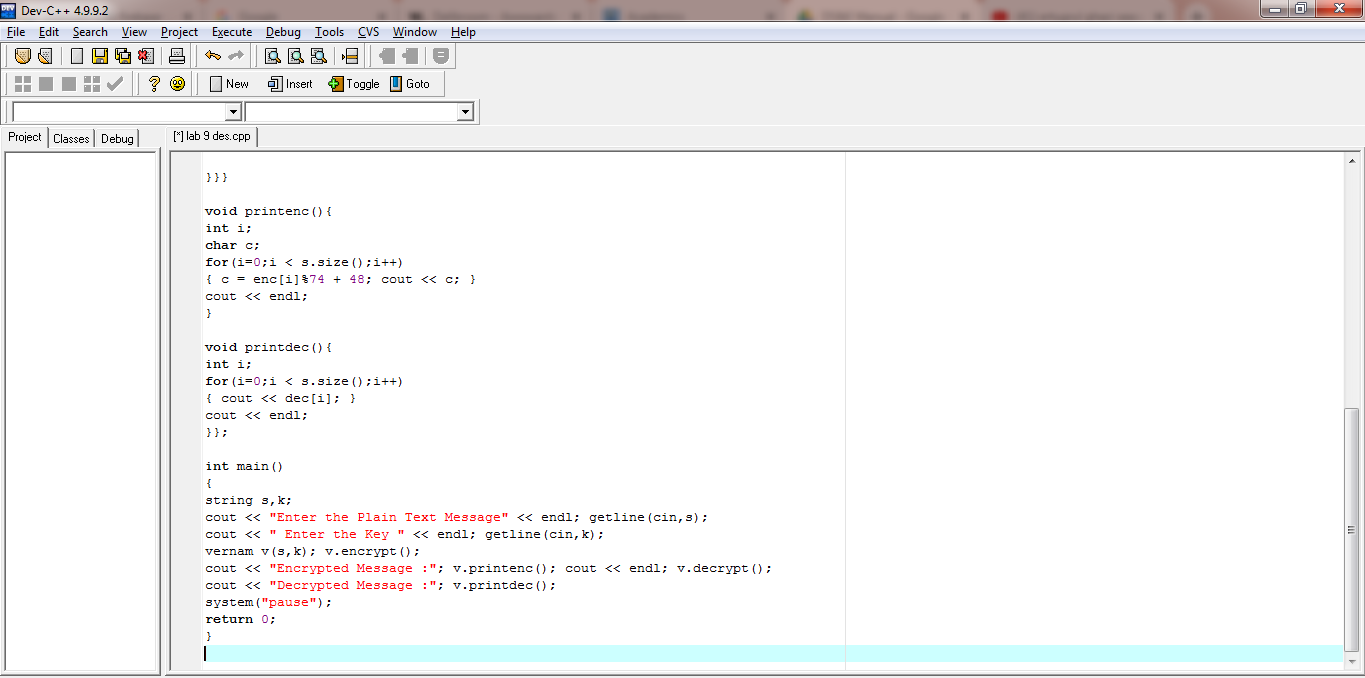


**Lab Task**

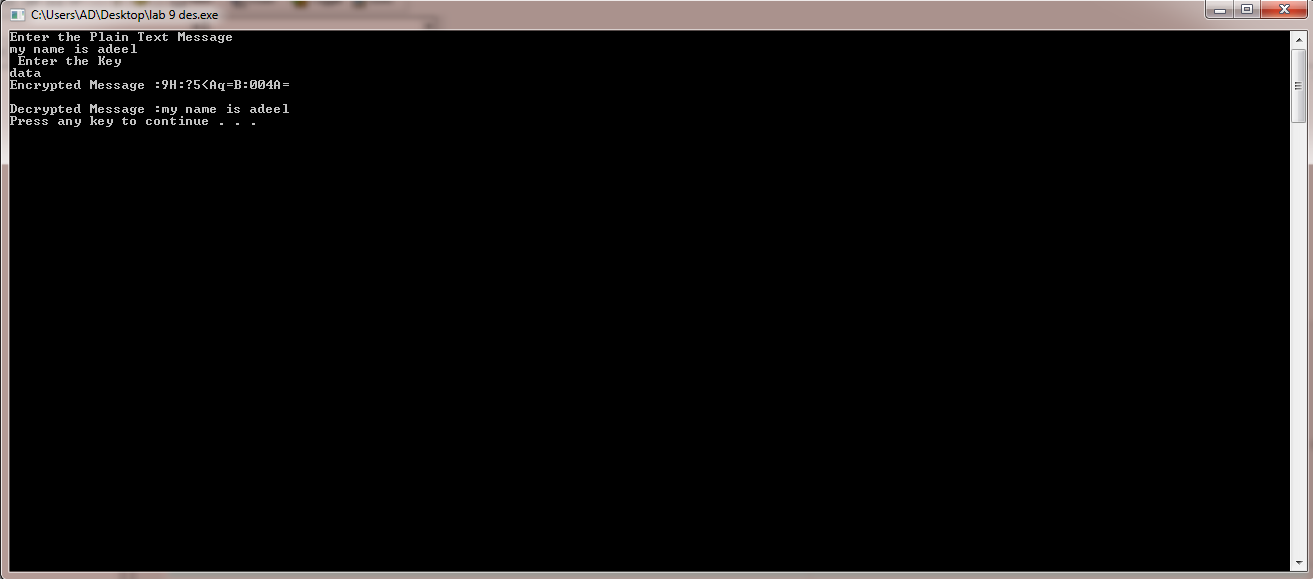
Run the above program and generate the desired output.

