

Date : 21/08/2020

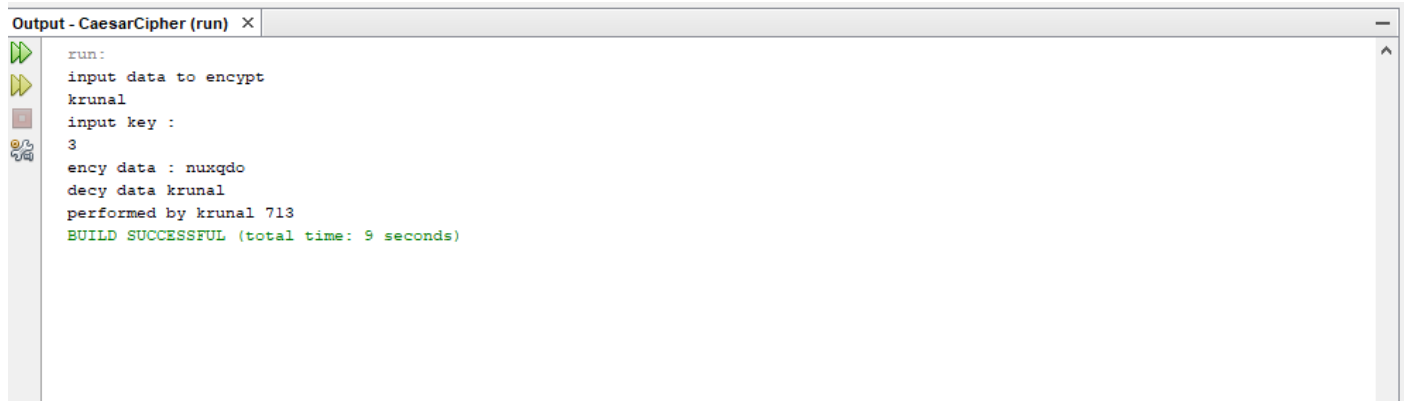
**Practical No 1****AIM:** WAP in Java to implement the following Substitution Cipher Techniques.**CODE****1) Caesar cipher :-**

```
package
javaapplicationins;
import java.io.*;
import
java.util.Scanner;
public class
CaesarCipher
{
    public static void main(String[] args)
    {
        CaesarCipher c=new CaesarCipher();
        Scanner s= new Scanner(System.in);
        System.out.println("Performed by krunal
713"); System.out.println("Input Data to
encrypt"); String str=s.nextLine();
        System.out.println("Input the key");
        int key=s.nextInt();
        String encrypted=c.encrypt(str,key);
        System.out.println("Encrypted
Data:"+encrypted); String
        decrypted=c.decrypt(encrypted,key);
        System.out.println("Decrypted
Data:"+decrypted);
    }
    String encrypt(String str,int key)
    {
        String encrypted="";
        for(int i=0;i<str.length();i++)
        {
            int c=str.charAt(i);
            if(Character.isUpperCase(
c))
            {
                c=c+ke
                y;
                if(c>'Z')
```

```
        {
            c=c-26;
        }
    }
    if(Character.isLowerCase(c))
    {

        c=c+
        key;
        if(c>'
        z'){
            c=c-
            26;
        }
    }
    encrypted +=(char) c;
}
return encrypted;
}
String decrypt(String str,int key)
{
    String decrypted="";
    for(int i=0;i<str.length();i++)
    {
        int c=str.charAt(i);
        if(Character.isUpperCase(
        c))
        {
            c=c-key;
            if(c
            <'A'){
                c=c+26;
            }
        }
        if(Character.isLowerCase(c))
        {
            c=c-
            key;
            if(c
            <'a'){
                c = c + 26;
            }
        }
        decrypted += (char) c;
    }
}
```

```
    return decrypted;
  }
}
```

**Output:**

```
run:
input data to encrypt
krunal
input key :
3
ency data : nuxqdo
decy data krunal
performed by krunal 713
BUILD SUCCESSFUL (total time: 9 seconds)
```

**B) Monoalphabetic Cipher****Program code:**

```
package
javaapplicationins;
import java.io.*;
import java.util.Scanner;

    public class
    MonoalphabeticCipher {

    public static char p[]={ 'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o',

                            'p','q','r','s','t','u','v','w','x','y','z'};

    public static char ch[]={ 'Q','W','E','R','T','Y','U','I','O','P','A','S','D','F','G',

                            'H','J','K','L','Z','X','C','V','B','N','M'};

    public static String doEncryption(String s)
    {

        char c[]=new
        char[(s.length())]; for (int
        i=0;i<s.length();i++)
        {

            for(int j=0;j<26;j++)

            {

                if(p[j]==s.charAt(i))

                {

                    c[i]=ch

                    [j];

                    break;

                }

            }

        }

    }

}
```

```
    }

    }

    return(new String(c));
}

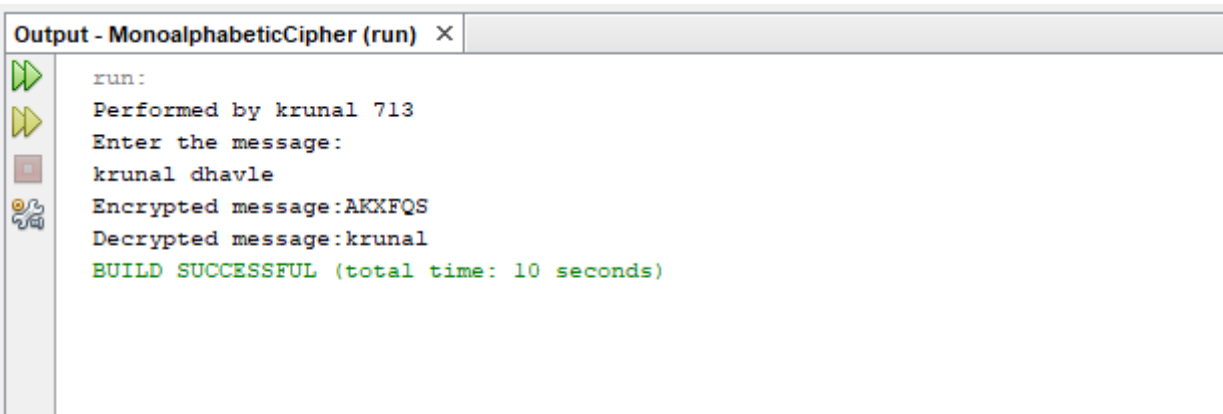
public static String doDecryption(String s)
{
    char pt[]=new
    char[(s.length())]; for (int
    i=0;i<s.length();i++)
    {
        for(int j=0;j<26;j++)
        {
            if(ch[j]==s.charAt(i))
            {
                pt[i]=p[
                j];
                break;
            }
        }
    }

    return(new String(pt));
}

public static void main(String args[])
{
    Scanner sc=new Scanner(System.in);

    System.out.println("Performed by krunal
    713"); System.out.println("Enter the
```

```
message:");  
  
String en=doEncryption(sc.next().toLowerCase());  
  
System.out.println("Encrypted message:"+en);  
  
System.out.println("Decrypted  
message:"+doDecryption(en));  
  
sc.close();  
  
}  
}
```

**Output:**

```
run:  
Performed by krunal 713  
Enter the message:  
krunal dhavle  
Encrypted message:AKXFQS  
Decrypted message:krunal  
BUILD SUCCESSFUL (total time: 10 seconds)
```

Date: 02/09/2020

**Practical no 2****AIM:** Write program to implement the following Substitution Cipher Techniques

a)Vernam Cipher b)Playfair Cipher

**Code:****a)Vernam Cipher**

```
import java.util.Scanner;
public class Vernam {
    String encrypt(String str, String pad) {
        String encrypted = "";
        for (int i = 0; i < str.length(); i++) {
            int c = str.charAt(i);
            int p = pad.charAt(i);
            c = (c + p);
            if (c > 'Z') {
                c = c%26;
                c = c+65;
            }
            encrypted += (char) c;
        }
        return encrypted;
    }
    String decrypt(String str, String pad) {
        String decrypted = "";
        for (int i = 0; i < str.length(); i++) {
            int c = str.charAt(i);
            int p = pad.charAt(i);
            c = (c - p)+26;
            if (c < 'A') {
                c = (c%26);
                c = c+65;
            }
            decrypted += (char) c;
        }
        return decrypted;
    }
}
```

```
public static void main(String[] args) {  
    System.out.println("performed by krunal 713");  
    System.out.println("----*--Encrypting string using Vernam Cipher--*----");  
    Vernam v = new Vernam();  
    Scanner s = new Scanner(System.in);  
    System.out.println("Input Data in Uppercase to encrypt:");  
    String str = s.nextLine();  
    System.out.println("Input the Pad in Uppercase");  
    String pad = s.nextLine();  
    String encrypted = v.encrypt(str, pad);  
    System.out.println("Encrypted Data :" + encrypted);  
    String decrypted = v.decrypt(encrypted, pad);  
    System.out.println("Decrypted Data:" + decrypted);  
}  
}
```

**Output:**

```
performed by krunal 713  
----*--Encrypting string using Vernam Cipher--*----  
Input Data in Uppercase to encrypt:  
HELLO  
Input the Pad in Uppercase  
WORLD  
Encrypted Data :DSCWR  
Decrypted Data:HELLO
```



**b)Playfair Cipher**

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.Arrays;

public class PlayFair {

    public static char keymat[][] = new char[5][5];
    public static String trans = "J";
    public static char subs = 'X';

    private static int decrem(int pos) {
        if (pos < 0) {
            return pos + 5;
        } else {
            return pos;
        }
    }

    private static int[] srch(char c) {
        int i, j;
        int[] pos = new int[2];
        for (i = 0; i < 5; i++) {
            for (j = 0; j < 5; j++) {
                if (keymat[i][j] == c) {
                    pos[0] = i;
                    pos[1] = j;
                    break;
                }
            }
        }
        return pos;
    }

    private static String encrypt(char c1, char c2) {
```

```
int[] pos1 = new int[2];
int[] pos2 = new int[2];
String frag = "";
pos1 = srch(c1);
pos2 = srch(c2);
if (pos1[0] == pos2[0]) { //condition for same row
    c1 = keymat[pos1[0]][(pos1[1] + 1) % 5];
    c2 = keymat[pos2[0]][(pos2[1] + 1) % 5];
    frag = ("" + c1 + c2 + "");
} else if (pos1[1] == pos2[1]) { //condition for same column
    c1 = keymat[(pos1[0] + 1) % 5][pos1[1]];
    c2 = keymat[(pos2[0] + 1) % 5][pos2[1]];
    frag = ("" + c1 + c2 + "");
} else { //condition for different row & column
    c1 = keymat[pos2[0]][pos1[1]];
    c2 = keymat[pos1[0]][pos2[1]];
    frag = ("" + c1 + c2 + "");
}
return frag;
}
```

```
private static String decrypt(char c1, char c2) {
    int[] pos1 = new int[2];
    int[] pos2 = new int[2];
    String frag = "";
    pos1 = srch(c1);
    pos2 = srch(c2);
    if (pos1[0] == pos2[0]) { //condition for same row
        c1 = keymat[pos1[0]][decrem(pos1[1] - 1) % 5];
        c2 = keymat[pos2[0]][decrem(pos2[1] - 1) % 5];
        frag = ("" + c1 + c2 + "");
    } else if (pos1[1] == pos2[1]) { //condition for same column
        c1 = keymat[decrem(pos1[0] - 1) % 5][pos1[1]];
        c2 = keymat[decrem(pos2[0] - 1) % 5][pos2[1]];
        frag = ("" + c1 + c2 + "");
    } else { //condition for different row & column
```

```
        c1 = keymat[pos2[0]][pos1[1]];
        c2 = keymat[pos1[0]][pos2[1]];
        frag = (" " + c1 + c2 + " ");
    }
    return frag;
}

public static void main(String[] args) throws IOException {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    String key;
    int p = 0, k = 0, c = 65;
    System.out.print("Enter Key:\t");
    key = br.readLine();
    for (int i = 0; i < 5; i++) {
        for (int j = 0; j < 5; j++) {
            if (p < key.length()) {
                keymat[i][j] = key.charAt(p);
                p++;
            } else {
                if ((char) c == 'J') {
                    c++;
                }
                for (; k < key.length(); k++) {
                    if ((char) c == key.charAt(k)) {
                        k = 0;
                        c++;
                    }
                    k++;
                }
                keymat[i][j] = (char) c;
                c++;
                k = 0;
            }
        }
    }
    System.out.println("\nMatrix of characters:");
}
```

```
for (int i = 0; i < 5; i++) {
    for (int j = 0; j < 5; j++) {
        System.out.print(keymat[i][j] + "\t");
    }
    System.out.println();
}
String etext = "", dtext = "";
System.out.print("\nEnter Text: \t");
String text = br.readLine();
text = text.toUpperCase();
text = text.replaceAll("\\s", ""); //removes whitespaces
text = text.replace(trans, "I"); //replaces J with I
text = text.replaceAll("([A-Z])\\1+", "$1" + subs + "$1");
if (text.length() % 2 != 0) {
    text += subs;
}
char[] PTC = text.toCharArray();
System.out.println("Padded Text:\t" + text);
for (int i = 0; i < text.length(); i += 2) {
    etext += encrypt(PTC[i], PTC[i + 1]);
}
System.out.println("Encrypted Text:\t" + etext);
char[] OTC = etext.toCharArray();
System.out.println("P: " + Arrays.toString(OTC));
for (int i = 0; i < etext.length(); i += 2) {
    dtext += decrypt(OTC[i], OTC[i + 1]);
}
System.out.println("Decrypted Text:\t" + dtext);
System.out.println("Performed by: 713 Krunal Dhavle");
}}
```

**Output**

```
Enter Key:      CONNECT

Matrix of characters:
C      O      N      N      E
C      T      A      B      D
F      G      H      I      K
L      M      P      Q      R
S      U      V      W      X

Enter Text:      KRUNAL
Padded Text:      KRUNAL
Encrypted Text: RXOVPC
P: [R, X, O, V, P, C]
Decrypted Text: KRUNAL
Performed by: 713 Krunal Dhavle
```

Date: 28/08/2020

**Practical no 3****AIM:** Write program to implement the following Transposition Cipher Techniques

- a) Rail Fence Cipher    b) Simple Columnar Technique

**Code:****a) Rail Fence Cipher**

```
import java.util.Scanner;
import java.util.logging.Level;
import java.util.logging.Logger;
public class Rails {
    String Encrypyton(String plainText,int depth) throws Exception{
        int r=depth, len = plainText.length();
        int c= len/depth;
        char mat[][] = new char[r][c];
        int k=0;

