```
<body>

<?php
print" <body style='background:#CCFFCC;'>";
print "<font face= 'courier'  size='4' >";
// define variables and set to empty values
$passwordErr = $bitsErr = $textErr = $transformErr="";
$pw = $nrOfBits = $text = $trans="";
/*
       AES Electronic Code Book Mode

       Programmed with reference to:

https://engineering.purdue.edu/kak/compsec/NewLectures/Lecture8.pdf
            http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-
38a.pdf
        I have also used some of the programming ideas from Chris
Veness' excellent
        AES section at http://www.movable-
type.co.uk/scripts/aes.html, though with
        changes and additions to enable inverse functions of
Rijndael, choice of
        keylengths, modes other than CTR, selection of nonce and to
suit my own style
        of programming.

       Program is controlled in functions encipher() & decipher().
       Rijndael algorithm is performed in function getKeySchedule()
       & in function Rijndael() and RijndaelInverse().

       This program has been validated using the AES-CTR data in
Sections F.1
       in the document at:
       http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-
38a.pdf

       No warranty of any form is offered for this program.

       Mike Cowan 26:05:2014
       */

 function RijndaelInverse($in, $word)
   {
    $state=array(4);
    for($r=0;$r<4;$r++) $state[$r]=array(4);
    $t=0;
    for($c=0;$c<4;$c++)
      for($r=0;$r<4;$r++)
         {$state[$r][$c] = $in[$t];$t++;}

     $nrOfRounds = (count($word)/4) - 1;


    $state = addRoundKey($state, $word, $nrOfRounds);
    $state = shiftRowsInverse($state);
    $state = subBytesInverse($state);

    for($round=$nrOfRounds-1;$round>0;$round--)
      {
       $state = addRoundKey($state, $word, $round);
       $state = mixColumnsInverse($state);
```

```
       $state = shiftRowsInverse($state);
       $state = subBytesInverse($state);
      }

     $state = addRoundKey($state, $word, 0);

      //...make 16-byte out by stripping state column by column...
    $out=array(16);
    $t=0;
    for($c=0;$c<4;$c++)
      for($r=0;$r<4;$r++)
         {$out[$t] = $state[$r][$c]; $t++;}

    return $out;
   }

    function subBytesInverse($state)
    {
    global $sBoxInverse;
    for ($r=0; $r<4; $r++)
       {
         for ($c=0; $c<4; $c++) $state[$r][$c] =
$sBoxInverse[$state[$r][$c]];
       }
    return $state;
  }

  function shiftRowsInverse($state)
   {
    $copy = array(4);
    for ($r=1; $r<4; $r++)
      {
       for ($c=0; $c<4; $c++)
            {
             $x=$c-$r;
             if($x<0) $x+=4;
             $copy[$c] = $state[$r][$x];
            }
       for ($c=0; $c<4; $c++) $state[$r][$c] = $copy[$c];
      }
    return $state;
  }

  function mixColumnsInverse($state)
    {
      global $table9,$table11,$table13,$table14;
      for ($c=0; $c<4; $c++)
           {
              $a = array(4);
              for($j=0;$j<4;$j++) $a[$j]=$state[$j][$c];

              $state[0][$c]
=$table14[$a[0]]^$table11[$a[1]]^$table13[$a[2]]^$table9[$a[3]];
              $state[1][$c]
=$table9[$a[0]]^$table14[$a[1]]^$table11[$a[2]]^$table13[$a[3]];
              $state[2][$c]
=$table13[$a[0]]^$table9[$a[1]]^$table14[$a[2]]^$table11[$a[3]];
              $state[3][$c]
=$table11[$a[0]]^$table13[$a[1]]^$table9[$a[2]]^$table14[$a[3]];
           }
      return $state;
```

```
    }

 $table9 = array(

0,9,18,27,36,45,54,63,72,65,90,83,108,101,126,119,144,153,130,139,180
,189,166,175,216,209,202,195,252,

245,238,231,59,50,41,32,31,22,13,4,115,122,97,104,87,94,69,76,171,162
,185,176,143,134,157,148,227,234,

241,248,199,206,213,220,118,127,100,109,82,91,64,73,62,55,44,37,26,19
,8,1,230,239,244,253,194,203,208,

217,174,167,188,181,138,131,152,145,77,68,95,86,105,96,123,114,5,12,2
3,30,33,40,51,58,221,212,207,198,

249,240,235,226,149,156,135,142,177,184,163,170,236,229,254,247,200,1
93,218,211,164,173,182,191,128,137,

146,155,124,117,110,103,88,81,74,67,52,61,38,47,16,25,2,11,215,222,19
7,204,243,250,225,232,159,150,141,

132,187,178,169,160,71,78,85,92,99,106,113,120,15,6,29,20,43,34,57,48
,154,147,136,129,190,183,172,165,

