Niranjana S
312217205059

AES ALGORITHM:

```
import java.util.*;


class aes {
	private static class AES
	{
		String[][] sbox={
	{"63","7C","77","7B","F2","6B","6F","C5","30","01","67","2B","FE","D7","AB","76"},
	{"CA","82","C9","7D","FA","59","47","F0","AD","D4","A2","AF","9C","A4","72","C0"},
	{"B7","FD","93","26","36","3F","F7","CC","34","A5","E5","F1","71","D8","31","15"},
	{"04","C7","23","C3","18","96","05","9A","07","12","80","E2","EB","27","B2","75"},
	{"09","83","2C","1A","1B","6E","5A","A0","52","3B","D6","B3","29","E3","2F","84"},
	{"53","D1","00","ED","20","FC","B1","5B","6A","CB","BE","39","4A","4C","58","CF"},
	{"D0","EF","AA","FB","43","4D","33","85","45","F9","02","7F","50","3C","9F","A8"},
	{"51","A3","40","8F","92","9D","38","F5","BC","B6","DA","21","10","FF","F3","D2"},
	{"CD","0C","13","EC","5F","97","44","17","C4","A7","7E","3D","64","5D","19","73"},
	{"60","81","4F","DC","22","2A","90","88","46","EE","B8","14","DE","5E","0B","DB"},
	{"E0","32","3A","0A","49","06","24","5C","C2","D3","AC","62","91","95","E4","79"},
	{"E7","C8","37","6D","8D","D5","4E","A9","6C","56","F4","EA","65","7A","AE","08"},
	{"BA","78","25","2E","1C","A6","B4","C6","E8","DD","74","1F","4B","BD","8B","8A"},
```

```
{"70","3E","B5","66","48","03","F6","0E","61","35","57","B9","86","C1","1D","9E"},

{"E1","F8","98","11","69","D9","8E","94","9B","1E","87","E9","CE","55","28","DF"},

{"8C","A1","89","0D","BF","E6","42","68","41","99","2D","0F","B0","54","BB","16"}
};
String[][] lbox={
                    {"
","00","19","01","32","02","1A","C6","4B","C7","1B","68","33","EE","DF","03"},

{"64","04","E0","0E","34","8D","81","EF","4C","71","08","C8","F8","69","1C","C1"},

{"7D","C2","1D","B5","F9","B9","27","6A","4D","E4","A6","72","9A","C9","09","78"},

{"65","2F","8A","05","21","0F","E1","24","12","F0","82","45","35","93","DA","8E"},

{"96","8F","DB","BD","36","D0","CE","94","13","5C","D2","F1","40","46","83","38"},

{"66","DD","FD","30","BF","06","8B","62","B3","25","E2","98","22","88","91","10"},

{"7E","6E","48","C3","A3","B6","1E","42","3A","6B","28","54","FA","85","3D","BA"},

{"2B","79","0A","15","9B","9F","5E","CA","4E","D4","AC","E5","F3","73","A7","57"},

{"AF","58","A8","50","F4","EA","D6","74","4F","AE","E9","D5","E7","E6","AD","E8"},

{"2C","D7","75","7A","EB","16","0B","F5","59","CB","5F","B0","9C","A9","51","A0"},

{"7F","0C","F6","6F","17","C4","49","EC","D8","43","1F","2D","A4","76","7B","B7"},

{"CC","BB","3E","5A","FB","60","B1","86","3B","52","A1","6C","AA","55","29","9D"},

{"97","B2","87","90","61","BE","DC","FC","BC","95","CF","CD","37","3F","5B","D1"},
```

{"53","39","84","3C","41","A2","6D","47","14","2A","9E","5D","56","F2","D3","AB"},

{"44","11","92","D9","23","20","2E","89","B4","7C","B8","26","77","99","E3","A5"},

{"67","4A","ED","DE","C5","31","FE","18","0D","63","8C","80","C0","F7","70","07"}
};