**Lab Task 11**

**Code: **

****

**Output:**

****

**LAB ACTIVITY NO 12:**

**RSA**[**Algorithm**](https://simple.wikipedia.org/wiki/Algorithm)**:**

**RSA** is the [algorithm](https://simple.wikipedia.org/wiki/Algorithm) used by modern computers to [encrypt](https://simple.wikipedia.org/wiki/Encryption) and decrypt messages. It is an asymmetric cryptographic algorithm. Asymmetric means that there are two different [keys.](https://simple.wikipedia.org/wiki/Key_(cryptography)) This is also called [*public key cryptography*](https://en.wikipedia.org/wiki/Public-key_cryptography), because one of them can be given to everyone. The *public key* can be shared with everyone, whereas the *private key* must be kept secret. In RSA cryptography, both the public and the private keys can encrypt a message; the opposite key from the one used to encrypt a message is used to decryptit.

It is based on the fact that finding the factors of an integer is hard (the [factoringproblem](https://simple.wikipedia.org/w/index.php?title=Factoring_problem&amp;action=edit&amp;redlink=1)). RSA stands for [Ron Rivest](https://en.wikipedia.org/wiki/Ron_Rivest), [Adi Shamir](https://en.wikipedia.org/wiki/Adi_Shamir) and [Leonard Adleman](https://en.wikipedia.org/wiki/Leonard_Adleman), who first publicly described it in 1978. A user of RSA creates and then publishes the product of two large [prime numbers](https://simple.wikipedia.org/wiki/Prime_number), along with an auxiliary value, as their public key. The prime factors must be kept secret. Anyone can use the public key to encrypt a message, but with currently published methods, if the public key is large enough, only someone with knowledge of the prime factors can feasibly decode themessage.

To generate the encryption and decryption keys, we can proceed as follows. The keys for the RSA algorithm are generated the following way:

1. Choose two distinct [prime numbers](https://en.wikipedia.org/wiki/Prime_number) *p* and*q*.
2. Compute *n* =*p.q*.
3. Compute φ(*n*) = φ(*p*)φ(*q*) = (*p* − 1)(*q* − 1) = *n* − (*p* + *q* − 1), where φ is [Euler's totient function](https://en.wikipedia.org/wiki/Euler%27s_totient_function). This value is keptprivate.
4. Choose an integer *e* such that 1 <*e* <φ(*n*) and [gcd](https://en.wikipedia.org/wiki/Greatest_common_divisor)(*e*, φ(*n*)) = 1; i.e., *e* and φ(*n*) are[coprime.](https://en.wikipedia.org/wiki/Coprime)
5. Determine *d* as *de* mod φ(*n*) = 1; i.e., *d* is the [modular multiplicative inverse](https://en.wikipedia.org/wiki/Modular_multiplicative_inverse) of *e* (modulo φ(*n*)) and keep d assecret
   * Public key = (e,n) *e* is released as the public keyexponent.
   * Private key = (d,n) *d* is kept as the private keyexponent.
   * C = Me (mod n) and transmits C as ciphertext and M is Plaintext or Message
   * Decrypts by calculating M = Cd (modn).

