International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-8 Issue-12, October 2019

# Pythagorean Triples in Cryptography and Associated Networks

**V. Yegnanarayanan, Poojitha Yakkala**

***Abstract*: Any organization is obliged to ensure secrecy of data from hacking criminals complying with the increasing demand for secured data. So data preservation is indispensablethrough cryptographic methods. It is adoptedseveralreal life applications such as ecommerce, In this paper we indicated a procedure for generating different Pythagorean triples and with the help of C++ coding developed a mechanism for both encoding and decoding of a plain text in English alphabets. We also demonstrated them with illustrative examples.**

***Keywords* : Pythagorean Triple, Encoding, Decoding, Wireless Sensor Networks (WSN).**

## I. INTRODUCTION

Is is a challenge to devise an efficient procedurefor key generation. The author in [MonishaPrabhu. (2013)] proposed one such scheme through Primitive Pythagorean triples (PPTs) as pointed outin [Kak (2010)]. In [ArtanLuma and BujarRaufi. (2014)] the authors suggested a Pythagorean Triple Algorithm. Using this they extended the definition of the Pythagorean Theorem. That is, there exist two or sometimes even three solutions to Pythagorean Theorem for any two numbers p and q such that either $p \in 2Z$ and $q \in 2Z+1$ or vice versa. Based on these they created an encryption and decryption key for simple symmetric cryptosystem.

It is interesting to know that there are many methods available to obtain Pythagorean triples (PT). In [Duvvuri Surya Rahul and SnehanshuSaha (2015)] the authors highlight a structured approach to find all PPTs with no repeat digit in the triples. A connection between the sum of side lengths and prime numbers are observed of a primitive right triangle. A Fundamental result about Primitive Pythagorean Primes (PPP) on the lines of Fundamental Theorem of Arithmetic is conjectured. One cannot find a variety of methods in the literature to identify all PPTs and PTs with no repeat digit. In [Barning (1963)] the author developed a nice procedure for determining such triples. Price [Price (2008)] later came out with his own method.One can also refer to [Saunders and Randall (1994)], Mitchell's formula [Mitchell (2001)]and [William (2004)] for more on this.

**Revised Manuscript Received on October 30, 2019.**
***** Correspondence Author
**V. Yegnanarayanan***, Department of Mathematics, School of Arts, Science, SASTRA Deemed to be University, Thanjavur, India, 613401. Email: prof.yegna@gmail.com
**Poojitha Yakkala**, School of Computing, SASTRA Deemed to be University, Thanjavur, India, 613401. Email: pooji6453@gmail.com



There are several analytical and numerical methods [Mushtaque and Snehanshu (2014)] for estimating prime numbers. Cryptographic algorithms are divided into two types viz., symmetric-key and public key algorithms. In symmetric-key algorithm, for both encryption and decryption same key is adoptedand it has be sent via a communication channel or sometimes manually. A drawback is that a data thief may get an accessto the secret key through illegal means during transmission. Symmetric key algorithms are further divided into stream cipher and block cipher algorithms. Symmetric-key encryption algorithm through block cipheris less preferred than to a cipherstreamon smaller units of plaintext, usually bits or bytes. Stream ciphers are much faster than block ciphers. Most of the stream ciphers facing the problem of generating one random bit in each round of process as the output stream of cryptosystem [Majid Bakhtiari and MohdAizainiMaarof. (2011)]. This increases the risk of algebraic correlation against those cryptosystems[Majid Bakhtiari and MohdAizainiMaarof. (2011),Meier and Staffelbach, (1990)]. Vernam's one-time pad is one of the stream cipher involving a string of randomly generated bits.As the whole keystream is random, an opponent can guess the plaintext. It is noted that though, Vernam's onetime pad is perfectly secured, remembering and storing such a key is too tedious because the size of the key is always taken as the size of plaintext and hence it is at least practical [Charles Pfleeger and Shari Lawrence Pfleeger. (2007)]. In DES symmetric-key algorithm, initially 64-bit key called DES-key is given as input and in each round a 56-bit round key has been generated to produce the ciphertext. Suppose, if the DES keys chosen are weak keys, the Hamming distance between a plaintext and a ciphertext produced by DES may be less giving the data thief a chance to recover from the ciphertextthe intended plaintext. To overcome these, a PPT based key stream can be generated and its length can be determined by both the sender and the receiver as in the case of Vernam's one time pad. Further to develop from the PPT based keystream a new DES key, the crucial seed PPT and beginning position should be known by the participatingparties and this aspect makes the keystream fromthe key sequence unpredictable. This makes the the PPT based keystream better than traditional DES as in the former different key is generated for each round and in the latterin all rounds only one key is used.