String[][] ebox={

{"01","03","05","0F","11","33","55","FF","1A","2E","72","96","A1","F8","13","35"},

{"5F","E1","38","48","D8","73","95","A4","F7","02","06","0A","1E","22","66","AA"},

{"E5","34","5C","E4","37","59","EB","26","6A","BE","D9","70","90","AB","E6","31"},

{"53","F5","04","0C","14","3C","44","CC","4F","D1","68","B8","D3","6E","B2","CD"},

{"4C","D4","67","A9","E0","3B","4D","D7","62","A6","F1","08","18","28","78","88"},

{"83","9E","B9","D0","6B","BD","DC","7F","81","98","B3","CE","49","DB","76","9A"},

{"B5","C4","57","F9","10","30","50","F0","0B","1D","27","69","BB","D6","61","A3"},

{"FE","19","2B","7D","87","92","AD","EC","2F","71","93","AE","E9","20","60","A0"},

{"FB","16","3A","4E","D2","6D","B7","C2","5D","E7","32","56","FA","15","3F","41"},

{"C3","5E","E2","3D","47","C9","40","C0","5B","ED","2C","74","9C","BF","DA","75"},

{"9F","BA","D5","64","AC","EF","2A","7E","82","9D","BC","DF","7A","8E","89","80"},

{"9B","B6","C1","58","E8","23","65","AF","EA","25","6F","B1","C8","43","C5","54"},

{"FC","1F","21","63","A5","F4","07","09","1B","2D","77","99","B0","CB","46","CA"},

```
{"45","CF","4A","DE","79","8B","86","91","A8","E3","3E","42","C6","51","F3","0E"},

{"12","36","5A","EE","29","7B","8D","8C","8F","8A","85","94","A7","F2","0D","17"},

{"39","4B","DD","7C","84","97","A2","FD","1C","24","6C","B4","C7","52","F6","01"}
};

String[][]
invsbox={{"52","09","6A","D5","30","36","A5","38","BF","40","A3","9E","81","F3","D7","FB"
},

{"7C","E3","39","82","9B","2F","FF","87","34","8E","43","44","C4","DE","E9","CB"},

{"54","7B","94","32","A6","C2","23","3D","EE","4C","95","0B","42","FA","C3","4E"},

{"08","2E","A1","66","28","D9","24","B2","76","5B","A2","49","6D","8B","D1","25"},

{"72","F8","F6","64","86","68","98","16","D4","A4","5C","CC","5D","65","B6","92"},

{"6C","70","48","50","FD","ED","B9","DA","5E","15","46","57","A7","8D","9D","84"},

{"90","D8","AB","00","8C","BC","D3","0A","F7","E4","58","05","B8","B3","45","06"},

{"D0","2C","1E","8F","CA","3F","0F","02","C1","AF","BD","03","01","13","8A","6B"},

{"3A","91","11","41","4F","67","DC","EA","97","F2","CF","CE","F0","B4","E6","73"},

{"96","AC","74","22","E7","AD","35","85","E2","F9","37","E8","1C","75","DF","6E"},

{"47","F1","1A","71","1D","29","C5","89","6F","B7","62","0E","AA","18","BE","1B"},

{"FC","56","3E","4B","C6","D2","79","20","9A","DB","C0","FE","78","CD","5A","F4"},

{"1F","DD","A8","33","88","07","C7","31","B1","12","10","59","27","80","EC","5F"},
```

```
{"60","51","7F","A9","19","B5","4A","0D","2D","E5","7A","9F","93","C9","9C","EF"},

{"A0","E0","3B","4D","AE","2A","F5","B0","C8","EB","BB","3C","83","53","99","61"},

{"17","2B","04","7E","BA","77","D6","26","E1","69","14","63","55","21","0C","7D"},
};
String[] rcon={"01","02","04","08","10","20","40","80","1B","36"};

String hextoBin(String input)
        {
                int n = input.length() * 4;
                input = Long.toBinaryString(
                        Long.parseUnsignedLong(input, 16));
                while (input.length() < n)
                        input = "0" + input;
                return input;
        }

        // binary to hexadecimal conversion
        String binToHex(String input)
        {
                int n = (int)input.length() / 4;
                input = Long.toHexString(
                        Long.parseUnsignedLong(input, 2));
                while (input.length() < n)
                        input = "0" + input;
                return input;
        }
        String[] leftCircularShift(String[] input, int numBits)
```

```
{
        //int n = input.length() * 4;
        String perm[] = new String[5];
        if(numBits==1)
        {
        for (int i = 0; i < 3; i++)
                perm[i] = input[i+1];
        perm[3] = input[0];
        }
        else if(numBits==2)
        {
                for (int i = 0; i < 2; i++)
                        perm[i] = input[i+2];
        perm[3] = input[1];
        perm[2] =input[0];
        }
        else if(numBits==3)
        {
                perm[0]=input[3];
                perm[1]=input[0];
                perm[2]=input[1];
                perm[3]=input[2];
        }
        else
                return input;
        return perm;
}
String[] invleftCircularShift(String[] input, int numBits)
{
```

```
        //int n = input.length() * 4;
        String perm[] = new String[5];
        if(numBits==1)
        {
        for (int i = 0; i < 3; i++)
                perm[i+1] = input[i];
        perm[0] = input[3];
        }
        else if(numBits==2)
        {
                for (int i = 0; i < 2; i++)
                        perm[i+2] = input[i];
        perm[1] = input[3];
        perm[0] =input[2];
        }
        else if(numBits==3)
        {
                perm[0]=input[3];
                perm[1]=input[0];
                perm[2]=input[1];
                perm[3]=input[2];
        }
        else
                return input;