#### Practical No. 9: Program in C++ for RSA algorithm taking p and q randomly

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Coding \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

#include <iostream>#include <stdlib.h>#include <math.h>#include <string.h>

using namespace std;

longintgcd(long inta, long intb)

{

if(a ==0)

returnb; if(b ==0)

returna; return gcd(b,a%b);

}

longintisprime(long inta)

{

inti;

for(i = 2; i< a; i++){ if((a % i) == 0)

return0;

}

return1;

}

longintencrypt(char ch, long intn, long inte)

{

inti;

longinttemp = ch; for(i = 1; i< e; i++)

temp = (temp \* ch) % n; return temp;

}

chardecrypt(long intch, long intn, long intd)

{

inti;

longinttemp = ch; for(i = 1; i< d; i++)

ch =(temp \* ch) % n; return ch;

}

intmain()

{

longinti, len;

longintp, q, n, phi, e, d, cipher[50]; char text[50];

cout<<"Enter the text to be encrypted: "; cin.getline(text, sizeof(text));

len = strlen(text); do {

p = rand() % 30;

} while (!isprime(p));

do{

q = rand() % 30;

} while (!isprime(q)); n = p \* q;

phi = (p - 1) \* (q - 1); do {

e = rand() % phi;

} while (gcd(phi, e) != 1); do {

d = rand() % phi;

} while (((d \* e) % phi) != 1);

cout<<"Two prime numbers (p and q) are: " << p <<" and " << q

<<endl;

cout<<"n(p \* q) = " << p <<" \* " << q <<" = " << p\*q <<endl; cout<<"(p - 1) \* (q - 1) = "<< phi <<endl;

cout<<"Publickey(n, e): (" << n <<", " << e <<")\n";

cout<<"Private key (n, d): (" << n <<", " << d <<")\n"; for (i = 0; i<len; i++)

cipher[i] = encrypt(text[i], n, e); cout<<"Encrypted message: ";

for(i = 0; i<len; i++) cout<< cipher[i];

for(i = 0; i<len; i++)

text[i] = decrypt(cipher[i], n, d); cout<<endl;

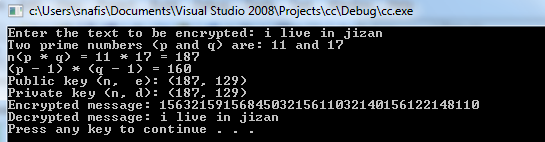
cout<<"Decrypted message: "; for (i = 0; i<len; i++)

cout<< text[i]; cout<<endl;

system ("pause"); return 0;

}

#### /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* OUTPUT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/



**Practical No. 10: Program in C++ for RSA Algorithm by inputting value of two prime numbers**

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Coding \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

#include<stdio.h>#include<conio.h>#include<stdlib.h>#include<math.h>#include<string.h>

long intp,q,n,t,flag,e[100],d[100],temp[100],j,m[100],en[100],i; char msg[100];

intprime(long int); void ce();

longintcd(long int); void encrypt();

voiddecrypt(); void main()

{

system("cls");

printf("\nENTER FIRST PRIME NUMBER\n");

scanf("%d",&p); flag=prime(p); if(flag==0)

{

printf("\nWRONG INPUT\n"); getch();

exit(1);

}

printf("\nENTER ANOTHER PRIME NUMBER\n");

scanf("%d",&q); flag=prime(q); if(flag==0||p==q)

{

printf("\nWRONG INPUT\n"); getch();

exit(1);

}

printf("\nENTER MESSAGE\n"); fflush(stdin); scanf("%s",msg); for(i=0;msg[i]!=NULL;i++) m[i]=msg[i];

n=p\*q;

t=(p-1)\*(q-1);

ce();

printf("\nPOSSIBLE VALUES OF e AND d ARE\n"); for(i=0;i<j-1;i++) printf("\n%ld\t%ld",e[i],d[i]);

encrypt(); decrypt(); getch();

}

intprime(long intpr)

{

inti; j=sqrt((double)pr); for(i=2;i<=j;i++)

{

if(pr%i==0) return 0;

}

return1;

}

voidce()

{

intk; k=0;

for(i=2;i<t;i++)

{

if(t%i==0) continue; flag=prime(i);

if(flag==1&&i!=p&&i!=q)

{

e[k]=i; flag=cd(e[k]); if(flag>0)

{

d[k]=flag; k++;

}

if(k==99) break;

}

}

}

longintcd(long intx)

{

longintk=1; while(1)

{

k=k+t; if(k%x==0) return(k/x);

}

}

voidencrypt()

{

longintpt,ct,key=e[0],k,len; i=0;

len=strlen(msg); while(i!=len)

{

pt=m[i]; pt=pt-96; k=1;

for(j=0;j<key;j++)

{

k=k\*pt; k=k%n;

}

temp[i]=k; ct=k+96; en[i]=ct; i++;

}

en[i]=-1;

printf("\nTHE ENCRYPTED MESSAGE IS\n");

for(i=0;en[i]!=-1;i++)

printf("%c",en[i]);