## II. PYTHAGOREAN TRIPLES

Generation of events of with given probability is pertinent in cryptography [Kolmogorov (1965), Kak, and Chatterjee (1981), Kak (1985), Kak (1987) Kak (1989), Kak (2007)].

One can produce infinite number of Pythagorean triples of the form x (a, b, c) where x>1 and such triples have the ability of creating probability events. A primitive Pythagorean triple (a, b, c) is the one in which a, b and c are pairwise relatively prime. For a detailed description see [Kak (2010),Euclid (1997), Heath (1956)].
Kak (1987), Kak (2011), O'Conner and Robertson, (2000), Seidenberg (1978),Parakh and Kak(2009).Parakh and Kak(2011)].

## III. PYTHAGOREAN TRIPLES FOR ENCRYPTION AND DECRYPTION

An ordered triple (a,b,c) $\in Z^3$ is called a Pythagorean triple if $a^2+b^2=c^2$.One familiar approach to generate a Pythagorean triple is: for any x, y$\in Z^+$ with x>y,set $a=x^2-y^2$, $b=2xy$, $c=x^2+y^2$.This formula is due to Euclid. Then Newton opined an integer solution: a is d times $(x^2-y^2)$,bis 2d times xy,c is d times $(x^2+y^2)$ with x, y>0, x > y,gcd(x,y)=1, x and y differ in parity and d is the gcd of (a,b,c). We call the triple (a,b, c) to be Primitive Pythagorean if d=1. Suppose that $a^2+b^2=c^2$ and gcd(a,b)=1.Then there exists r,s$\in$ Z such that c=a+r, c=b+s, where gcd(a, r)=1 an gcd(b,s)=1. Clearly a+r=b+s and a-s=b-r. Set a-s=b-r=μ. So that a=s+μ and b=r+μ.Then we get c=r+s+μ.So (a=s+μ, b=r+μ, c=r+s+μ) standsfor new solution to the system $a^2+b^2=c^2$ .That is,$(r+\mu)^2+(s+\mu)^2=(r+s+\mu)^2$ .This gives $\mu^2=2rs$.Here one can pick r and s in such away that we deduce a solution of the type $s=2x^2$,$r=y^2$,s>r,gcd(x,y) =1.Putting these values in $\mu^2=2rs$,we get $\mu^2=4x^2y^2$ and hence$\mu=\pm 2xy$.Hence we get $a=2x^2\pm 2xy$ ,$b=y^2\pm 2xy$,$c=2x^2+y^2\pm 2xy$.This reveals a fact that for any two given integers x and y, with either x $\in$ 2Z and y $\in$ 2Z+1 or vice versathere are $(a_1,b_1,c_1)$ and $(a_2,b_2,c_2)$ that acts as solutions.

**Example 1**: If x=3,y=1 then we have two solutions $a_1=24$,$b_1=7$,$c_1=25$ and $a_2=12$,$b_2=-5$,$c_2=13$.

We now use the above said procedure to encrypt and decrypt a text. If 't' stands for plaintext, α for the key and e for the encrypted message then e=t+α(mod 26) where the alphabets a to z are represented by 0 to 25, in order.To decrypt the above we have to perform t=e-α(mod 26) and the key is generated by using the values.$x_1=2x^2+2xy$,$x_2=2x^2-2xy$, $x_3=2xy$;$y_1=y^2+2xy$, $y_2=y^2-2xy$, $y_3=x^2-y^2$; $z_1=2x^2+y^2+2xy$, $z_2=2x^2+y^2-2xy$, $z_3=x^2+y^2$ and $(x_1,y_1,z_1)$, $(x_2,y_2,z_2)$, $(x_3,y_3,z_3)$ is reduced with mod 26 and create an encryption key of the type: $x_1$, $y_1$, $z_1$, $x_2$, $y_2$, $z_2$, $x_3$, $y_3$, $z_3$.
**Example 2**: Suppose that we have SASTRA DEEMED TO BE UNIVERSITYas plaintext.