        return perm;
}

String[][] mix(String[][] pt)
{
```

```
int i,j,k;
String[][] mixcol=new String[4][4];
String[][] key={{"02","03","01","01"},
                {"01","02","03","01"},
                {"01","01","02","03"},
                {"03","01","01","02"}};
//String temp="";
int value=0;
for (i=0;i<4;i++)
{
    for (j=0;j<4;j++)
    {
        mixcol[i][j]="00";
        for (k=0;k<4;k++)
        {
            value=Integer.parseInt(pt[k][j],16);
            if(key[i][k].equals("03"))
            {
                if(value>=128)
                {
                    int
temp=((2*value)^Integer.parseInt(pt[k][j],16))^283;

    mixcol[i][j]=Integer.toHexString(Integer.parseInt(mixcol[i][j],16)^temp);
                }
                else
                {
                    int
find=(2*value)^Integer.parseInt(pt[k][j],16);
```

```
mixcol[i][j]=Integer.toHexString(Integer.parseInt(mixcol[i][j],16)^find);
                                        }
                                }
                                else if(key[i][k].equals("02"))
                                {
                                        if(value>=128)
                                        {
                                                int fin=((2*value)^283);

mixcol[i][j]=Integer.toHexString(Integer.parseInt(mixcol[i][j],16)^fin);
                                        }
                                        else
                                        {

mixcol[i][j]=Integer.toHexString(Integer.parseInt(mixcol[i][j],16)^(2*value));
                                        }
                                }
                                else
                                {
                                        int tem=Integer.parseInt(key[i][k],16);

mixcol[i][j]=Integer.toHexString(Integer.parseInt(mixcol[i][j],16)^(tem*value));
                                }
                        }


                }
        }
```

```
		return mixcol;
	}
	String[][] invmix(String[][] pt)
	{
		int i,j,k;
		String[][] mixcol=new String[4][4];
		String[][] key={{"14","11","13","09"},
						{"09","14","11","13"},
						{"13","09","14","11"},
						{"11","13","09","14"}};
		//String temp="";
		int value=0;
		for (i=0;i<4;i++)
		{
			for (j=0;j<4;j++)
			{
				mixcol[i][j]="00";
				for (k=0;k<4;k++)
				{

	//mixcol[i][j]=Integer.toHexString((Integer.parseInt(key[i][k])^(Integer.parseInt(pt[k][
j],16)))^(Integer.parseInt(mixcol[i][j],16)));
					value=Integer.parseInt(pt[k][j],16);
					if(key[i][k].equals("09"))
					{
						//if(value>=128)
						//{
							int
temp=((((2*value)*2)*2)^value);
```

```
        mixcol[i][j]=Integer.toHexString(Integer.parseInt(mixcol[i][j],16)^temp);
                                                        //}
                                                        //else
                                                        //{
                                                        //      int
find=(2*value)^Integer.parseInt(pt[k][j],16);
                                                        //
        mixcol[i][j]=Integer.toHexString(Integer.parseInt(mixcol[i][j],16)^find);
                                                        //}
                                                }
                                                else if(key[i][k].equals("11"))
                                                {
                                                        int
temp=(((((2*value)*2)^value)*2)^value);

        mixcol[i][j]=Integer.toHexString(Integer.parseInt(mixcol[i][j],16)^temp);
                                                }
                                                else if(key[i][k].equals("13"))
                                                {
                                                        int
temp=(((((2*value)^value)*2)*2)^value);

        mixcol[i][j]=Integer.toHexString(Integer.parseInt(mixcol[i][j],16)^temp);
                                                }
                                                else
                                                {
                                                        int
temp=(((((2*value)^value)*2)^value)*2);

        mixcol[i][j]=Integer.toHexString(Integer.parseInt(mixcol[i][j],16)^temp);
                                                }
```

```
                        }


                }
        }
        return mixcol;
}
String xor(String a, String b)
{


        int n=a.length();
        int i;
        String output="";
        for(i=0;i<8;i++)
        {
                if(a.charAt(i)==b.charAt(i))
                        output+="0";
                else
                        output+="1";
        }
        return output;
}
String biadd(String a,String b)
{
        int b1=Integer.parseInt(a,2);
        int b2=Integer.parseInt(b,2);
        int sum=b1+b2;
        return Integer.toBinaryString(sum);
}
String binadd(String a,String b)
```

```
{