210,219,192,201,246,255,228,237,10,3,24,17,46,39,60,53,66,75,80,89,10
2,111,116,125,161,168,179,186,133,

140,151,158,233,224,251,242,205,196,223,214,49,56,35,42,21,28,7,14,12
1,112,107,98,93,84,79,70);

 $table11 = array(

0,11,22,29,44,39,58,49,88,83,78,69,116,127,98,105,176,187,166,173,156
,151,138,129,232,227,254,245,196,207,

210,217,123,112,109,102,87,92,65,74,35,40,53,62,15,4,25,18,203,192,22
1,214,231,236,241,250,147,152,133,142,

191,180,169,162,246,253,224,235,218,209,204,199,174,165,184,179,130,1
37,148,159,70,77,80,91,106,97,124,119,

30,21,8,3,50,57,36,47,141,134,155,144,161,170,183,188,213,222,195,200
,249,242,239,228,61,54,43,32,17,26,7,

12,101,110,115,120,73,66,95,84,247,252,225,234,219,208,205,198,175,16
4,185,178,131,136,149,158,71,76,81,90,

107,96,125,118,31,20,9,2,51,56,37,46,140,135,154,145,160,171,182,189,
212,223,194,201,248,243,238,229,60,55,

42,33,16,27,6,13,100,111,114,121,72,67,94,85,1,10,23,28,45,38,59,48,8
9,82,79,68,117,126,99,104,177,186,167,

172,157,150,139,128,233,226,255,244,197,206,211,216,122,113,108,103,8
6,93,64,75,34,41,52,63,14,5,24,19,202,
 193,220,215,230,237,240,251,146,153,132,143,190,181,168,163);

 $table13 = array(
```

```
0,13,26,23,52,57,46,35,104,101,114,127,92,81,70,75,208,221,202,199,22
8,233,254,243,184,181,162,175,140,129,

150,155,187,182,161,172,143,130,149,152,211,222,201,196,231,234,253,2
40,107,102,113,124,95,82,69,72,3,14,25,

20,55,58,45,32,109,96,119,122,89,84,67,78,5,8,31,18,49,60,43,38,189,1
76,167,170,137,132,147,158,213,216,207,

194,225,236,251,246,214,219,204,193,226,239,248,245,190,179,164,169,1
38,135,144,157,6,11,28,17,50,63,40,37,

110,99,116,121,90,87,64,77,218,215,192,205,238,227,244,249,178,191,16
8,165,134,139,156,145,10,7,16,29,62,51,

36,41,98,111,120,117,86,91,76,65,97,108,123,118,85,88,79,66,9,4,19,30
,61,48,39,42,177,188,171,166,133,136,159,

146,217,212,195,206,237,224,247,250,183,186,173,160,131,142,153,148,2
23,210,197,200,235,230,241,252,103,106,

125,112,83,94,73,68,15,2,21,24,59,54,33,44,12,1,22,27,56,53,34,47,100
,105,126,115,80,93,74,71,220,209,198,203,
 232,229,242,255,180,185,174,163,128,141,154,151);

 $table14 = array(

0,14,28,18,56,54,36,42,112,126,108,98,72,70,84,90,224,238,252,242,216
,214,196,202,144,158,140,130,168,166,180,

186,219,213,199,201,227,237,255,241,171,165,183,185,147,157,143,129,5
9,53,39,41,3,13,31,17,75,69,87,89,115,125,

111,97,173,163,177,191,149,155,137,135,221,211,193,207,229,235,249,24
7,77,67,81,95,117,123,105,103,61,51,33,47,

5,11,25,23,118,120,106,100,78,64,82,92,6,8,26,20,62,48,34,44,150,152,
138,132,174,160,178,188,230,232,250,244,222,

208,194,204,65,79,93,83,121,119,101,107,49,63,45,35,9,7,21,27,161,175
,189,179,153,151,133,139,209,223,205,195,

233,231,245,251,154,148,134,136,162,172,190,176,234,228,246,248,210,2
20,206,192,122,116,102,104,66,76,94,80,10,

4,22,24,50,60,46,32,236,226,240,254,212,218,200,198,156,146,128,142,1
64,170,184,182,12,2,16,30,52,58,40,38,124,

114,96,110,68,74,88,86,55,57,43,37,15,1,19,29,71,73,91,85,127,113,99,
109,215,217,203,197,239,225,243,253,167,169,
 187,181,159,145,131,141);

  $sBoxInverse = array(
    82,9,106,213,48,54,165,56,191,64,163,158,129,243,215,251,
124,227,57,130,155,47,255,135,52,142,67,68,196,222,233,203,
84,123,148,50,166,194,35,61,238,76,149,11,66,250,195,78,
8,46,161,102,40,217,36,178,118,91,162,73,109,139,209,37,
114,248,246,100,134,104,152,22,212,164,92,204,93,101,182,146,
108,112,72,80,253,237,185,218,94,21,70,87,167,141,157,132,
144,216,171,0,140,188,211,10,247,228,88,5,184,179,69,6,
```

```
208,44,30,143,202,63,15,2,193,175,189,3,1,19,138,107,
58,145,17,65,79,103,220,234,151,242,207,206,240,180,230,115,
150,172,116,34,231,173,53,133,226,249,55,232,28,117,223,110,
71,241,26,113,29,41,197,137,111,183,98,14,170,24,190,27,
252,86,62,75,198,210,121,32,154,219,192,254,120,205,90,244,
31,221,168,51,136,7,199,49,177,18,16,89,39,128,236,95,
96,81,127,169,25,181,74,13,45,229,122,159,147,201,156,239,
160,224,59,77,174,42,245,176,200,235,187,60,131,83,153,97,
23,43,4,126,186,119,214,38,225,105,20,99,85,33,12,125);

//.....................................................................