Table.1 shows the message encoding for the above word.

**Table 1 Message Encoding**

| S | A | S | T | R | A | D | E | E | M | E | D | T | O | B | E |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 18 | 0 | 18 | 19 | 17 | 0 | 3 | 4 | 4 | 12 | 4 | 3 | 19 | 14 | 1 | 4 |

| U | N | I | V | E | R | S | I | T | Y |
|---|---|---|---|---|---|---|---|---|---|
| 20 | 13 | 8 | 21 | 4 | 17 | 18 | 8 | 19 | 24 |

We encrypt using x=7,y=2. Simple calculation reveals $x_1=2.7^2+2.7.2=126$; $y_1=2^2+2.7.2=32$; $z_1=2.7^2+2^2+2.7.2=130$; $x_2=2.7^2-2.7.2=70$; $y_2=2^2-2.7.2=-24$; $z_2=2.7^2+2^2-2.7.2=74$;$x_3=2.7.2=28$; $y_3=7^2-2^2=25$; $z_3=7^2+2^2=53$;So, applying (mod 26) on (126,32,130,70,-24,74,28,45,53) we get the key (22,6,0,18,2,22,2,19,1). As the key is created it is simple to decrypt the encrypted message through t=e-α(mod 26). Table 2 shows message encryption. Table 3 shows message decryption.

**Table 2 Message Encryption**

| S | A | S | T | R | A | D | E | E | M | E | D | T | O | B | E |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 18 | 0 | 18 | 19 | 17 | 0 | 3 | 4 | 4 | 12 | 4 | 3 | 19 | 14 | 1 | 4 |
| 22 | 6 | 0 | 18 | 2 | 22 | 2 | 19 | 1 | 22 | 6 | 0 | 18 | 2 | 22 | 2 |
| 14 | 6 | 18 | 11 | 19 | 22 | 5 | 23 | 5 | 8 | 10 | 3 | 11 | 16 | 23 | 6 |
| O | G | S | L | T | W | F | X | F | I | K | D | L | Q | X | G |

| U | N | I | V | E | R | S | I | T | Y |
|---|---|---|---|---|---|---|---|---|---|
| 20 | 13 | 8 | 21 | 4 | 17 | 18 | 8 | 19 | 24 |
| 19 | 1 | 22 | 6 | 0 | 18 | 2 | 22 | 2 | 19 |
| 13 | 14 | 4 | 1 | 4 | 9 | 20 | 4 | 21 | 17 |
| N | O | E | B | E | J | U | E | V | R |

Note that Newton's formula for generating Pythagorean triples sometimes duplicates Pythagorean triples.For instance, if d=1,x=9,y=3;d=9,x=3,y=1; andd=18, x=2, y=1 results in (a,b,c)=(72,54,90);(a,b,c)=(54,72,90). In literature only a few methods exists that produce distinct Pythagorean triples.

**Table 3 Message Decryption**

| O | G | S | L | T | W | F | X | F | I | K | D | L | Q | X | G |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 14 | 6 | 18 | 11 | 19 | 22 | 5 | 23 | 5 | 8 | 10 | 3 | 11 | 16 | 23 | 6 |
| 22 | 6 | 0 | 18 | 2 | 22 | 2 | 19 | 1 | 22 | 6 | 0 | 18 | 2 | 22 | 2 |
| 18 | 0 | 18 | 19 | 17 | 0 | 3 | 4 | 4 | 12 | 4 | 3 | 19 | 14 | 1 | 4 |
| S | A | S | T | R | A | D | E | E | M | E | D | T | O | B | E |

| N | O | E | B | E | J | U | E | V | R |
|---|---|---|---|---|---|---|---|---|---|
| 13 | 14 | 4 | 1 | 4 | 9 | 20 | 4 | 21 | 17 |
| 19 | 1 | 22 | 6 | 0 | 18 | 2 | 22 | 2 | 19 |
| 20 | 13 | 8 | 21 | 4 | 17 | 18 | 8 | 19 | 24 |
| U | N | I | V | E | R | S | I | T | Y |