        int b1=Integer.parseInt(a,16);
        int b2=Integer.parseInt(b,16);
        int mul=b1*b2;
        return Integer.toHexString(mul);
}
String permutation(String sequence)
{
        String output = "";
        //input = hextoBin(input);
        int flag=0;
        String g="",f="";
        char a=sequence.charAt(0);
        char b=sequence.charAt(1);

        if(Character.compare(a,'A')==0 || Character.compare(a,'a')==0)
                g="10";
        else if(Character.compare(a,'B')==0 || Character.compare(a,'b')==0)
                g="11";
        else if(Character.compare(a,'C')==0 || Character.compare(a,'c')==0)
                g="12";
        else if(Character.compare(a,'D')==0 || Character.compare(a,'d')==0)
                g="13";
        else if(Character.compare(a,'E')==0 || Character.compare(a,'e')==0)
                g="14";
        else if(Character.compare(a,'F')==0 || Character.compare(a,'f')==0)
                g="15";
        else
                g+=a;
```

```
        if(Character.compare(b,'A')==0 || Character.compare(b,'a')==0)
                f="10";
        else if(Character.compare(b,'B')==0 || Character.compare(b,'b')==0)
                f="11";
        else if(Character.compare(b,'C')==0 || Character.compare(b,'c')==0)
                f="12";
        else if(Character.compare(b,'D')==0 || Character.compare(b,'d')==0)
                f="13";
        else if(Character.compare(b,'E')==0 || Character.compare(b,'e')==0)
                f="14";
        else if(Character.compare(b,'F')==0 || Character.compare(b,'f')==0)
                f="15";
        else
                //a+=g;
                f+=b;

        //System.out.println("A "+g+" "+f);
        //if(box==1)
                output+=sbox[Integer.parseInt(g)][Integer.parseInt(f)];
        //else if(box==2)
        //      output+=lbox[Integer.parseInt(g)][Integer.parseInt(f)];
        //else
        //      output+=ebox[Integer.parseInt(g)][Integer.parseInt(f)];
        return output;
}
String permuta(char a,char b,int c)
{
        //System.out.print(a+" "+b+" ");
        String output="",g="",f="";
```

```
                if(c!=2)
                {
                        //System.out.print("fgf");
                        if(c==0)
                        {
                                //g="0";
                                //f="0";
                                output+=sbox[0][0];
                        }
                        else
                        {
                                //g="0";
                                if(Character.compare(b,'A')==0 ||
Character.compare(b,'a')==0)
                                        f="10";
                                else if(Character.compare(b,'B')==0 ||
Character.compare(b,'b')==0)
                                        f="11";
                                else if(Character.compare(b,'C')==0 ||
Character.compare(b,'c')==0)
                                        f="12";
                                else if(Character.compare(b,'D')==0 ||
Character.compare(b,'d')==0)
                                        f="13";
                                else if(Character.compare(b,'E')==0 ||
Character.compare(b,'e')==0)
                                        f="14";
                                else if(Character.compare(b,'F')==0 ||
Character.compare(b,'f')==0)
                                        f="15";
                                else
```

```
                    f+=b;
                output+=sbox[0][Integer.parseInt(f)];
            }

}
else
{
if(Character.compare(a,'A')==0 || Character.compare(a,'a')==0)
            g="10";
else if(Character.compare(a,'B')==0 || Character.compare(a,'b')==0)
            g="11";
else if(Character.compare(a,'C')==0 || Character.compare(a,'c')==0)
            g="12";
else if(Character.compare(a,'D')==0 || Character.compare(a,'d')==0)
            g="13";
else if(Character.compare(a,'E')==0 || Character.compare(a,'e')==0)
            g="14";
else if(Character.compare(a,'F')==0 || Character.compare(a,'f')==0)
            g="15";
else
            g+=a;
if(Character.compare(b,'A')==0 || Character.compare(b,'a')==0)
            f="10";
else if(Character.compare(b,'B')==0 || Character.compare(b,'b')==0)
            f="11";
else if(Character.compare(b,'C')==0 || Character.compare(b,'c')==0)
            f="12";
else if(Character.compare(b,'D')==0 || Character.compare(b,'d')==0)
            f="13";
```

```
else if(Character.compare(b,'E')==0 || Character.compare(b,'e')==0)
        f="14";
else if(Character.compare(b,'F')==0 || Character.compare(b,'f')==0)
        f="15";
else
        //a+=g;
        f+=b;
output+=sbox[Integer.parseInt(g)][Integer.parseInt(f)];
}
return output;
}
String invpermuta(char a,char b,int c)
{
        //System.out.print(a+" "+b+" ");
        String output="",g="",f="";
        if(c!=2)
        {
                //System.out.print("fgf");
                if(c==0)
                {