        String cipherText="";

        for(int i=0 ; i < c ; i++) {
            for (int j=0; j<r; j++) {
                if(k!=len) {
                    mat[j][i] = plainText.charAt(k++);
                }
            }
        }
        for(int i=0 ;i<r ;i++) {
            for (int j=0; j<c; j++){
                cipherText += mat[i][j];
            }
        }
        return cipherText ;
    }
    String Decryption(String cipherText,int depth)throws Exception{
        int r=depth,len=cipherText.length();
        int c=len/depth;
```

```
char mat[][]=new char[r][c];
int k=0;
String plainText="";
for(int i=0;i<r;i++) {
    for(int j=0;j<c;j++){
        mat[i][j] =cipherText.charAt(k++);
    }
}
for (int i=0; i<c ;i++){
    for(int j=0;j<r;j++){
        plainText += mat[j][i];

    }
}
return plainText ;
}

public static void main(String[] args) {
    try {
        System.out.println("INS Practical Performed by krunal 713");
        System.out.println("----*--Encrypting string using RailFence cipher--*----");
        Rails rf = new Rails();
        Scanner scn = new Scanner(System.in);
        int depth;
        String plainText,cipherText,decryptedText;
        System.out.println("Enter Plain Text");
        plainText=scn.nextLine();
        System.out.println("Enter depth for Encryption:");
        depth=scn.nextInt();
        while(plainText.length()%depth!=0){
            plainText+='X';
        }
        cipherText=rf.Encrypytion(plainText, depth);
        System.out.println("Encrypted text is:\n" + cipherText);
        decryptedText=rf.Decryption(cipherText, depth);
        decryptedText=decryptedText.replace("X","");
        System.out.println("Decrypted text is :\n"+decryptedText);
    }catch (Exception ex){
```

```
Logger.getLogger(Rails.class.getName()).log(Level.SEVERE,null,ex);  
}  
}  
}
```

**Output:**

```
INS Practical Performed by krunal 713  
----*--Encrypting string using RailFence cipher--*----  
Enter Plain Text  
kunal  
Enter depth for Encryption:  
2  
Encrypted text is:  
knluaX  
Decrypted text is :  
kunal
```

```
INS Practical Performed by krunal 713  
----*--Encrypting string using RailFence cipher--*----  
Enter Plain Text  
krunal  
Enter depth for Encryption:  
2  
Encrypted text is:  
kuarnl  
Decrypted text is :  
krunal
```



**b) Simple Columnar Technique**

```
package prac3b;
import java.io.BufferedReader;
import java.io.*;
import java.io.InputStreamReader;
import java.util.logging.Level;
import java.util.logging.Logger;

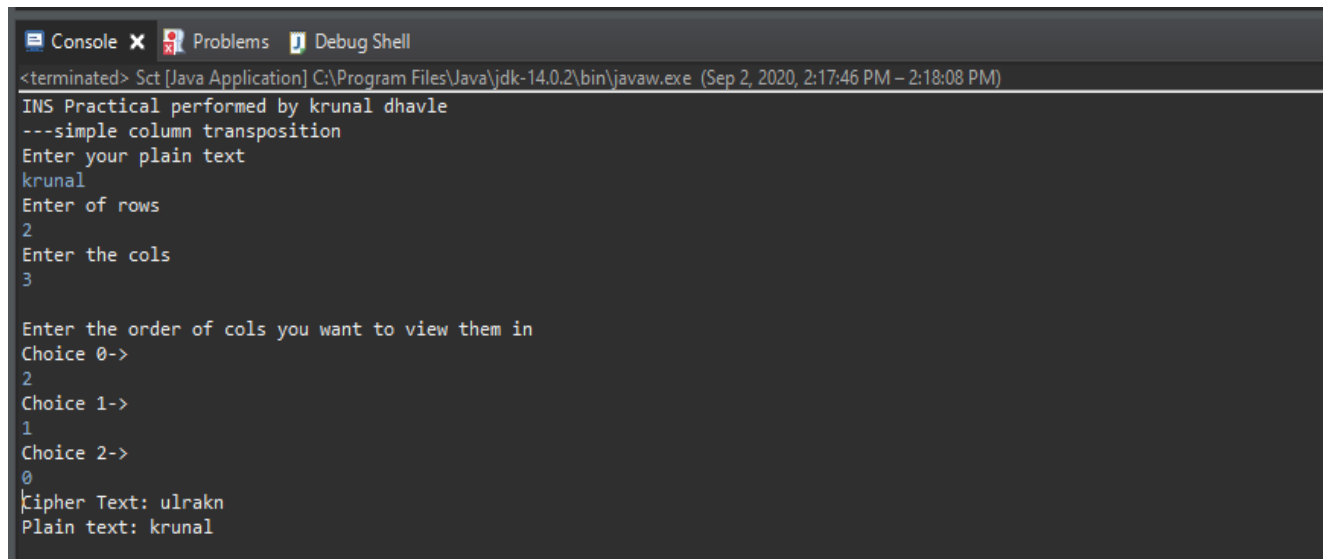
public class Sct {
    public static void main(String[] args) {

        try {

            System.out.println("INS Practical performed by krunal dhavle ");
            System.out.println("---simple column transposition ");
            BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
            System.out.println("Enter your plain text");
            String accept = br.readLine();
            System.out.println("Enter of rows ");
            int r = Integer.parseInt(br.readLine());
            System.out.println("Enter the cols");
            int c = Integer.parseInt(br.readLine());
            int count = 0;
            char table[][] = new char[r][c];
            for (int i = 0; i < r; i++)
            {
                for (int j = 0; j < c; j++)
                {
                    table[i][j] = accept.charAt(count);
                    count++;
                }
            }

            System.out.println("\nEnter the order of cols you want to view them in");
            int choice[] = new int[c];
            for (int k = 0; k < c; k++)
            {
```

```
        System.out.println("Choice " + k + "-> ");
        choice[k] = Integer.parseInt(br.readLine());
    }
    String cipher = "", plain = "";
    for (int j = 0; j < c; j++)
    {
        int k = choice[j];
        for (int i = 0; i < r; i++)
        {
            cipher += table[i][k];
        }
    }
    cipher = cipher.trim();
    System.out.println("Cipher Text: "+cipher);
    char mat[][] = new char[r][c];
    int t = 0;
    for (int j = 0; j < c; j++)
    {
        int k = choice[j];
        for (int i = 0; i < r; i++)
        {
            mat[i][k] = cipher.charAt(t++);
        }
    }
    for (int i = 0; i < r; i++)
    {
        for (int j = 0; j < c; j++)
        {
            plain += mat[i][j];
        }
    }
    plain = plain.trim();
    System.out.println("Plain text: "+plain);
}
catch (IOException ex)
{
    Logger.getLogger(Sct.class.getName()).log(Level.SEVERE, null, ex); } }
```

**Output:**

```
<terminated> Sct [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe (Sep 2, 2020, 2:17:46 PM – 2:18:08 PM)
INS Practical performed by krunal dhavle
---simple column transposition
Enter your plain text
krunal
Enter of rows
2
Enter the cols
3

Enter the order of cols you want to view them in
Choice 0->
2
Choice 1->
1
Choice 2->
0
Cipher Text: ulrakn
Plain text: krunal
```

Date:11/09/2020

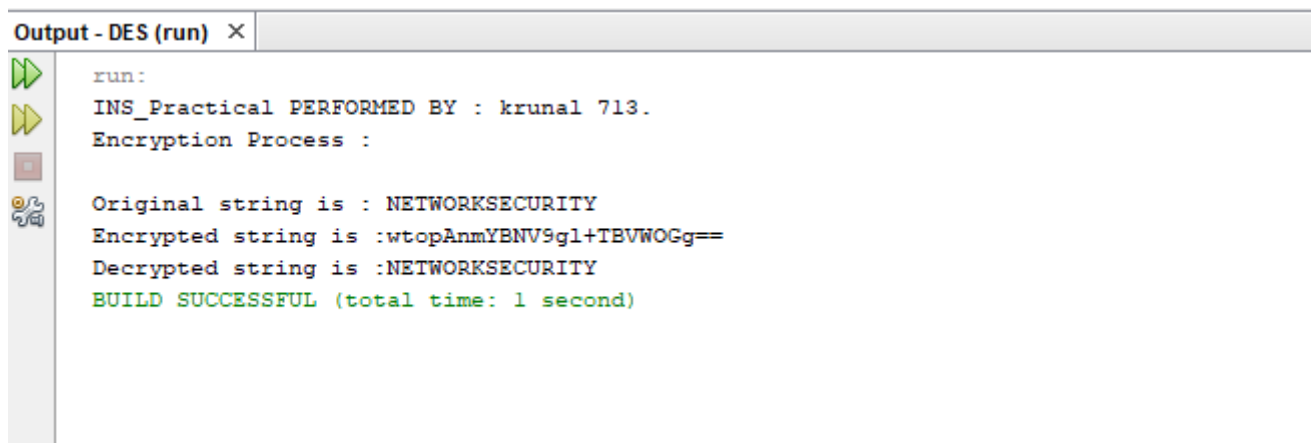
**Practical no 4****AIM:** Write program to encrypt and decrypt strings using

1) DES Algorithm 2) AES Algorithm

**CODE****1) DES Algorithm**

```
import java.util.logging.Level;
import java.util.logging.Logger;
import java.util.Base64;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
public class DES {
    public static SecretKey getSecretEncryptionKey() throws Exception{
        KeyGenerator generator=KeyGenerator.getInstance("DES");
        SecretKey secKey=generator.generateKey();
        return secKey;
    }
    public String encrypt(SecretKey key,String Plaintext) throws Exception{
        byte[] utf8=Plaintext.getBytes();
        Cipher ecipher=Cipher.getInstance("DES");
        ecipher.init(Cipher.ENCRYPT_MODE, key);
        byte[] enc=ecipher.doFinal(utf8);
        Base64.Encoder encoder=Base64.getEncoder();
        String et=encoder.encodeToString(enc);
        return et;
    }
    public String decrypt(SecretKey key,String Ciphertext) throws Exception{
        Base64.Decoder decoder = Base64.getDecoder();
        byte[] dec=decoder.decode(Ciphertext);
        Cipher dcipher=Cipher.getInstance("DES");
        dcipher.init(Cipher.DECRYPT_MODE, key);
        byte[] utf8=dcipher.doFinal(dec);
        return new String(utf8,"UTF8");
    }
}
```

```
}  
public static void main(String[] args){  
    try{  
        System.out.println("INS_Practical PERFORMED BY : krunal 713.");  
        System.out.println("----'--Encrypting string using DES--'----");  
        System.out.println();  
        String message ="NETWORKSECURITY";  
        DES d=new DES();  
        SecretKey key=getSecretEncryptionKey();  
        String Encrypted=d.encrypt(key, message);  
        String Decrypted=d.decrypt(key, Encrypted);  
        System.out.println("Original String is : "+ message);  
        System.out.println("Encrypted String is : "+ Encrypted);  
        System.out.println("Decrypted String is : "+ Decrypted);  
    }catch (Exception ex){  
        Logger.getLogger(DES.class.getName()).log(Level.SEVERE,null,ex);  
    }  
}
```

**Output:**

```
run:  
INS_Practical PERFORMED BY : krunal 713.  
Encryption Process :  
  
Original string is : NETWORKSECURITY  
Encrypted string is :wtopAnmYBNV9gl+TBVWOGg==  
Decrypted string is :NETWORKSECURITY  
BUILD SUCCESSFUL (total time: 1 second)
```

**b) AES CODE**

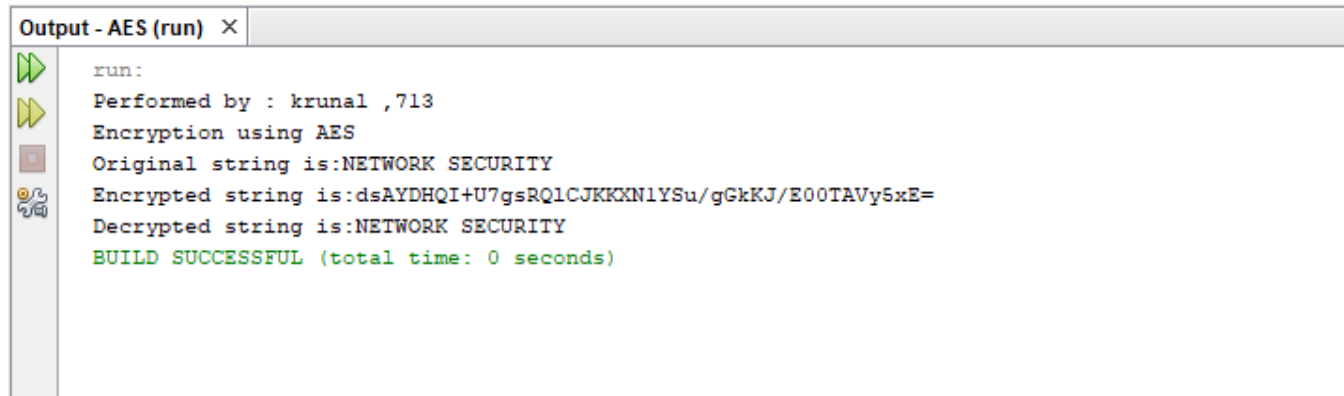
```
package aes;
import java.util.logging.Logger;
import java.util.logging.Level;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;

public class AES {
    public static SecretKey getSecretEncryptionKey() throws Exception{
        KeyGenerator generator = KeyGenerator.getInstance("AES");
        generator.init(128);
        SecretKey secKey= generator.generateKey();
        return secKey;
    }

    public String encrypt(SecretKey key,String Plaintext)throws Exception{
        byte[] utf8= Plaintext.getBytes("UTF8");
        Cipher ecipher= Cipher.getInstance("AES");
        ecipher.init(Cipher.ENCRYPT_MODE,key);
        byte[] enc= ecipher.doFinal(utf8);
        return new sun.misc.BASE64Encoder().encode(enc);
    }

    public String decrypt(SecretKey key,String Ciphertext) throws Exception{
        byte[] dec= new sun.misc.BASE64Decoder().decodeBuffer(Ciphertext);
        Cipher dcipher= Cipher.getInstance("AES");
        dcipher.init(Cipher.DECRYPT_MODE,key);
        byte[] utf8= dcipher.doFinal(dec);
        return new String(utf8, "UTF8");
    }

    public static void main (String[]args) throws Exception
    {
        try{
            System.out.println("Performed by : krunal ,713");
            System.out.println("Encryption using AES");
            String message="NETWORK SECURITY";
            AES d= new AES();
            SecretKey key= getSecretEncryptionKey();
            String Encrypted= d.encrypt(key, message);
            String decrypted = d.decrypt(key,Encrypted);
            System.out.println("Original string is:" +message);
            System.out.println("Encrypted string is:" + Encrypted);
            System.out.println("Decrypted string is:" +decrypted);
        }
        catch(Exception ex){
            Logger.getLogger(AES.class.getName()).log(Level.SEVERE,null,ex) ;
        }
    }
}
```

**Output:**