**Algorithm for Encoding**

Algorithm ENCODE(List)
This algorithm reads the input and prints a report.
Pre:List contains the plain text entered by the user.
Post:Cipher text will be displayed.
1.if(List not equal to ".")
  1.read List
  2.increment count
2.end if
3.decrementcount,set count1 to count.
4.loop(count1 not equal to 0)

1.convert list elements to respective ascii values and set to convertedlist.
2.decrement count.
5.end loop
6.set randomvalue1 to value1.
7.set randomvalue2 to value2.
8.setlistP to set1 pythagarean values.
9.setlistQ to set2 pythagarean values.
10.setlistR to set3 pythagarean values.
since,onlyine values.....
11.loop(index<3)
1.if(element of listP>=0)
1.setlistP element to mod listP element.
2.else
1.incrementlistP element by 26.
3.end if
4.if(element of listQ>=0)
1.setlistQ element to mod listQ element.
5.else
1.incrementlistQ element by 26.
6.end if
7.if(element of listR>=0)
1.setlistR element to mod listR element.
8.else
1.incrementlistR element by 26.
9.end if
10.increment index.
12.end loop
since, only 3 sets 9 with elements.....
13.setlistC to all the sets elements respectively.
14.set count2 to count.
15.set index to 0
16.loop1(count2 not equal to 0)
1.if(index <9)
1.newlist[count]=convertlist[count]+listC[index]

2.else
1.set index to index mod 9
2.newlist[count]=convertlist[count]+listC[index]
3.end if
4.increment index
5.increment count
6.decrement count2
17.end loop1
18.loop3(count not equal to 0)
1.if(newlist element >25)
1.decrementnewlist element with 26
2.end if
3.decrement count
19.end loop3
20.set choice to userchoice
21.if(choice is equal to UPPERCASE)
1.incrementnewlist to 65
2.convertnewlist elements to respective ascii characters.
3.displaynewlist
22.else if(choice is equal to LOWERCASE)
1.incrementnewlist to 97
2.convertnewlist elements to respective ascii characters.
3.displaynewlist
23.else
1.go to statement 20
24.end if
end ENCODE

**Algorithm for Decoding**

Algorithm DECODE(List)
This algorithm reads the input and prints a report.
Pre:List contains the cipher text.
Post:Plain text will be displayed.
1.if(List not equal to ".")
1.read List
2.increment count
2.end if
3.decrementcount,set count1 to count.
4.loop(count1 not equal to 0)
1.convert list elements to respective ascii values and set to convertedlist.
2.decrement count.
5.end loop
6.set randomvalue1 to value1.
7.set randomvalue2 to value2.
8.setlistP to set1 pythagarean values.
9.setlistQ to set2 pythagarean values.
10.setlistR to set3 pythagarean values.
since,only nine values.....
11.loop(index<3)
1.if(element of listP>=0)
1.setlistP element to mod listP element.
2.else
1.incrementlistP element by 26.
3.end if
4.if(element of listQ>=0)
1.setlistQ element to mod listQ element.
5.else
1.incrementlistQ element by 26.
6.end if
7.if(element of listR>=0)
1.setlistR element to mod listR element.
8.else
1.incrementlistR element by 26.
9.end if
10.increment index
12.end loop
since, only 3 sets 9 with elements.....
13.setlistC to all the sets elements respectively.
14.set count2 to count.
15.set index to 0
16.loop1(count2 not equal to 0)
1.if(index <9)
1.if(convertedlist[count]>=listC[index])
1.newlist[count]=convertlist[count]-listC[index]
2.else
1.newlist=convertlist[count]-listC[index]+26
3.end if
2.else
1.set index to index mod 9
2.if(convertedlist[count]>=listC[index])
1.newlist[count]=convertlist[count]-listC[index]
3.else
1.newlist=convertlist[count]-listC[index]+26
4.end if
3.end if
4.increment index
5.increment count
6.decrement count2
17.end loop1
18.set choice to userchoice

19.if(choice is equal to UPPERCASE)
  1.incrementnewlist to 65
  2.convertnewlist elements to respective ascii characters.
  3.displaynewlist
20.else if(choice is equal to LOWERCASE)
  1.incrementnewlist to 97
  2.convertnewlist elements to respective ascii characters.
  3.displaynewlist
21.else
  1.go to statement 18
22.end if
end DECODE