                        //g="0";
                        //f="0";
                        output+=sbox[0][0];
                }
                else
                {
                        //g="0";
                        if(Character.compare(b,'A')==0 ||
Character.compare(b,'a')==0)
```

```
                                        f="10";
                                    else if(Character.compare(b,'B')==0 ||
Character.compare(b,'b')==0)
                                        f="11";
                                    else if(Character.compare(b,'C')==0 ||
Character.compare(b,'c')==0)
                                        f="12";
                                    else if(Character.compare(b,'D')==0 ||
Character.compare(b,'d')==0)
                                        f="13";
                                    else if(Character.compare(b,'E')==0 ||
Character.compare(b,'e')==0)
                                        f="14";
                                    else if(Character.compare(b,'F')==0 ||
Character.compare(b,'f')==0)
                                        f="15";
                                    else
                                        f+=b;
                                    output+=sbox[0][Integer.parseInt(f)];
                                }

                            }
                            else
                            {
                            if(Character.compare(a,'A')==0 || Character.compare(a,'a')==0)
                                g="10";
                            else if(Character.compare(a,'B')==0 || Character.compare(a,'b')==0)
                                g="11";
                            else if(Character.compare(a,'C')==0 || Character.compare(a,'c')==0)
                                g="12";
                            else if(Character.compare(a,'D')==0 || Character.compare(a,'d')==0)
```

```
            g="13";
        else if(Character.compare(a,'E')==0 || Character.compare(a,'e')==0)
            g="14";
        else if(Character.compare(a,'F')==0 || Character.compare(a,'f')==0)
            g="15";
        else
            g+=a;
        if(Character.compare(b,'A')==0 || Character.compare(b,'a')==0)
            f="10";
        else if(Character.compare(b,'B')==0 || Character.compare(b,'b')==0)
            f="11";
        else if(Character.compare(b,'C')==0 || Character.compare(b,'c')==0)
            f="12";
        else if(Character.compare(b,'D')==0 || Character.compare(b,'d')==0)
            f="13";
        else if(Character.compare(b,'E')==0 || Character.compare(b,'e')==0)
            f="14";
        else if(Character.compare(b,'F')==0 || Character.compare(b,'f')==0)
            f="15";
        else
            //a+=g;
            f+=b;
        output+=invsbox[Integer.parseInt(g)][Integer.parseInt(f)];
        }
        return output;
}
String[][] getKeys(String[] key)
{
        String keys[][] = new String[12][16];
```

```
int i=0,j,k;
for(j=0;j<16;j++)
{
	keys[0][j]=key[j];
}
for (i = 0; i < 10; i++)
{
	 k=0;
	String[] word=new String[5];
	for(j=12;j<16;j++)
	{
		word[k]=keys[i][j];
		k++;
	}
	word=leftCircularShift(word,1);
	k=0;
	for(j=12;j<16;j++)
	{
		word[k]=permutation(word[k]);
		k++;
	}
	String bin=hextoBin(word[0]);
	String con=hextoBin(rcon[i]);
	String res=xor(bin,con);
	res=binToHex(res);
	word[0]=res;
	j=0;
	for(k=0;k<4;k++)
	{
```

```
keys[i+1][j]=binToHex(xor(hextoBin(word[k]),hextoBin(keys[i][j])));
                        j++;
                }
                for(k=0;k<4;k++)
                {

keys[i+1][j]=binToHex(xor(hextoBin(keys[i+1][k]),hextoBin(keys[i][j])));
                        j++;
                }
                for(k=0;k<4;k++)
                {

keys[i+1][j]=binToHex(xor(hextoBin(keys[i+1][k+4]),hextoBin(keys[i][j])));
                        j++;
                }
                for (k=0;k<4;k++)
                {

keys[i+1][j]=binToHex(xor(hextoBin(keys[i+1][k+8]),hextoBin(keys[i][j])));
                j++;
                }
        }
        for(i=0;i<11;i++)
        {
                for (j=0;j<16;j++)
                {
                        System.out.print(keys[i][j]+" ");
                }
                System.out.println(" ");
```

```
        }
        return keys;
}
String[] converttohex(String text)
{
        String[] arr=new String[16];
        for(int i=0;i<16;i++)
        {
                char a=text.charAt(i);
                int val=(int)a;
                arr[i]=Integer.toHexString(val);
        }
        return arr;
}
String convertostr(String[][] pt)
{
        int i,j;
        String out="";
        for(i=0;i<4;i++)
        {
                for(j=0;j<4;j++)
                {
                        int val=Integer.parseInt(pt[j][i],16);
                        //System.out.print(val+" ");
                        char a=(char)val;
                        out+=a;
                }
        }
        return out;
```

```
	}