```
run:
Performed by : krunal ,713
Encryption using AES
Original string is:NETWORK SECURITY
Encrypted string is:dsAYDHQI+U7gsRQ1CJKKXN1YSu/gGkKJ/E00TAVy5xE=
Decrypted string is:NETWORK SECURITY
BUILD SUCCESSFUL (total time: 0 seconds)
```

Date:29/09/2020

**Practical no 5**

**AIM:** Write a program to implement RSA algorithm to perform encryption / decryption of a given string.

**CODE**

```
package prac5;
import java.util.*;
import java.math.*;
public class Rsa {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner sc=new Scanner(System.in);
        int p,q,n,z,d=0,e,i;
        double c;
        BigInteger msgback;
        System.out.println("Enter 1st prime number p");
        p=sc.nextInt();
        System.out.println("Enter 2nd prime number q");
        q=sc.nextInt();
        sc.close();
        n=p*q;
        z=(p-1)*(q-1);
        System.out.println("the value of n = "+n);
        for(e=2;e<z;e++)
        {
            if(gcd(e,z)==1)        // e is for public key exponent
            {
                break;
            }
        }
        System.out.println("the value of e = "+e);
        for(i=0;i<=9;i++)
        {
```



```
        int x=1+(i*z);
        if(x%e==0)    //d is for private key exponent
        {
            d=x/e;
            break;
        }
    }

    System.out.println("the value of d = "+d);
    c=(Math.pow(2,e))%n;

    System.out.println("Encrypted message is : -");
    System.out.println(c);

    BigInteger N = BigInteger.valueOf(n);

    BigInteger C = BigDecimal.valueOf(c).toBigInteger();
    msgback = (C.pow(d)).mod(N);

    System.out.println("Derypted message is : -");
    System.out.println(msgback);
    System.out.println("performed by krunal dhavle 713");
}
static int gcd(int e, int z)
{
    if(e==0)
        return z;
    else
        return gcd(z%e,e);
}
}
```

**Output:**

```
<terminated> Rsa [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe (Sep 29, 2020, 2:34:32 PM – 2:34:36 PM)
Enter 1st prime number p
23
Enter 2nd prime number q
17
the value of n = 391
the value of e = 3
the value of d = 235
Encrypted message is : -
8.0
Derypted message is : -
2
performed by krupal dhavle 713
```

Date:06/10/2020

**Practical no 6**

**AIM:** Write a program to implement the Diffie-Hellman Key Agreement algorithm to generate symmetric keys.

**CODE:-****Method 1:-**

```
package prac6;
import java.util.*;
public class DiffieHellman {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter modulo(p)");
        int p=sc.nextInt();
        System.out.println("Enter primitive root of "+p);
        int g=sc.nextInt();
        System.out.println("Choose 1st key secret");
        int a=sc.nextInt();
        System.out.println("Choose 2nd key secret");
        int b=sc.nextInt();
        sc.close();
        int A = (int)Math.pow(g,a)%p;
        int B = (int)Math.pow(g,b)%p;

        int S_A = (int)Math.pow(B,a)%p;
        int S_B =(int)Math.pow(A,b)%p;

        if(S_A==S_B)
        {
            System.out.println("key1 and key2 matches they can communicate
with each other!!!");

            System.out.println("They share a secret no = "+S_A);
            System.out.println("Performed by krupal dhavle ,713");
        }

        else
        {
            System.out.println("key1 and key2 matches they cannot communicate
with each other!!!");

            System.out.println("Performed by krupal dhavle ,713");
        }
    }
}
```

**Output:**

```
<terminated> DiffieHellman [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe (Sep 29, 2020, 3:02:45 PM – 3:02:56 PM)
Enter modulo(p)
23
Enter primitive root of 23
9
Choose 1st key secret
4
Choose 2nd key secret
3
key1 and key2 matches they can communicate with each other!!!
They share a secret no = 9
Performed by krunal dhavle ,713
```

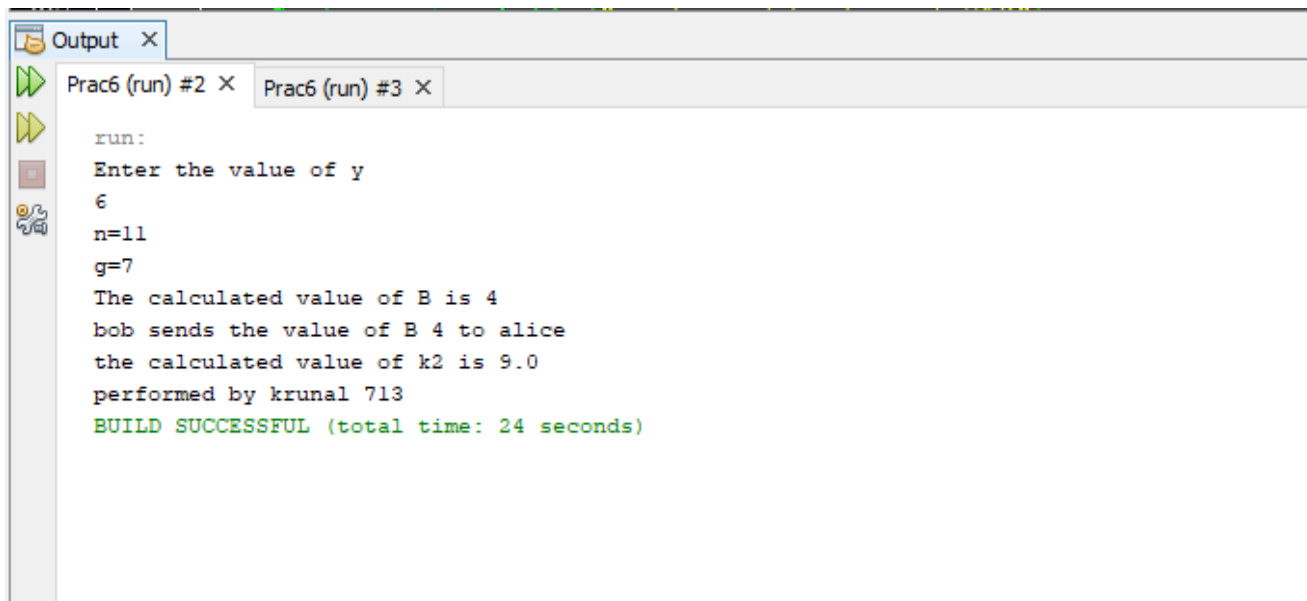
**Method 2 :-****Bob.java**

```
package prac6;
import java.io.*;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Scanner;
public class Bob {

    public static void main(String[] args) throws IOException {
        ServerSocket ss = new ServerSocket(5000);
        Socket s = ss.accept();
        DataInputStream in = new DataInputStream(s.getInputStream());
        int n = in.readInt();
        int g = in.readInt();
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the value of y");
        int y = sc.nextInt();
        System.out.println("n=" +n);
        System.out.println("g=" +g);
        int d =(int)Math.pow(g, y);
        int B =d%n;
        System.out.println("The calculated value of B is " +B);
        System.out.println("bob sends the value of B " +B+ " to alice");
        int A = in.readInt();
        int b = (int)Math.pow(A,y);
        double K2 = b%n;
        System.out.println("the calculated value of k2 is " +K2);
        DataOutputStream out = new DataOutputStream(s.getOutputStream());
        out.writeInt(B);
        System.out.println("performed by krunal 713");
    }
}
```

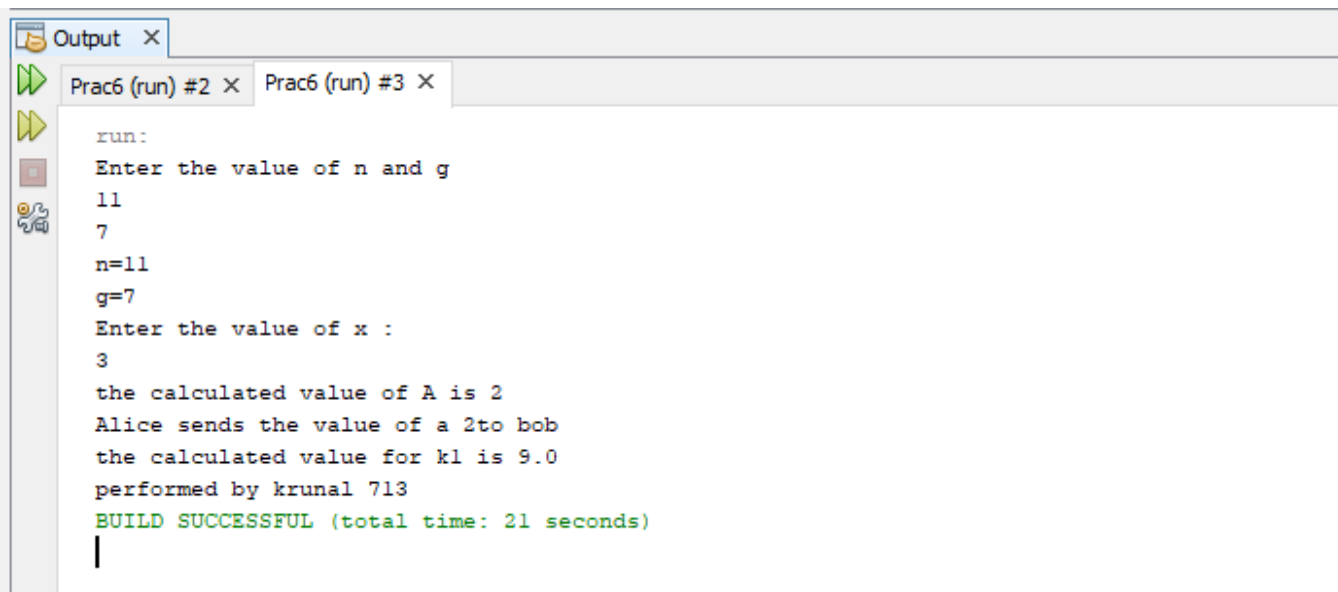
**Alice.java**

```
package prac6;
import java.io.*;
import java.net.Socket;
import java.util.Scanner;
public class Alice {
    public static void main(String[] args) throws IOException {
        Socket cs = new Socket("localhost",5000);
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the value of n and g ");
        int n = sc.nextInt();
        int g = sc.nextInt();
        System.out.println("n=" +n);
        System.out.println("g=" +g);
        DataOutputStream out = new DataOutputStream(cs.getOutputStream());
        out.writeInt(n);
        out.writeInt(g);
        System.out.println("Enter the value of x : ");
        int x = sc.nextInt();
        int c =(int)Math.pow(g,x);
        int A = c%n;
        System.out.println("the calculated value of A is " +A);
        out.writeInt(A);
        System.out.println("Alice sends the value of a " +A + "to bob");
        DataInputStream in = new DataInputStream(cs.getInputStream());
        int B = in.readInt();
        int a = (int)Math.pow(B, x);
        double K1 = a % n;
        System.out.println("the calculated value for k1 is " +K1);
        System.out.println("performed by krunal 713");
    }
}
```

**Output:**

The screenshot shows an IDE output window with a title bar 'Output X'. Below the title bar are two tabs: 'Prac6 (run) #2 X' and 'Prac6 (run) #3 X'. The output text is as follows:

```
run:
Enter the value of y
6
n=11
g=7
The calculated value of B is 4
bob sends the value of B 4 to alice
the calculated value of k2 is 9.0
performed by krunal 713
BUILD SUCCESSFUL (total time: 24 seconds)
```



The screenshot shows an IDE output window with a title bar 'Output X'. Below the title bar are two tabs: 'Prac6 (run) #2 X' and 'Prac6 (run) #3 X'. The output text is as follows:

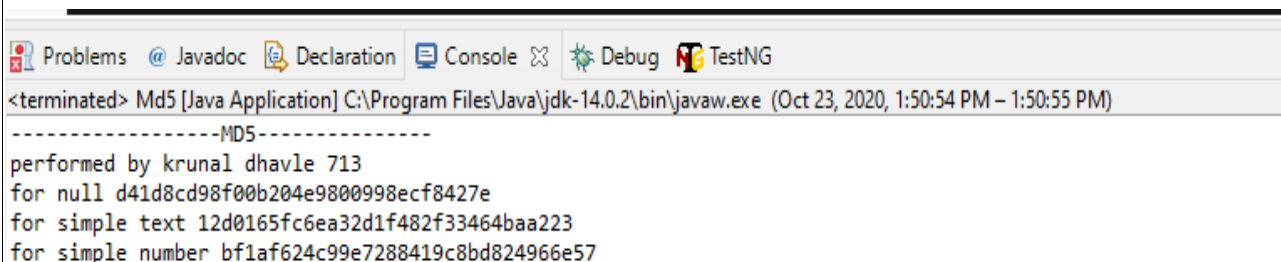
```
run:
Enter the value of n and g
11
7
n=11
g=7
Enter the value of x :
3
the calculated value of A is 2
Alice sends the value of a 2to bob
the calculated value for k1 is 9.0
performed by krunal 713
BUILD SUCCESSFUL (total time: 21 seconds)
|
```

Date:23/10/2020

**Practical no 7****AIM:** Write a program to implement the MD5 algorithm compute the message digest.**Code:-**

```
import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

public class Md5 {
    public static String md5(String input) throws NoSuchAlgorithmException {
        String md5 = null;
        if(null == input)
        {
            return null;
        }
        MessageDigest md = MessageDigest.getInstance("MD5");
        md.update(input.getBytes());
        md5 =new BigInteger(1 , md.digest()).toString(16);
        return md5;
    }
    public static void main(String[] args) throws NoSuchAlgorithmException {
        System.out.println("-----MD5-----");
        System.out.println("performed by krunal dhavle 713");
        System.out.println("for null " +md5(""));
        System.out.println("for simple text " +md5(" krunal dhavle 713 "));
        System.out.println("for simple number " +md5("291999"));
    }
}
```

**Output**