C++ Program For Encoding

```
#include<iostream>
#include<iomanip>
using namespace std;
int main()
{
        int p[3],q[3],r[3],x,y,co=1,n,i,j,flag,d[100],k=0;
        char a[1000],b[100];
        cout<<"enter the letters with a "." at the end:"<<endl;
        for(i=0;a[i-1]!='.';i++)
        {cin>>a[i];}
for(i=0;a[i]!='.';i++)
        {
                co++;
        }
        n=co-1;
        cout<<n;
        for(i=0;i<n;i++)
        { if((int)a[i]>96)
        {

        b[i]=((int)a[i]-97);
}
        else
{
                b[i]=((int)a[i]-65);}
                }
      cout<<"enter value1,value2(x,y)";
        cin>>x;
        cin>>y;
      for(i=0;i<3;i++)
      {if(i==0)
     {
     p[i]=((2*x*x)+(2*x*y));
     q[i]=((y*y)+(2*x*y));
     r[i]=((2*x*x)+(y*y)+(2*x*y));
     }
     if(i==1)
     {
     p[i]=((2*x*x)-(2*x*y));
     q[i]=((y*y)-(2*x*y));
     r[i]=((2*x*x)+(y*y)-(2*x*y));
     }
     if(i==2)
     {
     p[i]=(2*x*y);
     q[i]=((x*x)-(y*y));
     r[i]=((x*x)+(y*y));
}
}
for(i=0;i<3;i++)
cout<<p[i]<<" ";
cout<<endl;
for(i=0;i<3;i++)
cout<<q[i]<<" ";
cout<<endl;
for(i=0;i<3;i++)
cout<<r[i]<<" ";
cout<<endl;
        for(i=0;i<3;i++)
        {if(p[i]>=0)
```

```
        p[i]=p[i]%26;
        else
      p[i]=p[i]+26;
        if(q[i]>=0)
        q[i]=q[i]%26;
        else
      q[i]=q[i]+26;
        if(r[i]>=0)
        r[i]=r[i]%26;
        else
      r[i]=r[i]+26;
      }
int c[10];
c[0]=p[0];c[1]=q[0];c[2]=r[0];c[3]=p[1];c[4]=q[1];
c[5]=r[1];c[6]=p[2];c[7]=q[2];c[8]=r[2];
for(i=0;i<9;i++)
cout<<c[i]<<endl;
j=0;
for(i=0;i<n;i++)
{
if(j<9)
d[i]=b[i]+c[j];
else
{j=(j%9);
d[i]=b[i]+c[j];
}
j++;
}
cout<<"finally"<<endl;
for(i=0;i<n;i++)
{if(d[i]>25)
d[i]=d[i]-26;
else
{
d[i]=d[i];
int flag1=0;}
}
 cout<<endl;
 char ch;
 cout<<"enter ur choice as either l(lower) and u(upper) case
letters ";
     cin>>ch;
     while(ch!='U'&&ch!='u'&&ch!='l'&&ch!='L')
     {
{
cout<<"you have been entered a wrong option change it.";
cin>>ch;
cout<<endl;}
}
cout<<endl;
for(i=0;i<n;i++)
    {
        cout<<setw(2)<<(char)a[i]<<" ";
    }
     cout<<endl;
for(i=0;i<n;i++)
   {cout<<setw(2)<<(int)b[i]<<" ";
   }
      cout<<endl;
for(i=0;i<n;i++)
```

```
        {cout<<setw(2)<<d[i]<<" ";
        }
        cout<<endl;

        if(ch=='l'||ch=='L')
{
                for(i=0;i<n;i++)
     {  d[i]=d[i]+97;
        cout<<setw(2)<<(char)d[i]<<" ";
     }
}
else
{
for(i=0;i<n;i++)
     {  d[i]=d[i]+65;
        cout<<setw(2)<<(char)d[i]<<" ";
     }
}
        return 0;
}
```

```
enter the letters with a '.' at the end:
S A S T R A D E E M E D T O B E U N I V E R S I T Y .
26enter value1,value2(x,y)7 2
126 70 28
32 -24 45
130 74 53
22
6
0
18
2
22
2
19
1
finally

enter ur choice as either l(lower) and u(upper) case letters U

 S  A  S  T  R  A  D  E  E  M  E  D  T  O  B  E  U  N  I  V  E  R  S  I
T  Y
18  0 18 19 17  0  3  4  4 12  4  3 19 14  1  4 20 13  8 21  4 17 18  8
19 24
14  6 18 11 19 22  5 23  5  8 10  3 11 16 23  6 13 14  4  1  4  9 20  4
21 17
 O  G  S  L  T  W  F  X  F  I  K  D  L  Q  X  G  N  O  E  B  E  J  U  E
V  R
--------------------------------
```