	String[][] matxor(String[][] key,String[][] pt)
	{
		String[][] xk=new String[4][4];
		//int k=0;
		for(int i=0;i<4;i++)
		{
			for(int j=0;j<4;j++)
			{

xk[i][j]=Integer.toHexString(Integer.parseInt(pt[i][j],16)^Integer.parseInt(key[i][j],16)
);
				//k++;
			}
		}
		return xk;
	}
	void encrypt(String plainText, String key)
	{
		int i,j,k=0;
		// get round keys
		String[] keyhex=converttohex(key);
		String[] plainhex=converttohex(plainText);
		String[][] keys = getKeys(keyhex);
		String[][] pt=new String[4][4];
		String[][] keynew=new String[4][4];
		String[][] plain=new String[11][16];
		for(i=0;i<16;i++)
		{
			plain[0][i]=plainhex[i];
```

```
}
System.out.println("Round state matrix");
//int l=0;
for(i=0;i<4;i++)
{
	for(j=0;j<4;j++)
	{
		pt[j][i]=plain[0][k];
		//plain[0][k]=plainhex[k];
		k++;
	}
}
k=0;
for(i=0;i<4;i++)
{
	for(j=0;j<4;j++)
	{
		keynew[j][i]=keys[0][k];
		k++;
	}
}
pt=matxor(keynew,pt);
for(int l=0;l<9;l++)
{
for(j=0;j<4;j++)
{
	for (k=0;k<4;k++)
	{
```

```
                        if(pt[j][k].length()<=1)
                        {
                                if(pt[j][k].equals("0"))
                                {
                                        pt[j][k]=permuta('a','a',0);
                                }
                                else
                                {
                                        pt[j][k]=permuta('a',pt[j][k].charAt(0),1);

                                        //System.out.println(pt[j][k]+" ");
                                }
                        }
                        else

pt[j][k]=permuta(pt[j][k].charAt(0),pt[j][k].charAt(1),2);
                }
                pt[j]=leftCircularShift(pt[j],j);
        }
        pt=mix(pt);
        k=0;
        for(i=0;i<4;i++)
        {
                for(j=0;j<4;j++)
                {
                        keynew[j][i]=keys[l+1][k];
                        k++;
                }
        }
        for(i=0;i<4;i++)
```

```
                {
                        for(j=0;j<4;j++)
                        {

        pt[i][j]=Integer.toHexString(Integer.parseInt(pt[i][j],16)^(Integer.parseInt(keynew[i][j
],16)));
                        }
                        //System.out.println(" ");
                }
                //k=0;
                for(i=0;i<4;i++)
                {
                        for(j=0;j<4;j++)
                        {
                                System.out.print(pt[j][i]+" ");
                                //plain[l+1][k]=pt[i][j];
                                //k++;
                        }
                        //System.out.println(" ");
                }
                System.out.println(" ");
        }
                for(j=0;j<4;j++)
                {
                        for (k=0;k<4;k++)
                        {

                                if(pt[j][k].length()<=1)
                                {
                                        if(pt[j][k].equals("0"))
```

```
                                        {
                                                pt[j][k]=permuta('a','a',0);
                                        }
                                        else
                                        {
                                                pt[j][k]=permuta('a',pt[j][k].charAt(0),1);

                                                //System.out.println(pt[j][k]+" ");
                                        }
                                }
                                else

pt[j][k]=permuta(pt[j][k].charAt(0),pt[j][k].charAt(1),2);
                        }
                        pt[j]=leftCircularShift(pt[j],j);
                }
                k=0;
                for(i=0;i<4;i++)
                {
                        for(j=0;j<4;j++)
                        {
                                keynew[j][i]=keys[10][k];
                                k++;
                        }
                }
                //}
                for(i=0;i<4;i++)
                {
                        for(j=0;j<4;j++)
                        {
```

```
pt[i][j]=Integer.toHexString(Integer.parseInt(pt[i][j],16)^Integer.parseInt(keynew[i][j]
,16));
                        }
                        //System.out.println(" ");
                }
                String cipher=convertostr(pt);
                //k=0;
                String output="";
                System.out.println("Cipher Text :");
                for(i=0;i<4;i++)
                {
                        for(j=0;j<4;j++)
                        {
                                System.out.print(pt[j][i]+" ");
                                //plain[9][k]=pt[i][j];
                                //k++;
                        }
                        //System.out.println(" ");
                }
                //System.out.println("\noutput"+cipher);

        }
        void decrypt(String key)
        {