```
<terminated> Md5 [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe (Oct 23, 2020, 1:50:54 PM – 1:50:55 PM)
-----MD5-----
performed by krunal dhavle 713
for null d41d8cd98f00b204e9800998ecf8427e
for simple text 12d0165fc6ea32d1f482f33464baa223
for simple number bf1af624c99e7288419c8bd824966e57
```



Date:16/10/2020

**Practical no 8****AIM:** Write a program to calculate HMAC-SHA1 Signature**Code:-**

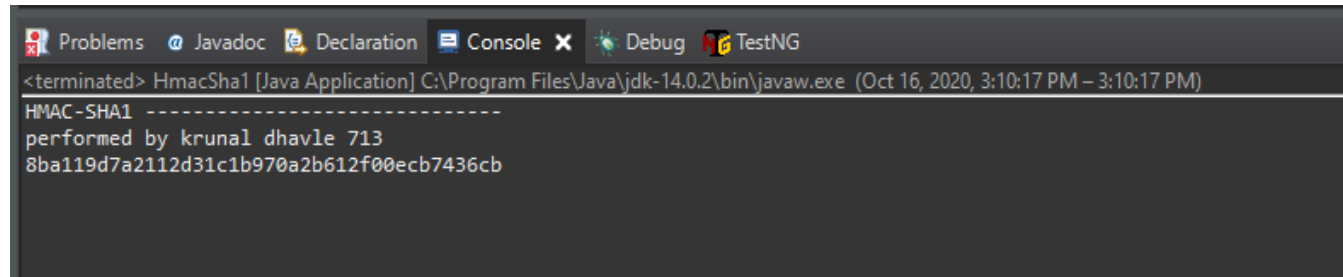
```
package prac8;
import java.util.Formatter;
import javax.crypto.*;
import javax.crypto.spec.SecretKeySpec;

public class HmacSha1 {
    private static String toHexString(byte[] bytes){
        Formatter formatter = new Formatter();
        for(byte b : bytes)
        {
            formatter.format("%02x" , b);
        }
        return formatter.toString();
    }

    public static String calculateHMAC(String data , String key) throws Exception
    {
        SecretKeySpec signingKey = new SecretKeySpec(key.getBytes() , "HmacSHA1");
        Mac mac = Mac.getInstance("HmacSHA1");
        mac.init(signingKey);
        return toHexString(mac.doFinal(data.getBytes()));
    }

    public static void main(String[] args) throws Exception
    {
        String hmac = calculateHMAC("krunal", "dhavle");
        System.out.println("HMAC-SHA1 -----");
        System.out.println("performed by krunal dhavle 713");
    }
}
```