## IV. DISCUSSION ON PYTHAGOREAN TRIPLES AND ITS ROLE IN WSN

The C++ program codes given below makes use of Pythagorean triples, where the input words given in text form by the user are converted into corresponding numerical values. According to Newton's formula for each set of two different numbers there exists three sets of Pythagorean triples. For Encoding: The input information which are converted into numerical value are manipulated according to the formula t = e-α (mod 26) using Pythagorean triples. Then numerical values are displayed as alphabets either as upper case or as lower case as per the user choice. For Decoding: We first consider the numeric value that corresponds to the entered code. Then by adding the manipulated set of Pythagorean triples to the numerical values as per the condition where necessary, we adjust the numerical values to the alphabetical number range to decode the input information.

International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-8 Issue-12, October 2019

One can see in the C++ code that alphabets entered are converted into corresponding numerical values. Then user will exercise his choice of the set of two numbers to generate three sets of Pythagorean triples. Then by manipulating the numerical values through the Pythagorean triples one can derive the encoded input. Similarly, to decode the encoded information one can convert the entered alphabets to corresponding numerical values and with the help of generated Pythagorean triples read the message.

One can view in literature how to produce efficientkeys for instance [Blom (1985), or Chan et.al (2003), Eschenauer and Gligor (2002)]who take into account variety of factors such as energy, memory and cost etc., The generation of Pythagorean triples also finds immense potential for application in developing key management schemes in WSN. WSN are vital for smart environments to achieve automation in facilities such as transportation, home, utilities and industries. For this we look to sensory data from real world. We depend on distributed WSN for collecting such data and we process it to elicit crucial information. With the threat of stealing, we intend to minimize loss on the nearby nodes and the network. To ensure this messages exchanged between two nodes should adhere to proper encryption and authentication. To begin a communication a sender and the receiver must have a prior knowledge about common key. Pythagorean triples are involved in good depth to derive certain novel approaches such as symmetric-key algorithms for generating the key from a key stream. The main advantage of this is that the key value generated from the key stream is chosen by both the sender and the receiver. Also the key sequence size is arbitrary in length. As the generated key stream is random both the sender and the receiver need not remember such keys. We hope to revert more on this elsewhere.

## V. CONCLUSION

We have given a crisp introduction about the Pythagorean triplet generation mechanisms available in the literature and then indicated a new procedure to obtain different sets of Pythagorean triples for the same set of input values. Then by using the Number theory formulae we have outlined a procedure to encode and decode any word given in English alphabets.