                int i,j;
                // get round keys
                //String keys[] = getKeys(key);
                String[] keyhex=converttohex(key);
                String[] plainhex=new String[16];
```

```
Scanner scan=new Scanner(System.in);
System.out.println("Enter ciphertext:");
for(i=0;i<16;i++)
{
	plainhex[i]=scan.nextLine();
}
String[][] keys = getKeys(keyhex);
String[][] plain=new String[10][16];
String[][] pt=new String[4][4];
for(i=0;i<16;i++)
{
	plain[0][i]=plainhex[i];
}
System.out.println("Round state matrix");
int k=0;
for(i=0;i<4;i++)
{
	for(j=0;j<4;j++)
	{
		pt[j][i]=plain[0][k];
		//plain[0][k]=plainhex[k];
		k++;
	}
}
String[][] keynew=new String[4][4];
k=0;
for(i=0;i<4;i++)
{
	for(j=0;j<4;j++)
```

```
                {
                        keynew[j][i]=keys[9][k];
                        k++;
                }
        }
        pt=matxor(keynew,pt);
        for(int l=0;l<9;l++)
        {
        for(j=0;j<4;j++)
        {
                pt[j]=invleftCircularShift(pt[j],j);
                for (k=0;k<4;k++)
                {

                        if(pt[j][k].length()<=1)
                        {
                                if(pt[j][k].equals("0"))
                                {
                                        pt[j][k]=invpermuta('a','a',0);
                                }
                                else
                                {

pt[j][k]=invpermuta('a',pt[j][k].charAt(0),1);
                                        //System.out.println(pt[j][k]+" ");
                                }
                        }
                        else

pt[j][k]=invpermuta(pt[j][k].charAt(0),pt[j][k].charAt(1),2);
```

```
                }
                //pt[j]=leftCircularShift(pt[j],j);
        }
        k=0;
        for(i=0;i<4;i++)
        {
                for(j=0;j<4;j++)
                {
                        keynew[j][i]=keys[8-l][k];
                        k++;
                }
        }
        for(i=0;i<4;i++)
        {
                for(j=0;j<4;j++)
                {

pt[i][j]=Integer.toHexString(Integer.parseInt(pt[i][j],16)^(Integer.parseInt(keynew[i][j
],16)));
                }
        }
        pt=invmix(pt);
        //k=0;
        for(i=0;i<4;i++)
        {
                for(j=0;j<4;j++)
                {
                        System.out.print(pt[j][i]+" ");
                        //plain[l+1][k]=pt[i][j];
                        //k++;
```

```
                }
                //System.out.println(" ");
            }
            System.out.println(" ");
        }
        for(j=0;j<4;j++)
            {
                pt[j]=invleftCircularShift(pt[j],j);
                for (k=0;k<4;k++)
                {

                    if(pt[j][k].length()<=1)
                    {
                        if(pt[j][k].equals("0"))
                        {
                            pt[j][k]=invpermuta('a','a',0);
                        }
                        else
                        {

pt[j][k]=invpermuta('a',pt[j][k].charAt(0),1);
                            //System.out.println(pt[j][k]+" ");
                        }
                    }
                    else

pt[j][k]=invpermuta(pt[j][k].charAt(0),pt[j][k].charAt(1),2);
                }

            }
```

```
k=0;
for(i=0;i<4;i++)
{
        for(j=0;j<4;j++)
        {
                keynew[j][i]=keys[0][k];
                k++;
        }
}
//}
for(i=0;i<4;i++)
{
        for(j=0;j<4;j++)
        {

pt[i][j]=Integer.toHexString(Integer.parseInt(pt[i][j],16)^Integer.parseInt(keynew[i][j]
,16));
        }
        //System.out.println(" ");
}
//k=0;
String output="";
System.out.println("Plain Text :");
for(i=0;i<4;i++)
{
        for(j=0;j<4;j++)
        {
                System.out.print(pt[j][i]+" ");
                //plain[9][k]=pt[i][j];
                //k++;
```

```
                        }
                        //System.out.println(" ");
                    }
                    System.out.println(" ");
            }
   }
   public static void main(String args[])
   {
            String key,cipherText,plainText;
  int choice;
do{
   Scanner scn=new Scanner(System.in);
  System.out.println("\nEnter choice 1)encrypt 2)Decrypt 3)exit");
choice=scn.nextInt();
switch(choice)
{
  case 1:
  {
  Scanner sc=new Scanner(System.in);
  System.out.println("Enter plaintext:");
  plainText=sc.nextLine();
  System.out.println("Enter Key:");
  key=sc.nextLine();
  AES cipher = new AES();
  System.out.println("Encryption:\n");
                    cipher.encrypt(plainText, key);
  break;
}
case 2:
```

```
        {
          Scanner scan=new Scanner(System.in);
          System.out.println("Enter Key:");
          key=scan.nextLine();
          AES cipher = new AES();
          System.out.println("Decryption\n");
                              cipher.decrypt(key);
         break;
        }
      }
    }while(choice!=3);


          }
}
```