...

function Rijndael($in, $word)
  {
    /*
     see
https://engineering.purdue.edu/kak/compsec/NewLectures/Lecture8.pdf
     and http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
pages 14 & 15.

     input of 16 bytes is put into a 4x4 'state array', column by
column.

     word is the Key Schedule, made earlier, comprising 4-byte words.
There are
     44/52/60 words, dependant on key length.
    */


    $state=array(4);
    for($r=0;$r<4;$r++) $state[$r]=array(4);
    $t=0;
    for($c=0;$c<4;$c++)
      for($r=0;$r<4;$r++)
         {$state[$r][$c] = $in[$t];$t++;}


    $nrOfRounds = (count($word)/4) - 1;

    $state = addRoundKey($state, $word, 0);

    for ($round=1; $round<$nrOfRounds; $round++)
       {
        $state = subBytes($state);
        $state = shiftRows($state);
        $state = mixColumns($state);
        $state = addRoundKey($state, $word, $round);
       }

    $state = subBytes($state);
    $state = shiftRows($state);
    $state = addRoundKey($state, $word, $nrOfRounds);

  //...make 16-byte out by stripping state column by column...
    $out=array(16);
    $t=0;
    for($c=0;$c<4;$c++)
      for($r=0;$r<4;$r++)
         {$out[$t] = $state[$r][$c]; $t++;}
```

```
    return $out;
  }
 //.................................................


  function subBytes($state)
    {
    global $sBox;
    for ($r=0; $r<4; $r++)
       {
         for ($c=0; $c<4; $c++) $state[$r][$c] =
$sBox[$state[$r][$c]];
       }
    return $state;
  }

function shiftRows($state)
   {
    $copy = array(4); //temporary usage
    for ($r=1; $r<4; $r++)
      {
       for ($c=0; $c<4; $c++) $copy[$c] = $state[$r][($c+$r)%4];
       for ($c=0; $c<4; $c++) $state[$r][$c] = $copy[$c];
      }
    return $state;
  }

function mixColumns($state)
   {
      //http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
page 17
      for ($c=0; $c<4; $c++)
        {
         $a = array(4);  // a[n] is a copy of the current input
column
         $b = array(4);  // b[n] is going to be 2 * a[n] in the
Galois Field
                        //  a[n]^b[n]   will be 3 * a[n]

            for($j=0;$j<4;$j++)
               {
                  $a[$j] = $state[$j][$c];
                  $b[$j] =  $state[$j][$c]<<1;   //left shift doubles
the value of $s[][]
                  if($b[$j]>255) {$b[$j]-=256;$b[$j]^=27;} // but if
result >255 then must
                                                           //
subtract 256 & xor with 27
               }

         $state[0][$c] = $b[0] ^ $a[1] ^ $b[1] ^ $a[2] ^ $a[3]; //
2a0 + 3a1 +  a2 +  a3;
         $state[1][$c] = $a[0] ^ $b[1] ^ $a[2] ^ $b[2] ^ $a[3]; //
a0 * 2a1 + 3a2 +  a3
         $state[2][$c] = $a[0] ^ $a[1] ^ $b[2] ^ $a[3] ^ $b[3]; //
a0 +  a1 + 2a2 + 3a3
         $state[3][$c] = $a[0] ^ $b[0] ^ $a[1] ^ $a[2] ^ $b[3]; //
3a0 +  a1 +  a2 + 2a3
        }
      return $state;
   }
```

```
function addRoundKey($state, $word, $roundNumber)
    {
      for ($r=0; $r<4; $r++)
       {
         for ($c=0; $c<4; $c++)
              {
                $nr= (4*$roundNumber) + $c; // word number
                $state[$r][$c] ^= $word[$nr][$r];
              }
       }
    return $state;
  }


 function rotate($word)
 {
    $copy = $word[0];
    for ($j=0; $j<3; $j++) $word[$j] = $word[$j+1];
    $word[3] = $copy;
    return $word;
  }



    $rCon = array(
    array(0,0,0,0),
    array(1,0,0,0),
    array(2,0,0,0),
    array(4,0,0,0),
    array(8,0,0,0),
    array(16,0,0,0),