C++ Program for Decoding

```
#include<iostream>
#include<iomanip>
using namespace std;
int main()
{
      int p[3],q[3],r[3],c[10],x,y,co=1,n,i,j,flag,d[100],k=0;
      char a[1000],b[100],ch;
      //taking the no.of letters are going to be entered.
            cout<<"enter the letters with a '.' at the end:"<<endl;
            for(i=0;a[i-1]!='.';i++)
      {cin>>a[i];}
for(i=0;a[i]!='.';i++)
      {
            co++;
      }
      n=co-1;
      cout<<n;
      //reading the data.
      //converting ascii value of the entered letters to alphabetical
order.
      for(i=0;i<n;i++)
      { if((int)a[i]>96)
      b[i]=((int)a[i]-97);
      else
      b[i]=((int)a[i]-65);}
       cout<<"enter value1,value2(x,y)";
      cin>>x;
      cin>>y;
      //by using pythagorean triplet rule:
     for(i=0;i<3;i++)
     {if(i==0)
    {
    p[i]=((2*x*x)+(2*x*y));
    q[i]=((y*y)+(2*x*y));
    r[i]=((2*x*x)+(y*y)+(2*x*y));
    }
    if(i==1)
    {
    p[i]=((2*x*x)-(2*x*y));
    q[i]=((y*y)-(2*x*y));
    r[i]=((2*x*x)+(y*y)-(2*x*y));
    }
    if(i==2)
    {
    p[i]=(2*x*y);
    q[i]=((x*x)-(y*y));
    r[i]=((x*x)+(y*y));
 }
 }
 cout<<endl<<"the sets of three triplets are:"<<endl;
 for(i=0;i<3;i++)
 cout<<p[i]<<" ";
 cout<<endl;
 for(i=0;i<3;i++)
 cout<<q[i]<<" ";
 cout<<endl;
 for(i=0;i<3;i++)
 cout<<r[i]<<" ";
 cout<<endl;
 //by the above deduction:
```

```
for(i=0;i<3;i++)
      {if(p[i]>=0)
      p[i]=p[i]%26;
      else
    p[i]=p[i]+26;
      if(q[i]>=0)
      q[i]=q[i]%26;
      else
    q[i]=q[i]+26;
      if(r[i]>=0)
      r[i]=r[i]%26;
      else
    r[i]=r[i]+26;
    }
c[0]=p[0];c[1]=q[0];c[2]=r[0];
c[3]=p[1];c[4]=q[1];c[5]=r[1];c[6]=p[2];
c[7]=q[2];c[8]=r[2];
for(i=0;i<9;i++)
cout<<c[i]<<endl;
j=0;
for(i=0;i<n;i++)
{
      if(j<9)
  {
      if(b[i]>=c[j])
      d[i]=b[i]-c[j];
      else
      d[i]=(b[i]+26)-c[j];
}
else
{
      j=j%9;
      if(b[i]>=c[j])
      d[i]=b[i]-c[j];
      else
      d[i]=(b[i]+26)-c[j];
}
    j++;
}
cout<<endl;

    cout<<"enter ur choice as either l(lower) and u(upper) case letters
";
    cin>>ch;
    while(ch!='U'&&ch!='u'&&ch!='l'&&ch!='L')
    {
{
            cout<<"you have been entered a wrong option change it .";
       cin>>ch;
      cout<<endl;}
}
cout<<endl;
for(i=0;i<n;i++)
{
      cout<<setw(2)<<(char)a[i]<<" ";
}
cout<<endl;
for(i=0;i<n;i++)
{cout<<setw(2)<<(int)b[i]<<" ";
}
```