}

voiddecrypt()

{

longintpt,ct,key=d[0],k; i=0;

while(en[i]!=-1)

{

ct=temp[i]; k=1;

for(j=0;j<key;j++)

{

k=k\*ct; k=k%n;

}

pt=k+96; m[i]=pt; i++;

}

m[i]=-1;

printf("\nTHE DECRYPTED MESSAGE IS\n");

for(i=0;m[i]!=-1;i++)

printf("%c",m[i]);

}

#### /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* OUTPUT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

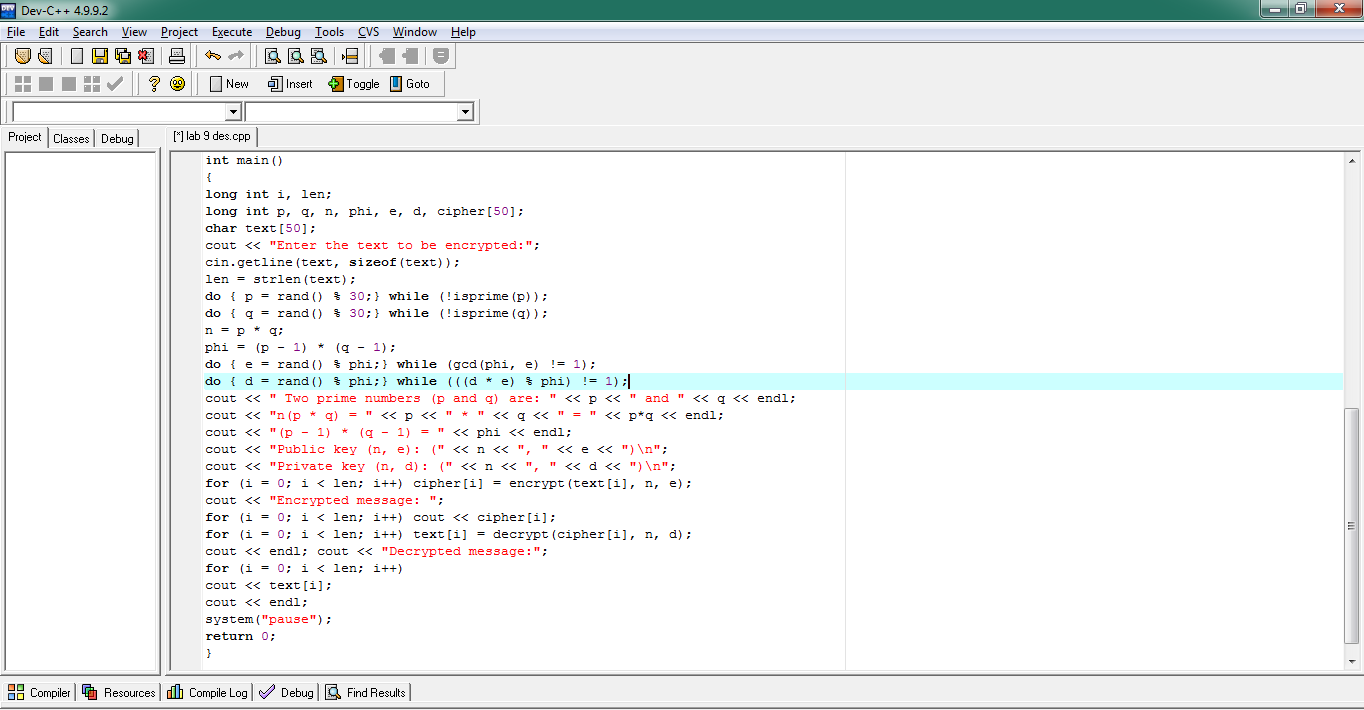
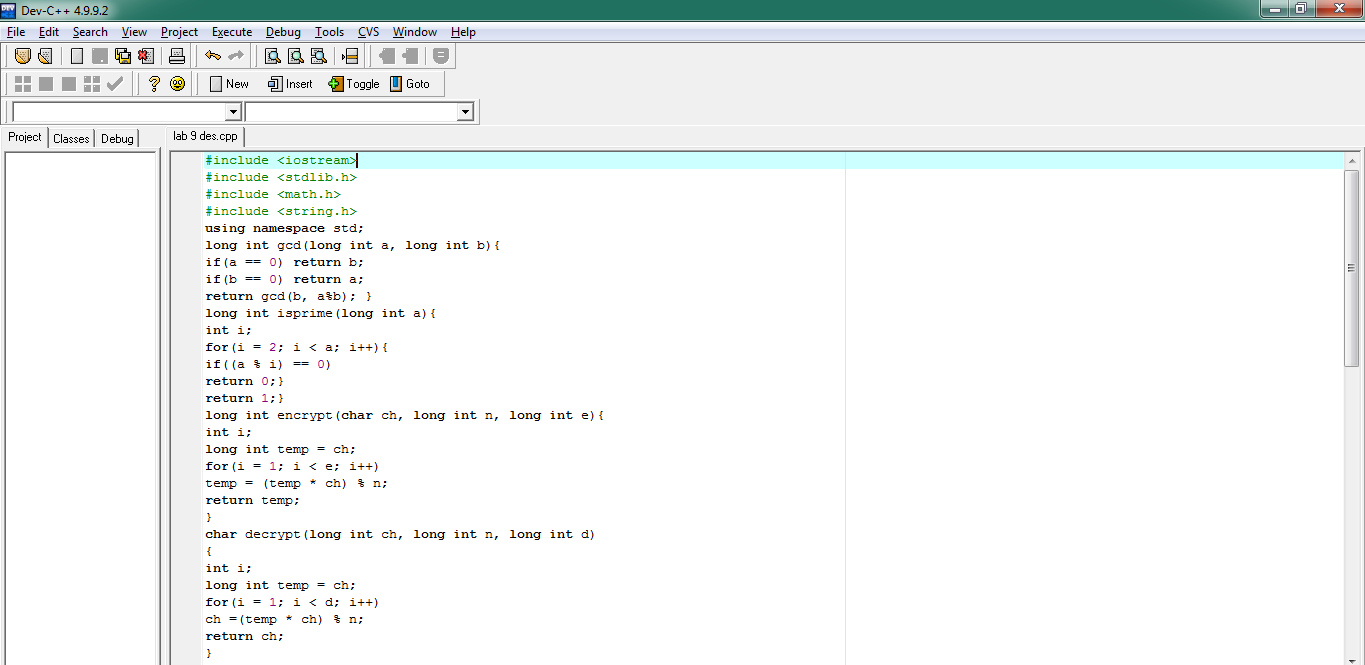
#### ­