SCREENSHOT:

Niranjana S
312217205059

```
C:\Users\Niranjana>java aes

Enter choice 1)encrypt 2)Decrypt 3)exit
1
Enter plaintext:
Two one Nine Two
Enter Key:
Thats my Kung Fu
Encryption:

54 68 61 74 73 20 6d 79 20 4b 75 6e 67 20 46 75
e2 32 fc f1 91 12 91 88 b1 59 e4 e6 d6 79 a2 93
56 08 20 07 c7 1a b1 8f 76 43 55 69 a0 3a f7 fa
d2 60 0d e7 15 7a bc 68 63 39 e9 01 c3 03 1e fb
a1 12 02 c9 b4 68 be a1 d7 51 57 a0 14 52 49 5b
b1 29 3b 33 05 41 85 92 d2 10 d2 32 c6 42 9b 69
bd 3d c2 87 b8 7c 47 15 6a 6c 95 27 ac 2e 0e 4e
cc 96 ed 16 74 ea aa 03 1e 86 3f 24 b2 a8 31 6a
8e 51 ef 21 fa bb 45 22 e4 3d 7a 06 56 95 4b 6c
bf e2 bf 90 45 59 fa b2 a1 64 80 b4 f7 f1 cb d8
28 fd de f8 6d a4 24 4a cc c0 a4 fe 3b 31 6f 26
Round state matrix
58 47 8 8b fb c1 6b 23 59 d4 e2 e8 cd 39 df ce
19 aa 9f 54 a2 1 96 32 db da 86 41 de c6 a2 d4
e9 c8 f5 28 82 80 c5 b 35 a6 2c f8 5 75 a fd
f4 4a cf ff ad ae 1b bf 91 fc f6 2c 6 42 5e 2a
44 66 d0 9e b1 51 2a f2 3d 44 68 1c a7 1 f4 43
bc 4a ac a aa 53 ec e4 59 aa ae 4b 38 bc f 5f
1 ca fc 35 c9 f8 ef f8 c4 46 13 44 a0 e2 a9 6
a7 47 99 47 f0 2a de e0 9e 80 46 88 d5 fc 57 ca
1d f5 2e 23 f1 19 f6 f0 73 dc c4 2e d3 4a 5 f3
Cipher Text :
8c 29 c2 f5 cc 22 4f 6c 43 16 95 72 5d d7 2d 17
```