## REFERENCES


1. Artan Luma and Bujar Raufi, "Data Encryption and Decryption Using New Pythagorean Triple Algorithm," Proceedings of the World Congress on Engineering (WCE 2014), vol I (2014) London, U.K.
2. F.J.M. Barning, "About Pythagorean and nearPythagorean triangles and a generation process using matrices unimodulaire (in Dutch)," Math. Centrum Amsterdam Afd. Zuivere Wisk. ZW-011 37 (1963).
3. R. Blom, "An Optimal Class of Symmetric Key Generation Systems", in Advances in Cryptology: EUROCRYPT'84, LNCS, vol. 209 (1963), pp. 335–338.
4. H. Chan, A. Perrig, D. Song, "Random key predistribution schemes for sensor networks", Proc. IEEE Symp. On Security and Privacy (2003), pp. 197-213.
5. Charles Pfleeger and Shari Lawrence Pfleeger, "Security in computing", Fourth Edition, Prentice Hall of India Pvt Ltd, New Delhi, 2007.
6. Duvvuri Surya Rahul and SnehanshuSaha, "A Discussion on Primitive Pythagorean Triples and Primitive Pythagorean Primes" (2015). https://www.researchgate.net/publication/282610987
7. L. Eschenauer and V.D. Gligor, "A key-management scheme for distributed sensor networks", Proc. ACM Conf. on Computer and Commun. Security (2002) pp. 41–47.
8. Euclid, "Works" (1997) http://aleph0.clarku.edu/~djoyce/java/elements/elements.html
9. T. Heath (ed.), "The Thirteen Books of Euclid's Elements", Dover Publications (1956).
10. S. Kak, "Encryption and error-correction coding using D sequences", IEEE Transactions on Computers, vol. C-34 (1985) pp. 803-809.
11. S. Kak, "New results on d-sequences," Electronics Letters, vol. 23 (1987) p. 617.
12. S. Kak, "On the chronology of ancient India", Indian Journal of History of Science, vol. 22 (1987) 222–234.
13. S. Kak, "A new method for coin flipping by telephone", Cryptologia, vol. 13 (1989) pp. 7378, 33.
14. S. Kak, "A cubic public-key transformation," Circuits, Systems and Signal Processing, vol. 26 (2007), pp. 353-359.
15. S. Kak, "Pythagorean Triples and Cryptographic Coding", 2010 (2010). arXiv:1004.3770
16. S. Kak, "The Astronomical Code of the Rigveda", Oklahoma State University, Stillwater (2011).
17. S. Kak and A. Chatterjee, "On decimal sequences", IEEE Transactions on Information Theory, vol. IT-27 (1981) pp. 647–652.
18. A. Kolmogorov, "Three approaches to the quantitative definition of information", Problems of Information Transmission, vol. 1 (1965), pp. 1–17.
19. Majid Bakhtiari and MohdAizainiMaarof, "An Efficient Stream Cipher Algorithm for Data Encryption", IJCSI International Journal of Computer Science Issues, vol. 8(3)(1) (2011) ISSN (Online): 1694-0814 www.IJCSI.org.
20. W. Meier and O. Staffelbach, "Nonlinearity Criteria for Cryptographic Functions, Advances in Cryptology", EUROCRYPT '89, J-J. Quisquater and J. Vandewalle, Editors, Springer Berlin /Heidelberg (1990) pp. 549–562.
21. D.W. Mitchell, "An alternative characterization of all primitive Pythagorean triples", Mathematical Gazette, vol. 85 (2001) 273–275.
22. MonishaPrabhu, "Key Dsitribution using Primitive Pythagorean triples", Master Thesis, Oklahoma State University (2013) pp. 1–51.
23. A. Mushtaque and S. Snehanshu, "Estimating the number of prime numbers less than a given positive integer by a novel quadrature method: A study of accuracy and convergence", ICACCI-IEEE, ISBN 978-1-4799-3078-4 (2014) pp. 415–421.
24. J.J. O'Conner and E.F. Robertson, "Baudhayana. History of Mathematics Project" (2000). http://www-history.mcs.st-and.ac.uk/~history/Biographies/Baudhayana.html
25. A. Parakh and S. Kak, "Online data storage using implicit security", Information Sciences, vol. 179 (2009), pp. 3323–3331.
26. A. Parakh and S. Kak, "Space efficient secret sharing for implicit data security", Information Sciences, vol. 181 (2011) pp. 335–341.
27. H.L. Price, "The Pythagorean Tree: A New Species" (2008) arXiv:0809.4324.
28. R.A. Saunders and T. Randall, T. (1994). "The family tree of the Pythagorean triplets revisited", Mathematical Gazette, vol. 78 (1994) pp. 190–193, JSTOR 3618576.
29. A. Seidenberg, "The origin of mathematics", Archive for History of Exact Sciences, vol. 18 (1978) pp. 301–42.
30. S. William, "Rethinking Pythagorean Triples" (2004) ISSN: 1932-9466.


## AUTHORS PROFILE

**V. Yegnanarayanan** obtained his Full Time PhD in Mathematics from Annamalai University in Nov 1996. He has 31 years of total experience in which 16 years as Professor & Dean/HOD. He has also worked as a Visiting Scientist on lien in TIFR and IMSC. He has authored 164 research papers and guided 6 students to Ph.D degree . He has delivered a number of invited talks, organized funded conferences, FDP etc, did a lot of review work for MR, zbMATH and reputed journals, completed research projects funded by NBHM-GOI, won Sentinel of Science Award from Publons. TN State (Periyar) University has recognized his qualifications and experience for the post of Principal in 2009.

**Poojitha Yakkala**, is doing her B.Tech in Computer Science and Engineering@ School of Computing, SASTRA Deemed to be University, Thanjavur, Tamilnadu. My current research interests are cyber security, ethical hacking and big data analytics.