**Lab Task**

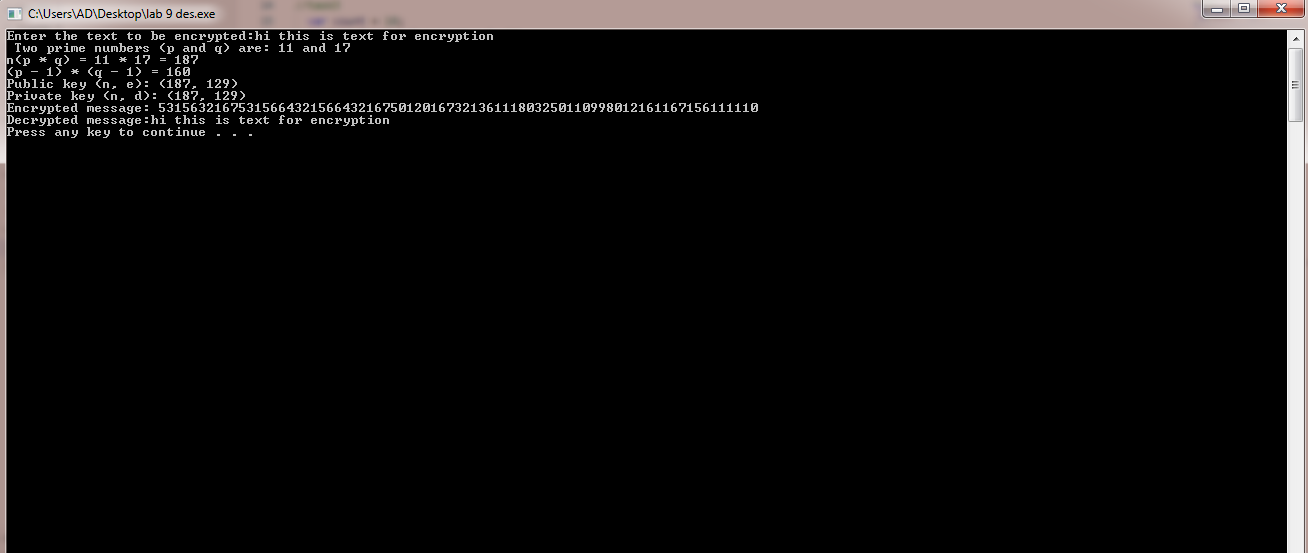
Run the above program and generate the desired output.

**LAB TASK 12**

**Source Code:**

****

**Output:**

****