```
        System.out.println(hmac);  
    }  
}
```

**Output:-**

The screenshot shows a Java IDE with a console window open. The console displays the output of a Java application named 'HmacSha1'. The output is as follows:

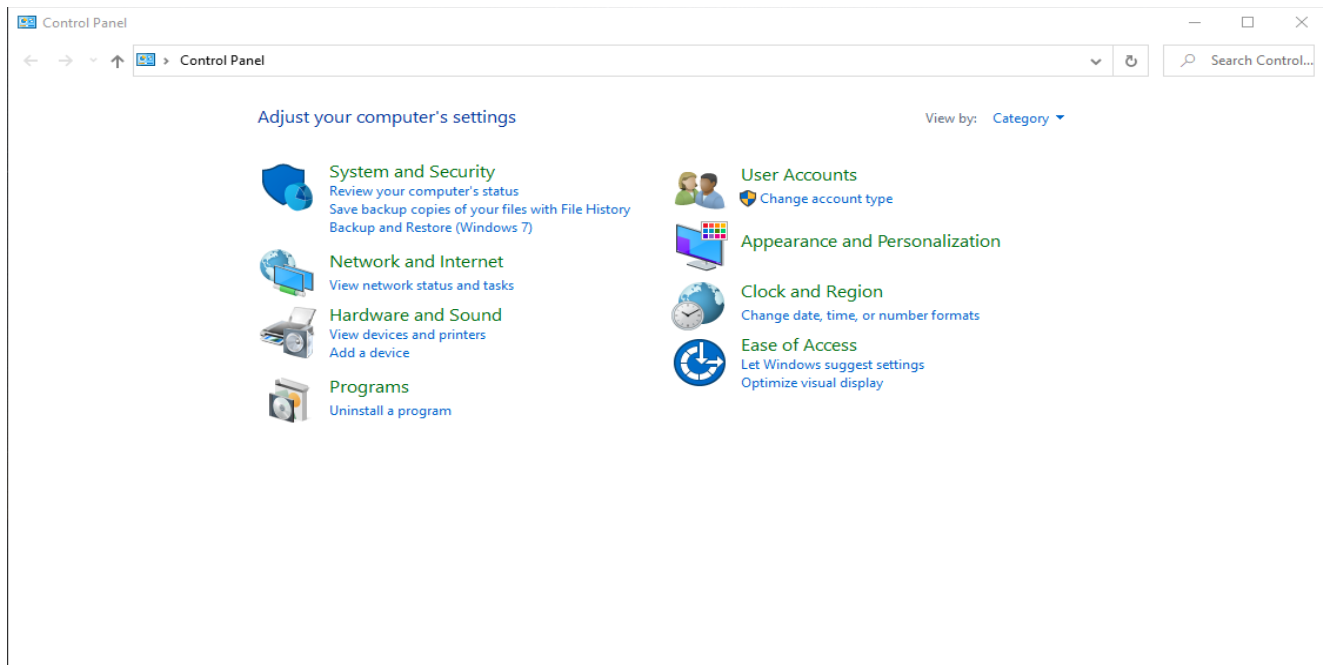
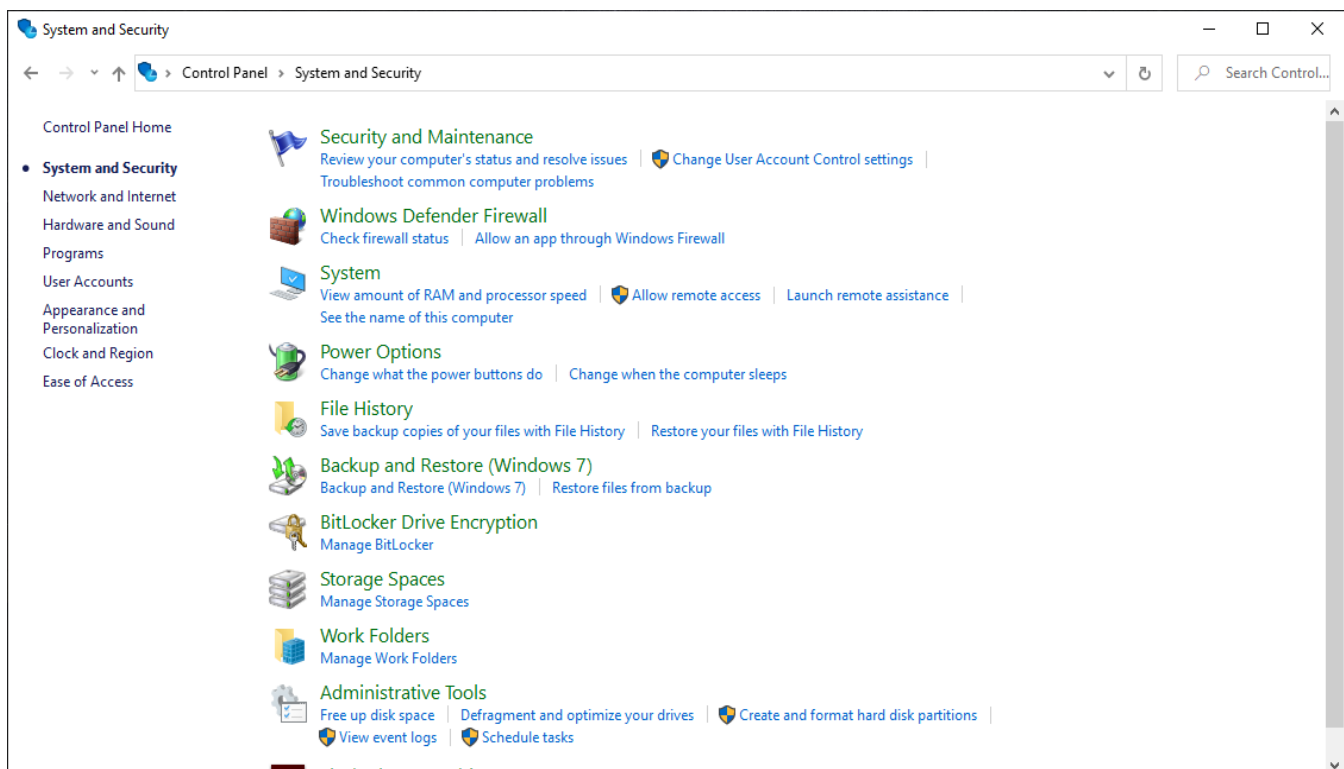
```
<terminated> HmacSha1 [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe (Oct 16, 2020, 3:10:17 PM – 3:10:17 PM)  
HMAC-SHA1 -----  
performed by krunal dhavle 713  
8ba119d7a2112d31c1b970a2b612f00ecb7436cb
```

Date:07/11/2020

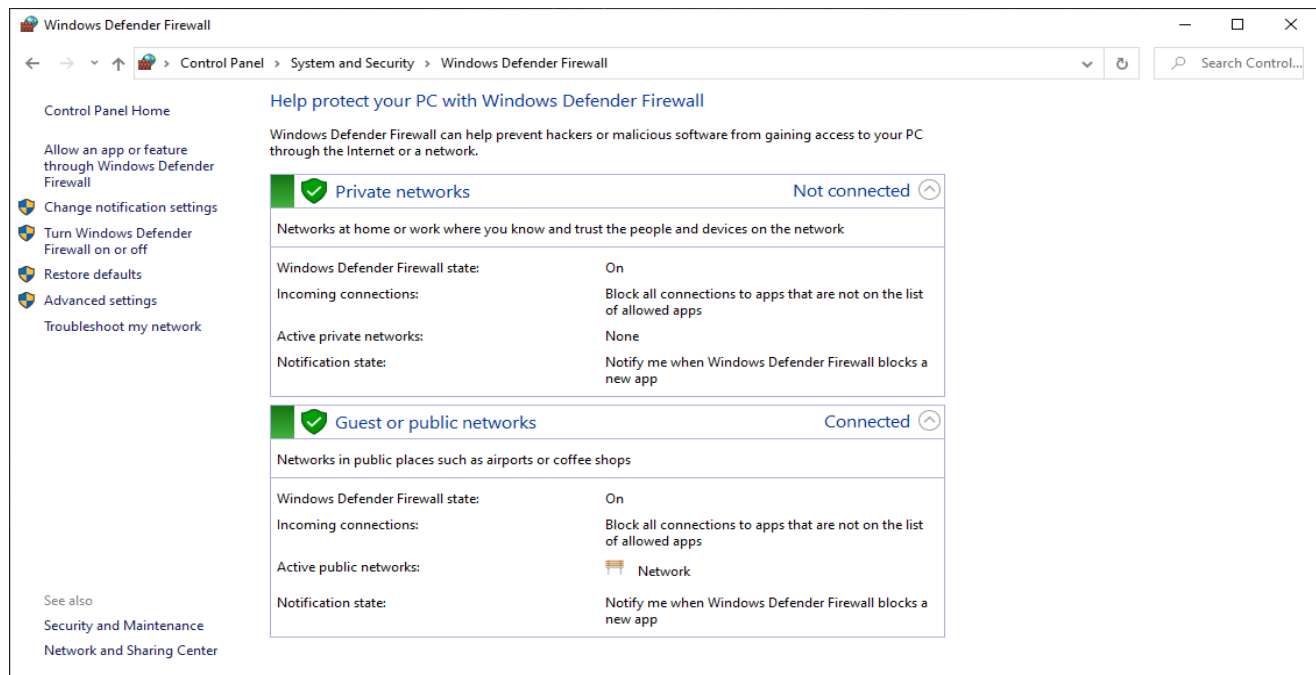
**Practical no 9****AIM:** Configure windows firewall to block

1) A port 2) An Program 3) A Website

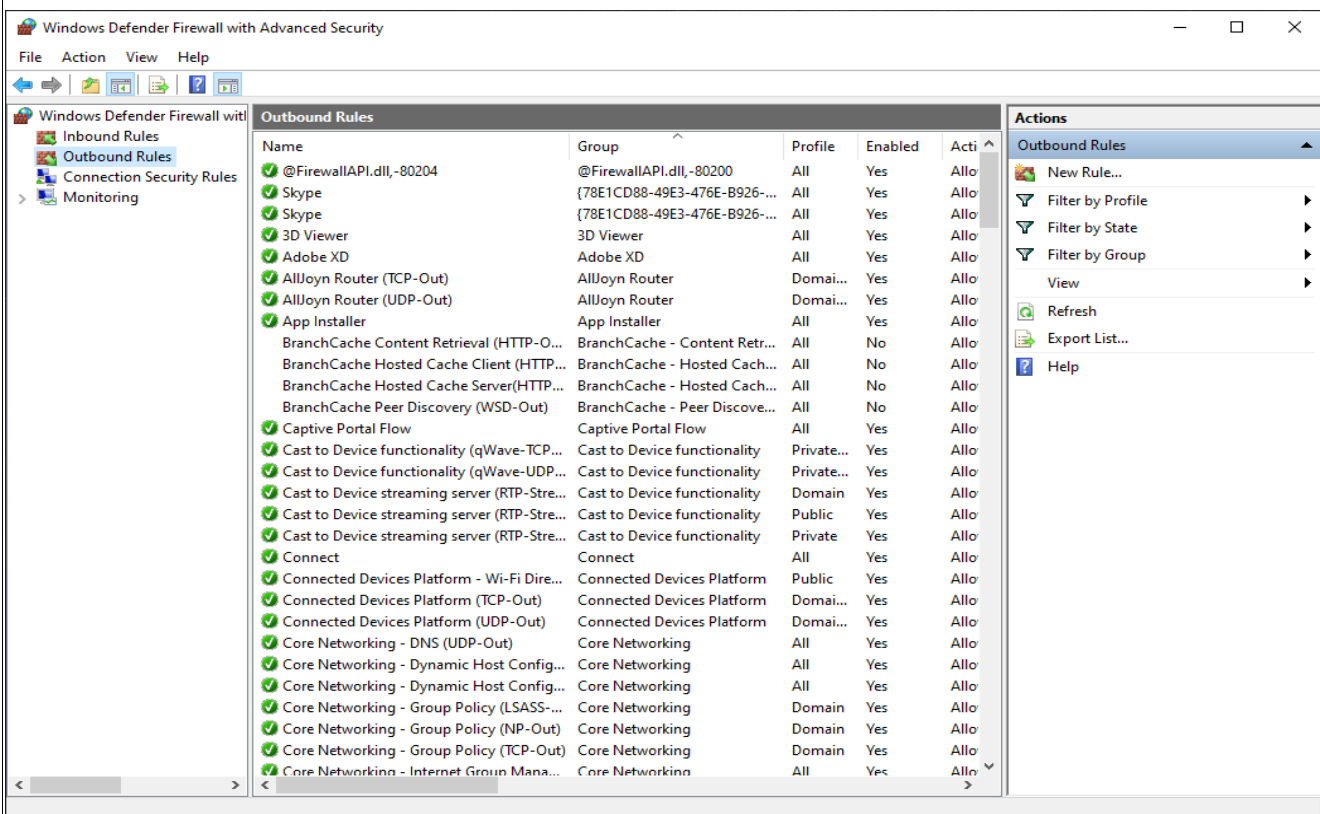
**Different Types of Profiles available/ When does this rule applies****Domain:** Applies when computer is connected to corporate domain**Private:** Applies when computer is connected to a private network location, such as a home or workplace.**Public:** Applies when computer is connected to public network connection.**Different types of actions available/What action should be taken when a connection matches the specified the conditions****Allow the connection:** This includes connections that are protected with IPsec as well as those are not**Allow the connection if it is secure:** This includes only connections that have been authenticated by using IPsec. Connections will be secured using the settings in IPsec properties and rules in the Connection Security Rule node. Block the connection.

**A) Blocking a port:****Step 1:** Open control panel and go to System Security**Step 2:** Now Select Windows Defender Firewall.

**Step 3:** Now you need to select Advanced setting.



**Step 4:** Now Select Outbound Rules.



**Step 5:** Inside Outbound rules -> Select New Rules -> select Port and then click on next.

New Outbound Rule Wizard

**Rule Type**

Select the type of firewall rule to create.

**Steps:**

- Rule Type
- Protocol and Ports
- Action
- Profile
- Name

What type of rule would you like to create?

☐ **Program**  
Rule that controls connections for a program.

☒ **Port**  
Rule that controls connections for a TCP or UDP port.

☐ **Predefined:**  
@FirewallAPI.dll,-80200  
Rule that controls connections for a Windows experience.

☐ **Custom**  
Custom rule.

< Back   Next >   Cancel

**Step 6:** Select the protocols and enter the port that you want to want to block

New Outbound Rule Wizard

**Protocol and Ports**

Specify the protocols and ports to which this rule applies.

**Steps:**

- Rule Type
- Protocol and Ports
- Action
- Profile
- Name

Does this rule apply to TCP or UDP?

☒ **TCP**

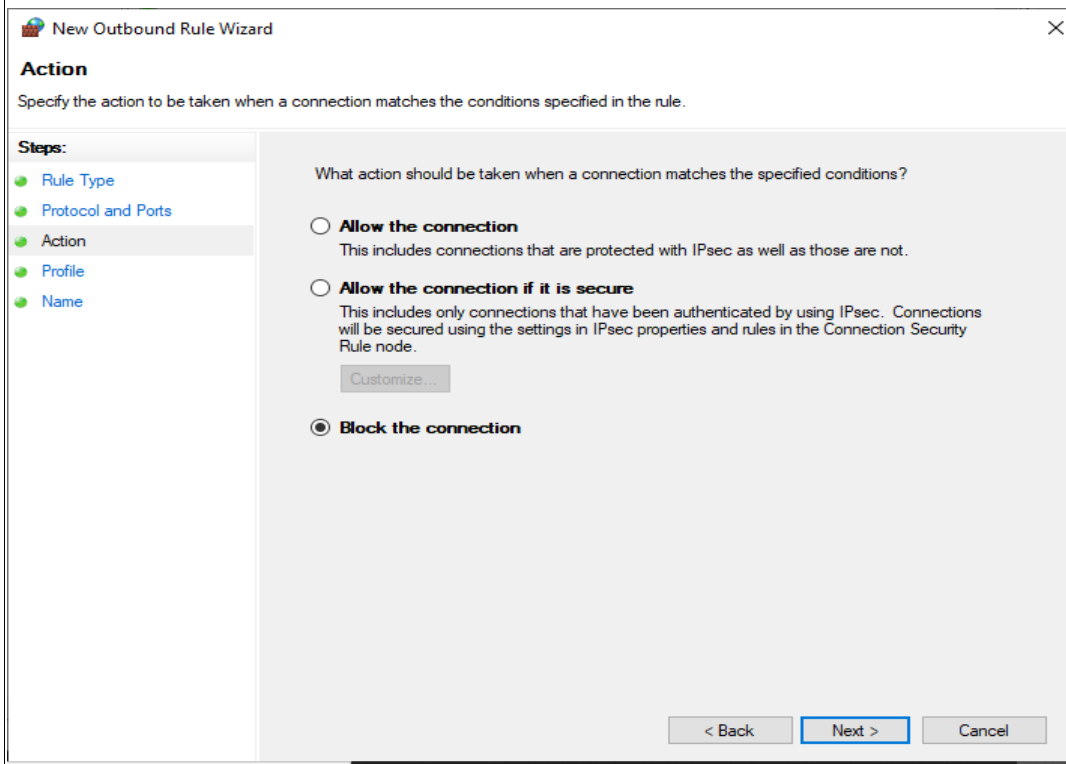
☐ **UDP**

Does this rule apply to all remote ports or specific remote ports?

☐ **All remote ports**

☒ **Specific remote ports:** 80  
Example: 80, 443, 5000-5010

< Back   Next >   Cancel

**Step 7:** Select the action block the connection for blocking a port

The screenshot shows the 'New Outbound Rule Wizard' window at the 'Action' step. The left sidebar lists the steps: Rule Type, Protocol and Ports, Action (selected), Profile, and Name. The main area asks 'What action should be taken when a connection matches the specified conditions?'. There are three radio button options: 'Allow the connection' (unselected), 'Allow the connection if it is secure' (unselected), and 'Block the connection' (selected). Below the first two options is a 'Customize...' button. At the bottom right are '< Back', 'Next >', and 'Cancel' buttons.

**New Outbound Rule Wizard**

**Action**

Specify the action to be taken when a connection matches the conditions specified in the rule.

**Steps:**

- Rule Type
- Protocol and Ports
- Action**
- Profile
- Name

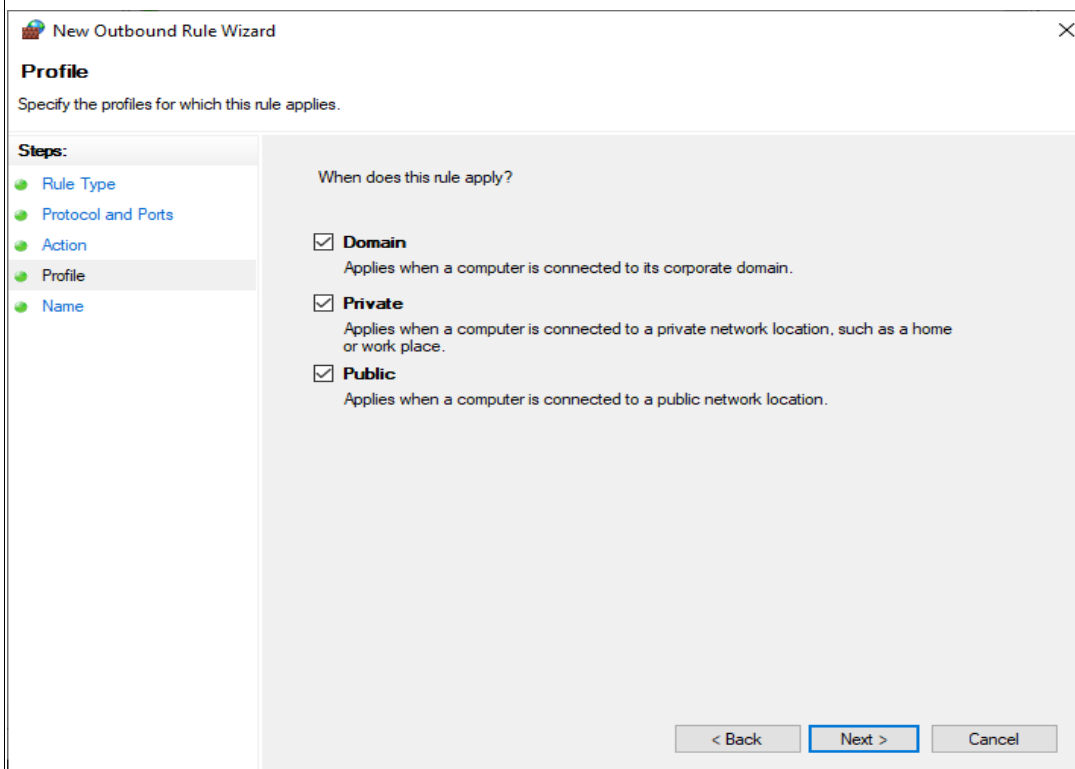
What action should be taken when a connection matches the specified conditions?

☐ **Allow the connection**  
This includes connections that are protected with IPsec as well as those are not.

☐ **Allow the connection if it is secure**  
This includes only connections that have been authenticated by using IPsec. Connections will be secured using the settings in IPsec properties and rules in the Connection Security Rule node.  
[Customize...](#)

☒ **Block the connection**

< Back   Next >   Cancel

**Step 8:** Select the profiles domain private or public.

The screenshot shows the 'New Outbound Rule Wizard' window at the 'Profile' step. The left sidebar lists the steps: Rule Type, Protocol and Ports, Action, Profile (selected), and Name. The main area asks 'When does this rule apply?'. There are three checked checkbox options: 'Domain', 'Private', and 'Public'. Each option has a description of when the rule applies. At the bottom right are '< Back', 'Next >', and 'Cancel' buttons.

**New Outbound Rule Wizard**

**Profile**

Specify the profiles for which this rule applies.

**Steps:**

- Rule Type
- Protocol and Ports
- Action
- Profile**
- Name

When does this rule apply?

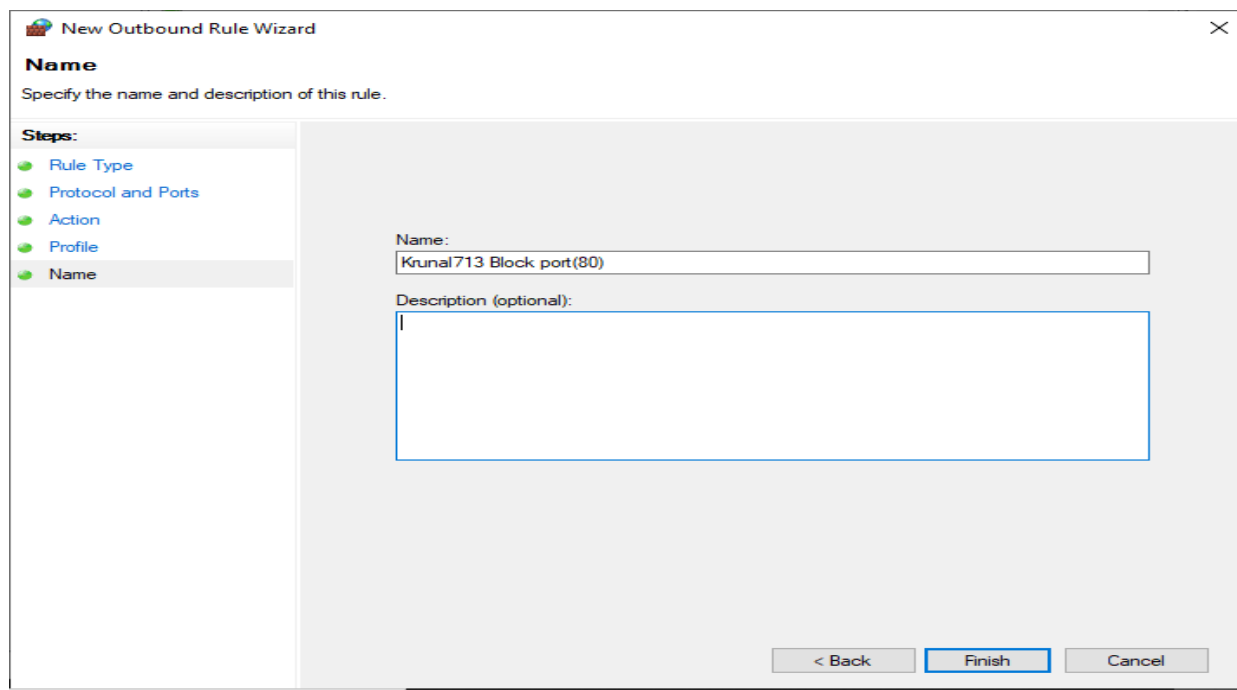
☒ **Domain**  
Applies when a computer is connected to its corporate domain.

☒ **Private**  
Applies when a computer is connected to a private network location, such as a home or work place.

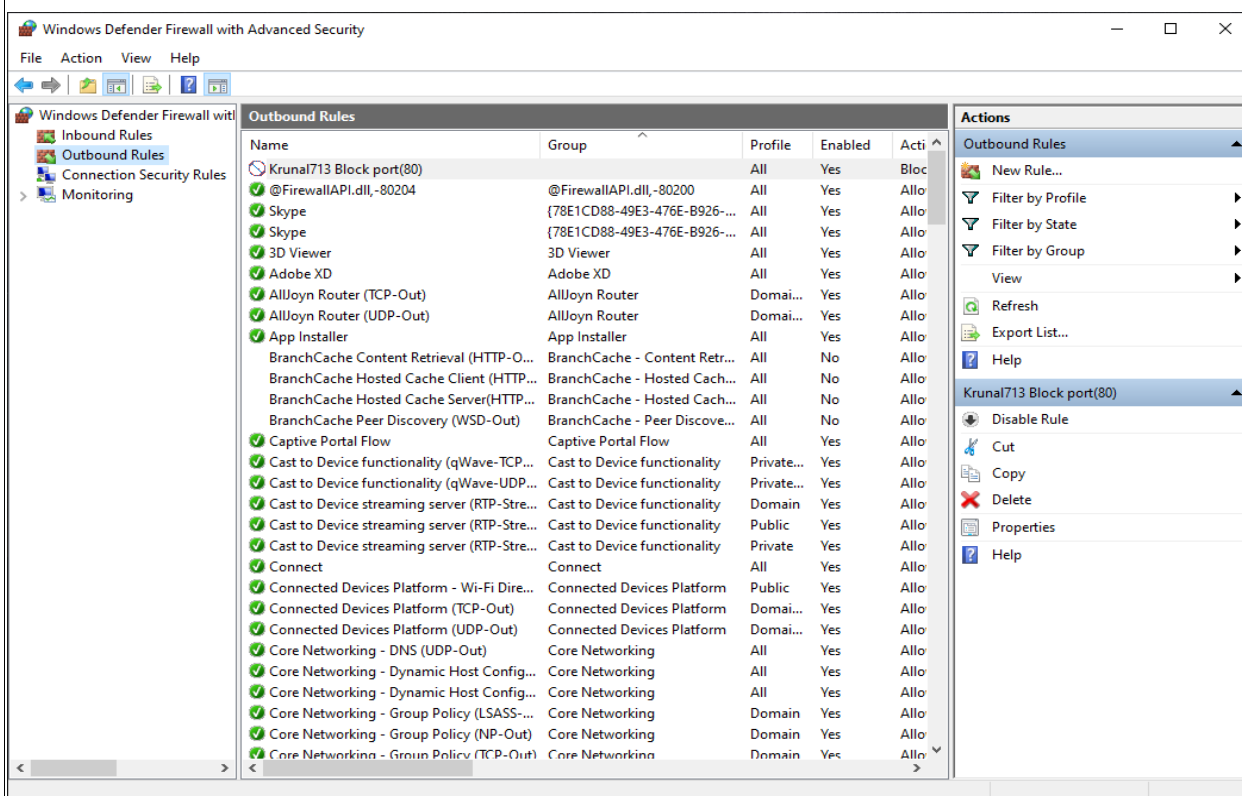
☒ **Public**  
Applies when a computer is connected to a public network location.

< Back   Next >   Cancel

**Step 9:** Give a name to our new set rule and click on finish.



**Output:**



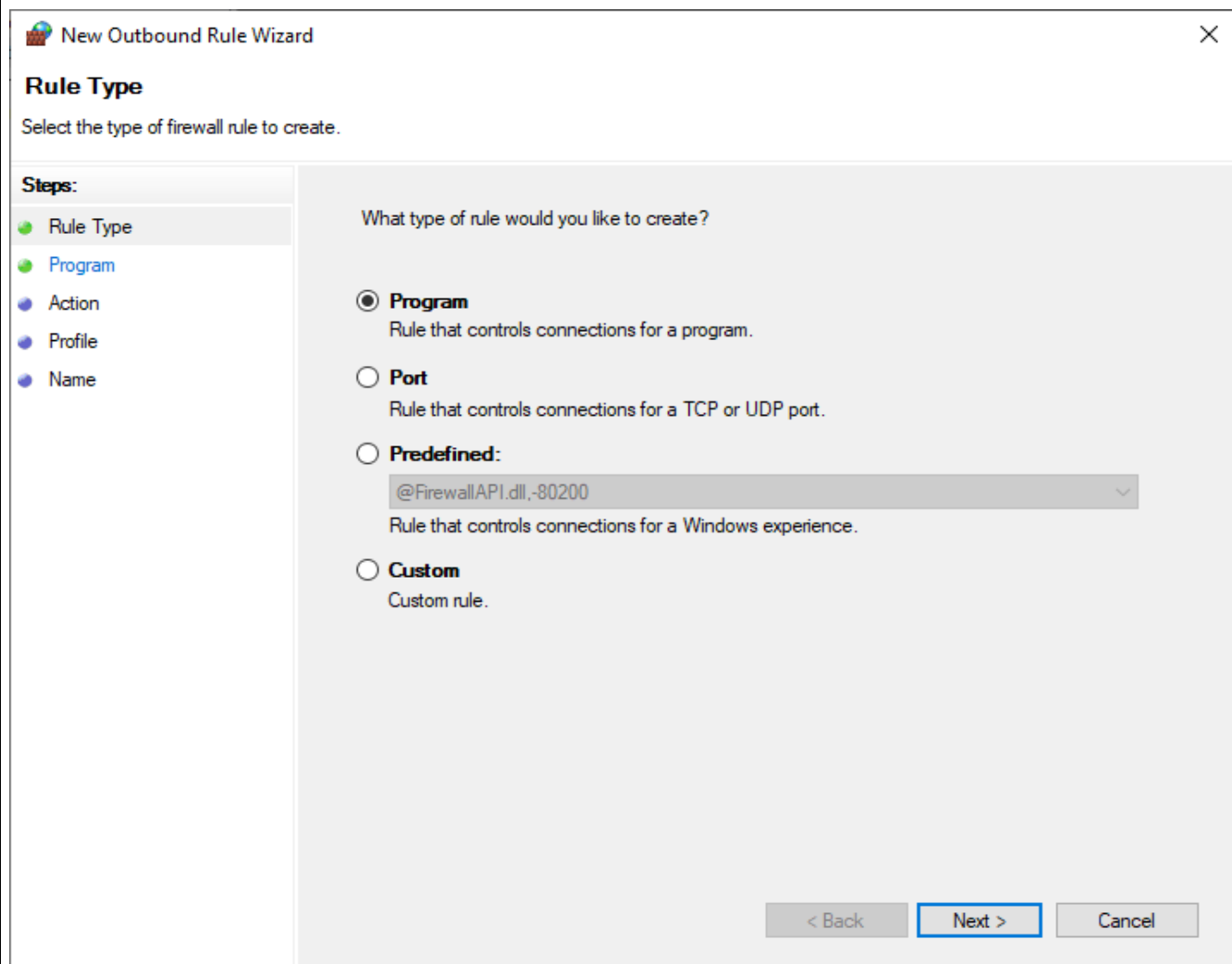


**B) Blocking a program:**

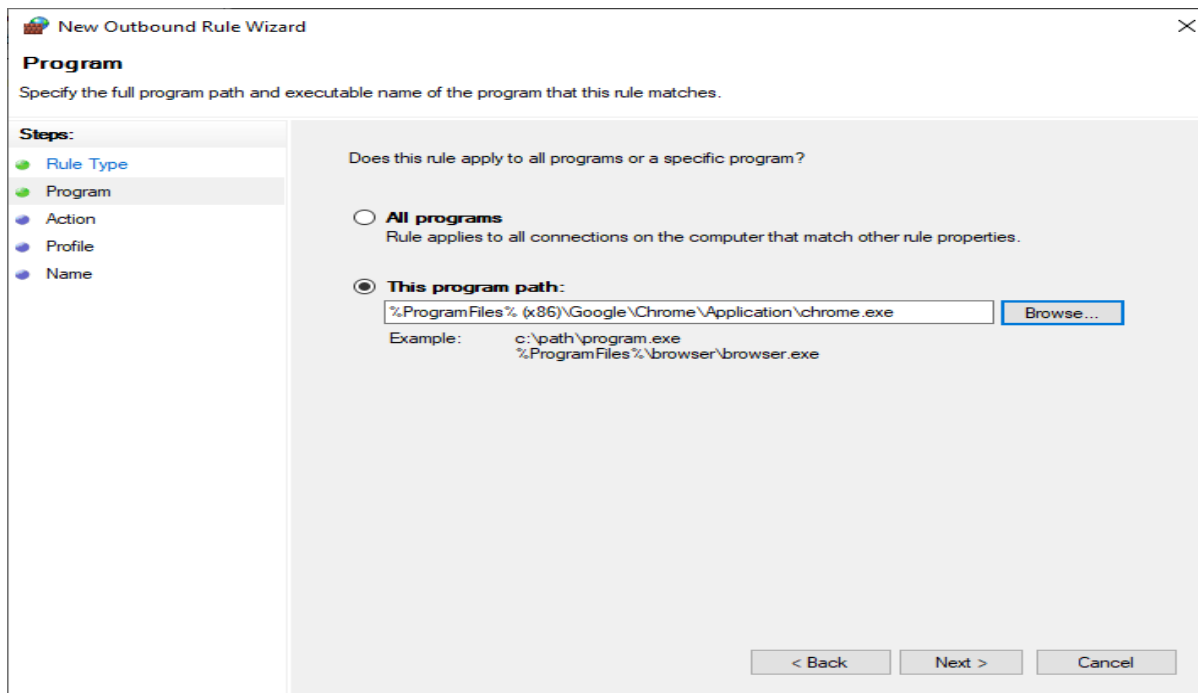
**Rule that controls the connection of a program**

**Step1: Repeat PartA Step1 to Step4.**

**Step 5:** Inside Outbound rules -> Select New Rules -> select a program and then click on next.

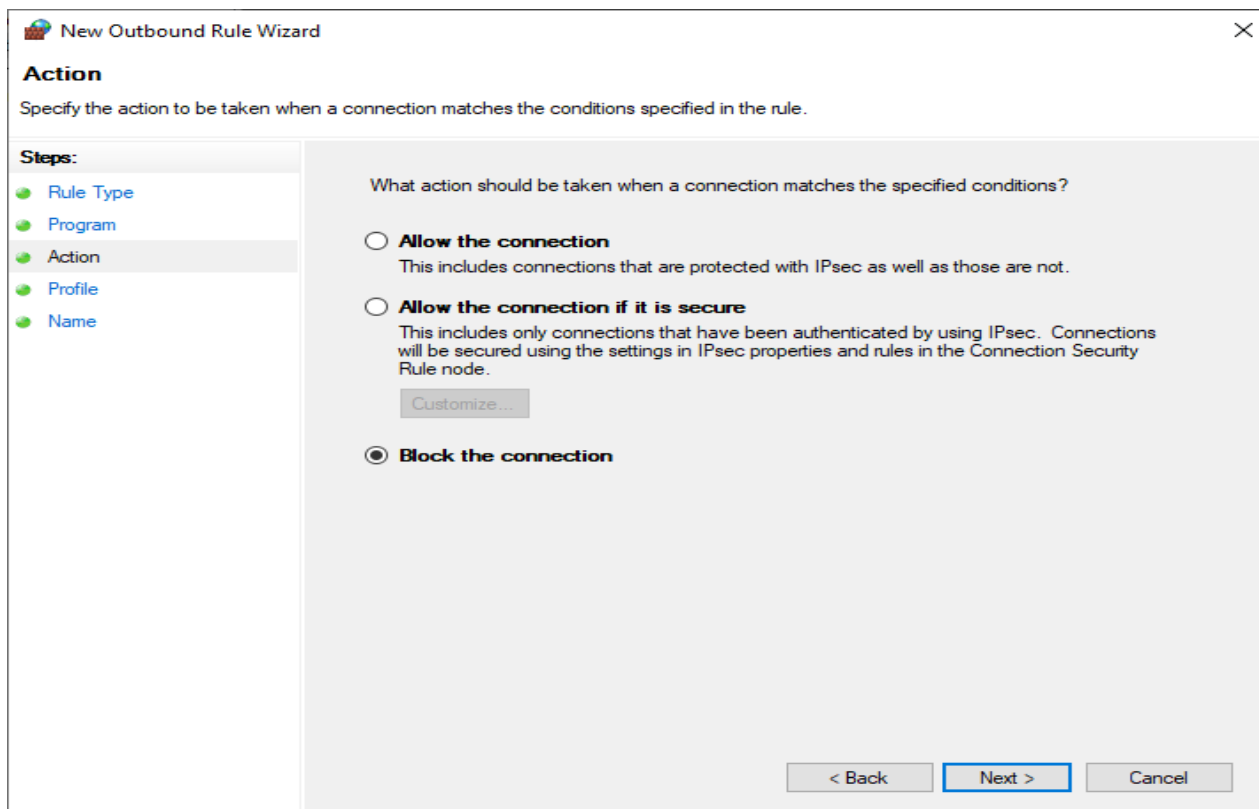


**Step 6:** Choose the path of the program from the directory.



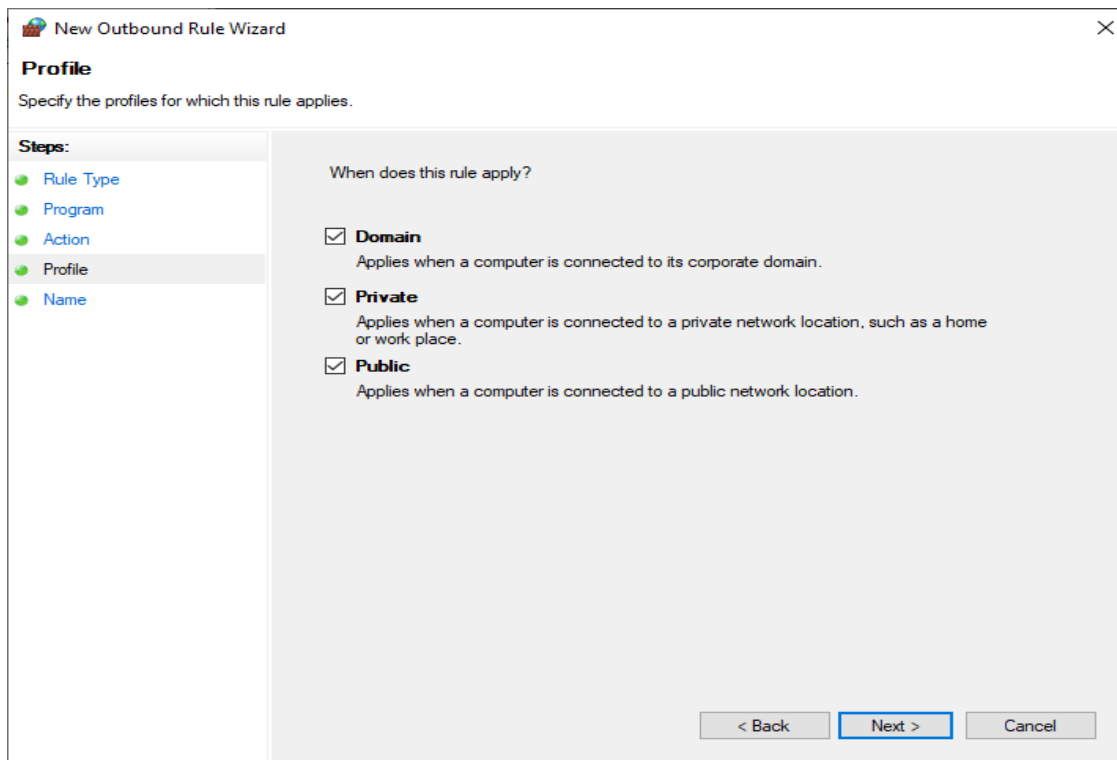
The screenshot shows the 'New Outbound Rule Wizard' window at the 'Program' step. The title bar reads 'New Outbound Rule Wizard'. The main heading is 'Program'. Below it, the instruction says: 'Specify the full program path and executable name of the program that this rule matches.' On the left, a 'Steps:' pane lists 'Rule Type', 'Program', 'Action', 'Profile', and 'Name', with 'Program' selected. The main area asks: 'Does this rule apply to all programs or a specific program?'. There are two radio button options: 'All programs' (unselected) and 'This program path:' (selected). The 'This program path:' option has a text box containing '%ProgramFiles%\Google\Chrome\Application\chrome.exe' and a 'Browse...' button. Below the text box, it says 'Example: c:\path\program.exe' and '%ProgramFiles%\browser\browser.exe'. At the bottom, there are three buttons: '< Back', 'Next >', and 'Cancel'.

**Step 7:** Click on Block the connection.



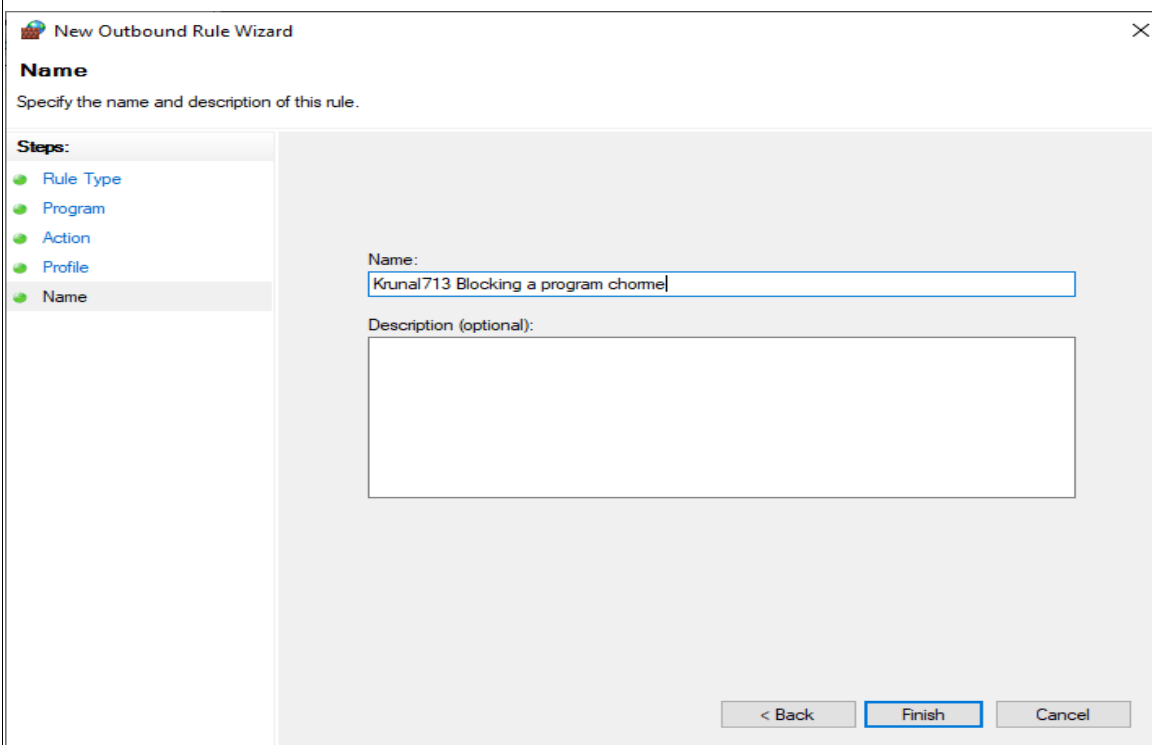
The screenshot shows the 'New Outbound Rule Wizard' window at the 'Action' step. The title bar reads 'New Outbound Rule Wizard'. The main heading is 'Action'. Below it, the instruction says: 'Specify the action to be taken when a connection matches the conditions specified in the rule.' On the left, a 'Steps:' pane lists 'Rule Type', 'Program', 'Action', 'Profile', and 'Name', with 'Action' selected. The main area asks: 'What action should be taken when a connection matches the specified conditions?'. There are three radio button options: 'Allow the connection' (unselected), 'Allow the connection if it is secure' (unselected), and 'Block the connection' (selected). The 'Allow the connection' option has a description: 'This includes connections that are protected with IPsec as well as those are not.' The 'Allow the connection if it is secure' option has a description: 'This includes only connections that have been authenticated by using IPsec. Connections will be secured using the settings in IPsec properties and rules in the Connection Security Rule node.' Below the 'Allow the connection if it is secure' option is a 'Customize...' button. At the bottom, there are three buttons: '< Back', 'Next >', and 'Cancel'.

**Step 8:** Select the profiles domain private or public.

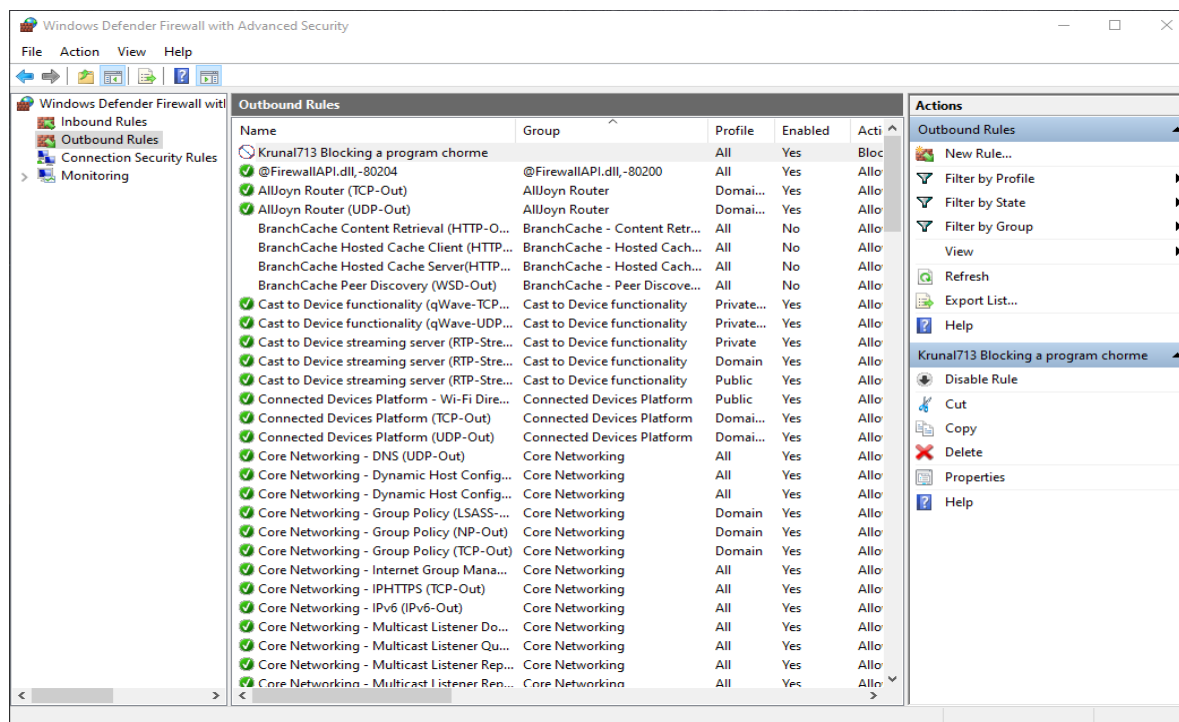
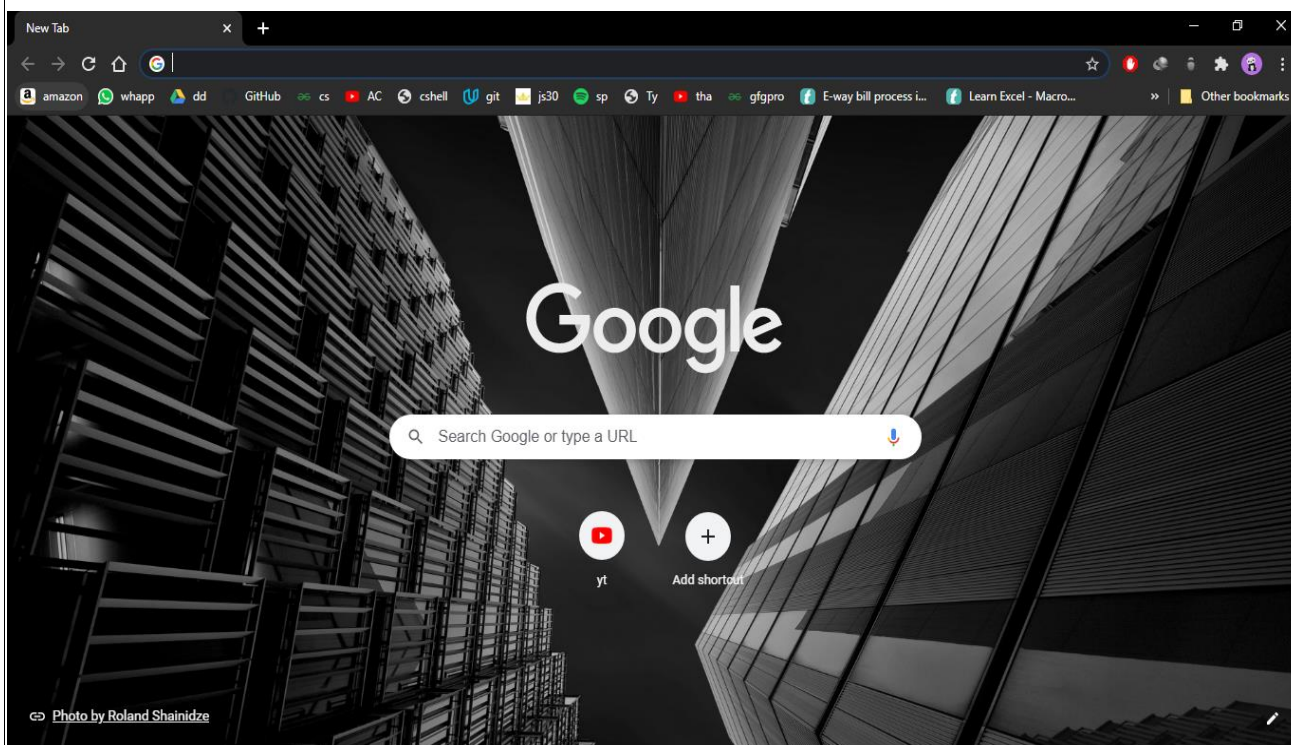


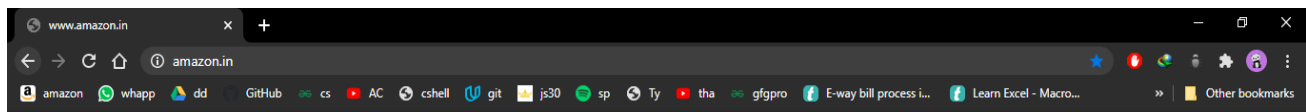
The screenshot shows the 'New Outbound Rule Wizard' window at the 'Profile' step. The title bar reads 'New Outbound Rule Wizard'. The main heading is 'Profile', with the instruction 'Specify the profiles for which this rule applies.' Below this is a 'Steps:' list on the left containing 'Rule Type', 'Program', 'Action', 'Profile' (highlighted), and 'Name'. The main area is titled 'When does this rule apply?' and contains three checked options: 'Domain' (Applies when a computer is connected to its corporate domain.), 'Private' (Applies when a computer is connected to a private network location, such as a home or work place.), and 'Public' (Applies when a computer is connected to a public network location.). At the bottom right are buttons for '< Back', 'Next >', and 'Cancel'.

**Step 9:** Give a name to your new set rule and click on finish.



The screenshot shows the 'New Outbound Rule Wizard' window at the 'Name' step. The title bar reads 'New Outbound Rule Wizard'. The main heading is 'Name', with the instruction 'Specify the name and description of this rule.' Below this is a 'Steps:' list on the left containing 'Rule Type', 'Program', 'Action', 'Profile', and 'Name' (highlighted). The main area contains a 'Name:' label followed by a text box containing 'Krunal713 Blocking a program chrome'. Below this is a 'Description (optional):' label followed by a large empty text box. At the bottom right are buttons for '< Back', 'Finish', and 'Cancel'.

**Output:****Before Applying the Rule:**

**After Applying the Rule:****Your Internet access is blocked**

Firewall or antivirus software may have blocked the connection.

Try:

- Checking the connection
- [Checking firewall and antivirus configurations](#)
- [Running Windows Network Diagnostics](#)

ERR\_NETWORK\_ACCESS\_DENIED

Details

**C) Blocking a website:****Step1: Repeat PartA Step1 to Step4.****Step 5:** Inside Outbound rules -> Select New Rules -> select custom and then click on next.

**New Outbound Rule Wizard**

**Rule Type**

Select the type of firewall rule to create.

**Steps:**

- Rule Type
- Program
- Protocol and Ports
- Scope
- Action
- Profile
- Name

What type of rule would you like to create?

☐ **Program**  
Rule that controls connections for a program.

☐ **Port**  
Rule that controls connections for a TCP or UDP port.

☐ **Predefined:**  
@FirewallAPI.dll,-80200  
Rule that controls connections for a Windows experience.

☒ **Custom**  
Custom rule.

< Back   **Next >**   Cancel

**Step 6:** When you click next you would see a window where you will see “Steps:” on left hand side of the screen. From that select “Scope”.

New Outbound Rule Wizard

**Scope**

Specify the local and remote IP addresses to which this rule applies.

**Steps:**

- Rule Type
- Program
- Protocol and Ports
- Scope**
- Action
- Profile
- Name

**Which local IP addresses does this rule apply to?**

☒ Any IP address

☐ These IP addresses:

Customize the interface types to which this rule applies:

**Which remote IP addresses does this rule apply to?**

☒ Any IP address

☐ These IP addresses:

< Back Next > Cancel

**Step 7:** In scope click on These IP addresses in remote IP

New Outbound Rule Wizard

**Scope**

Specify the local and remote IP addresses to which this rule applies.

**Steps:**

- Rule Type
- Program
- Protocol and Ports
- Scope**
- Action
- Profile
- Name

**Which local IP addresses does this rule apply to?**

☒ Any IP address

☐ These IP addresses:

Customize the interface types to which this rule applies:

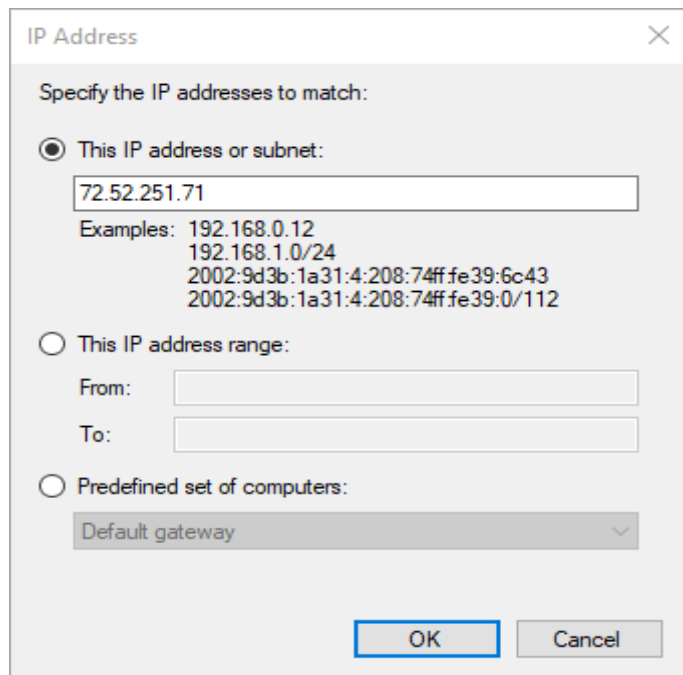
**Which remote IP addresses does this rule apply to?**

☐ Any IP address

☒ These IP addresses:

< Back Next > Cancel

**Step 8:** Click on add and Add the IP address of the website that you want to block and click ok.



IP Address

Specify the IP addresses to match:

☒ This IP address or subnet:

72.52.251.71

Examples: 192.168.0.12  
192.168.1.0/24  
2002:9d3b:1a31:4:208:74ff:fe39:6c43  
2002:9d3b:1a31:4:208:74ff:fe39:0/112

☐ This IP address range:

From:

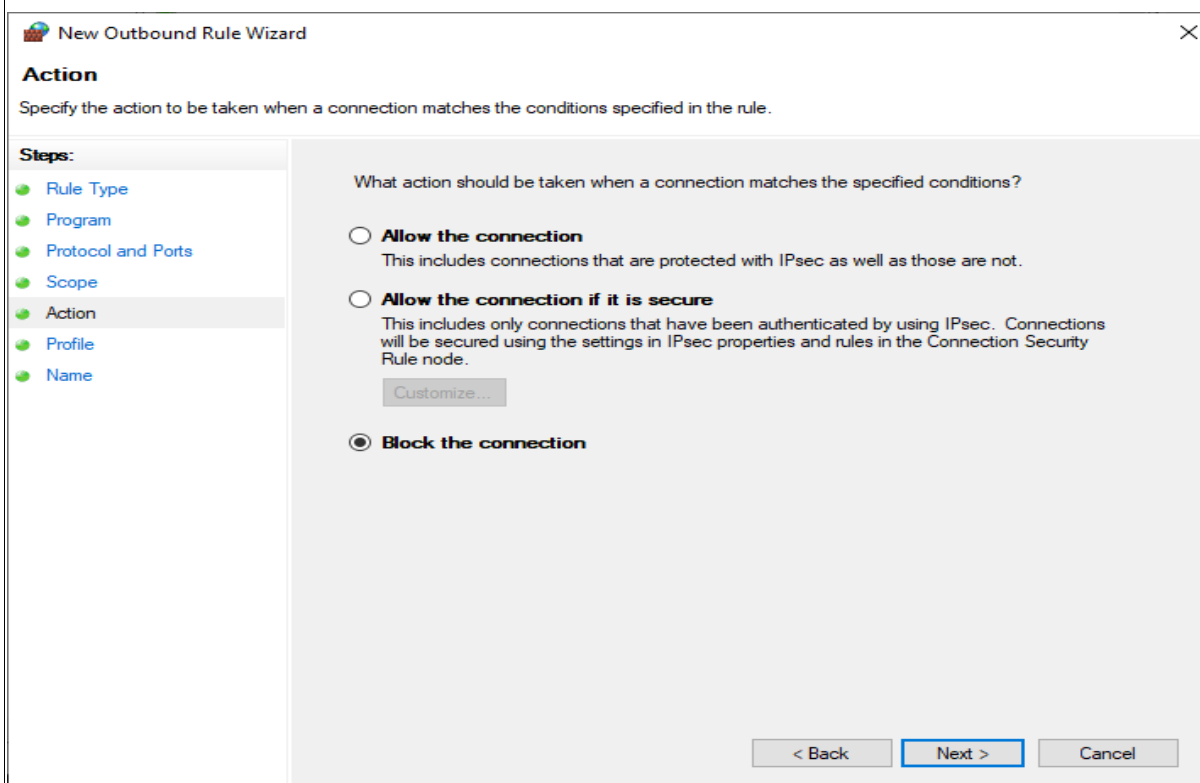
To:

☐ Predefined set of computers:

Default gateway

OK Cancel

**Step 9:** Click on Block the connection in action



New Outbound Rule Wizard

**Action**

Specify the action to be taken when a connection matches the conditions specified in the rule.

**Steps:**

- Rule Type
- Program
- Protocol and Ports
- Scope
- Action**
- Profile
- Name

What action should be taken when a connection matches the specified conditions?

☐ **Allow the connection**  
This includes connections that are protected with IPsec as well as those are not.

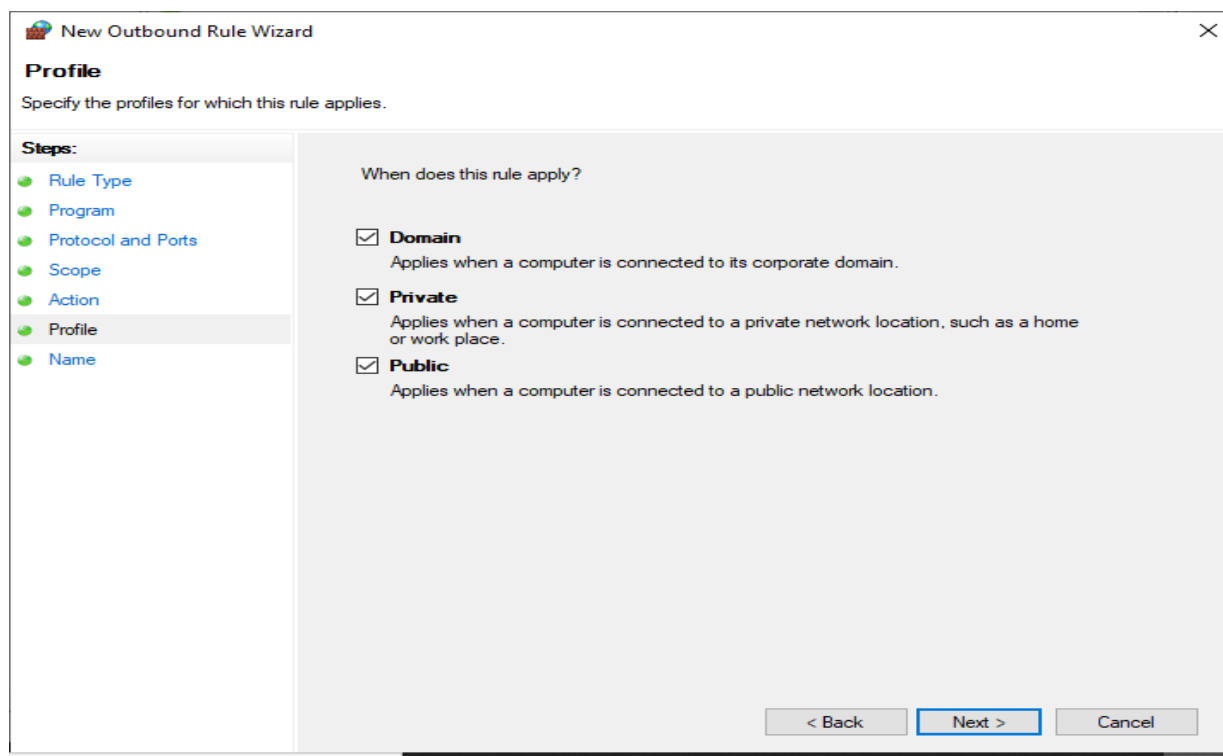
☐ **Allow the connection if it is secure**  
This includes only connections that have been authenticated by using IPsec. Connections will be secured using the settings in IPsec properties and rules in the Connection Security Rule node.

☒ **Block the connection**

< Back Next > Cancel

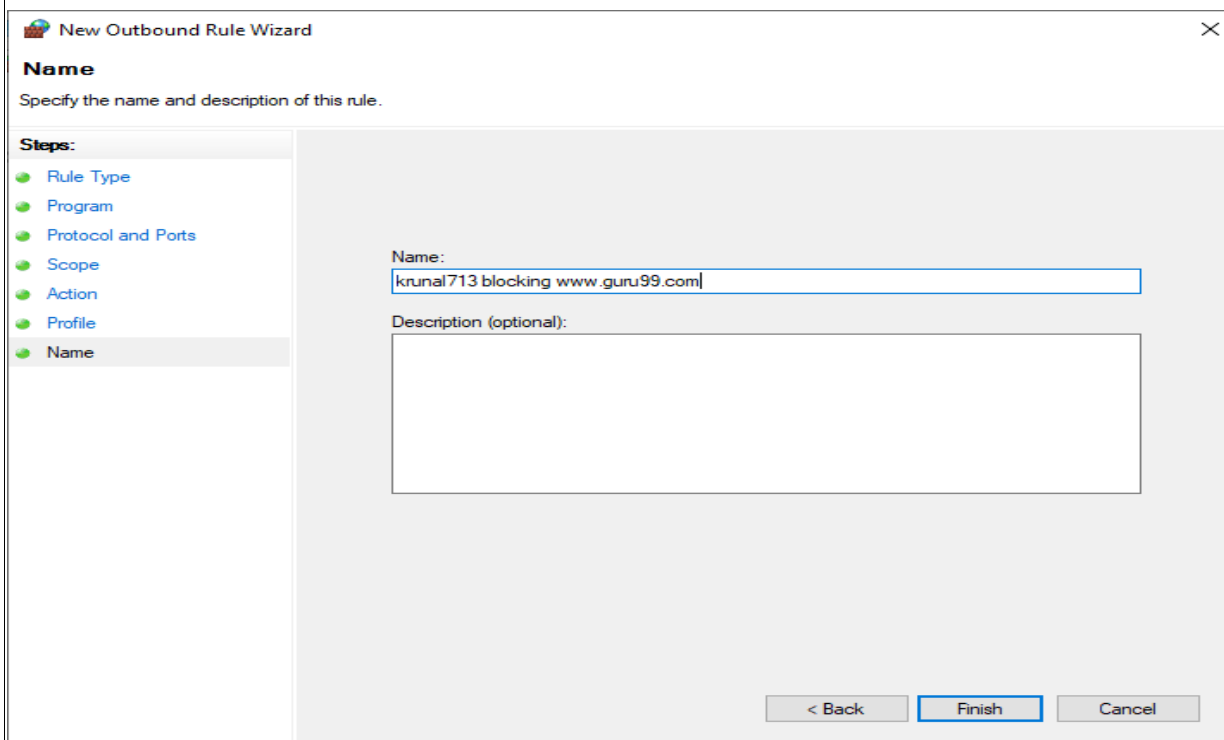


**Step 10:** Select the profiles domain private or public.

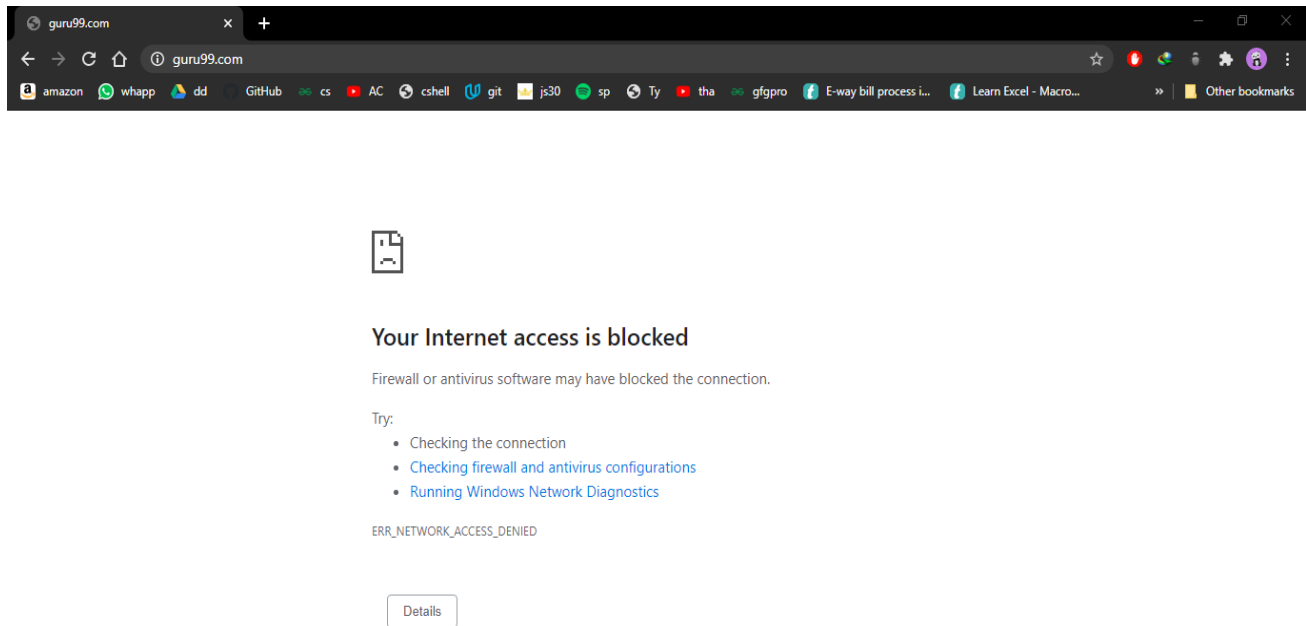


The screenshot shows the 'New Outbound Rule Wizard' window at the 'Profile' step. The title bar reads 'New Outbound Rule Wizard'. The main heading is 'Profile' with the instruction 'Specify the profiles for which this rule applies.' On the left, a 'Steps:' list includes 'Rule Type', 'Program', 'Protocol and Ports', 'Scope', 'Action', 'Profile' (highlighted), and 'Name'. The main area is titled 'When does this rule apply?' and contains three checked options: 'Domain' (Applies when a computer is connected to its corporate domain.), 'Private' (Applies when a computer is connected to a private network location, such as a home or work place.), and 'Public' (Applies when a computer is connected to a public network location.). At the bottom right are buttons for '< Back', 'Next >', and 'Cancel'.

**Step 11:** Give a name to your new set rule and click on finish.



The screenshot shows the 'New Outbound Rule Wizard' window at the 'Name' step. The title bar reads 'New Outbound Rule Wizard'. The main heading is 'Name' with the instruction 'Specify the name and description of this rule.' On the left, the 'Steps:' list includes 'Rule Type', 'Program', 'Protocol and Ports', 'Scope', 'Action', 'Profile', and 'Name' (highlighted). The main area has a 'Name:' label followed by a text box containing 'krunal713 blocking www.guru99.com'. Below it is a 'Description (optional):' label followed by a larger empty text box. At the bottom right are buttons for '< Back', 'Finish', and 'Cancel'.

**Output:**

```
Command Prompt

C:\Users\BlackBot>ping www.guru99.com

Pinging guru99.com [72.52.251.71] with 32 bytes of data:
General failure.
General failure.
General failure.
General failure.

Ping statistics for 72.52.251.71:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\Users\BlackBot>
```