    array(32,0,0,0),
    array(64,0,0,0),
    array(128,0,0,0),
    array(27,0,0,0),
    array(54,0,0,0) );

    $sBox = array(

99,124,119,123,242,107,111,197,48,1,103,43,254,215,171,118,202,130,20
1,125,250,89,71,240,173,212,162,175,

156,164,114,192,183,253,147,38,54,63,247,204,52,165,229,241,113,216,4
9,21,4,199,35,195,24,150,5,154,7,18,

128,226,235,39,178,117,9,131,44,26,27,110,90,160,82,59,214,179,41,227
,47,132,83,209,0,237,32,252,177,91,

106,203,190,57,74,76,88,207,208,239,170,251,67,77,51,133,69,249,2,127
,80,60,159,168,81,163,64,143,146,157,

56,245,188,182,218,33,16,255,243,210,205,12,19,236,95,151,68,23,196,1
67,126,61,100,93,25,115,96,129,79,220,

34,42,144,136,70,238,184,20,222,94,11,219,224,50,58,10,73,6,36,92,194
,211,172,98,145,149,228,121,231,200,

55,109,141,213,78,169,108,86,244,234,101,122,174,8,186,120,37,46,28,1
66,180,198,232,221,116,31,75,189,139,
```

```
138,112,62,181,102,72,3,246,14,97,53,87,185,134,193,29,158,225,248,15
2,17,105,217,142,148,155,30,135,233,

206,85,40,223,140,161,137,13,191,230,66,104,65,153,45,15,176,84,187,2
2);


//.............................................................
......


function getKeySchedule($key)
  {
    /* see
https://engineering.purdue.edu/kak/compsec/NewLectures/Lecture8.pdf
       pages 7 and 8.
       The key is loaded into words[0->3] of 4 bytes each.
       There will be 10/12/14 rounds for 128/192/256 bit keys, which
means there
        will be 44, 52, 60 words in total.
       In each round:
       -the first word is made from the previous word processed
through function 'g'
         (see below) and then xored with the first word of the
previous round;
       -the next three words are made from the previous word xored
with the word
         in the same place from the previous round.

       The function 'g' submits copy to functions rotate, subBytes
and rCon.
      */

    global $sBox,$rCon;

    $nrOfWords = count($key)/4;       // number of 4-byte words in
key =4/6/8
    $nrOfRounds = $nrOfWords + 6;     // number of rounds =10/12/14

    $word=array(70);
    for($i=0;$i<4;$i++) $word[$i]=array(4);

    $t=0;
    for($i=0;$i<$nrOfWords;$i++)
      for($k=0;$k<4;$k++)
         {$word[$i][$k] = $key[$t];$t++;}

      //....for 128 bit key make four 4-byte words for 10 rounds.....
      // ..........numbered $w[4] to $w[43].................
       //.. up to word[51] or word[59] for 192 or 256 bit keys
resp.....
   for ($j=$nrOfWords; $j<(4*($nrOfRounds+1)); $j++)
      {
       //...make a copy of the previous four words.......
       for ($t=0; $t<4; $t++) $copy[$t] = $word[$j-1][$t];

       //...if we are dealing with the first word of a group of four
      //..... then make function g from copy.................
       if ($j % $nrOfWords == 0)
         {
```

```
          //...do rotate......
          $copy = rotate($copy);
          //.....do subBytes........
         for ($t=0; $t<4; $t++) $copy[$t] = $sBox[$copy[$t]];
          //.....do rCon..................
          for ($t=0; $t<4; $t++) $copy[$t] ^=
$rCon[$j/$nrOfWords][$t];
         }
             //...extra routine if key is 256 bits....
        else if($nrOfWords==8 && ($j-4)%8==0 )
            {
             for ($t=0; $t<4; $t++) $copy[$t] = $sBox[$copy[$t]];
            }

      //...finally  xor copy with the word from same position in last
round....
      for ($t=0; $t<4; $t++) $word[$j][$t] = $word[$j-$nrOfWords][$t]
^ $copy[$t];
     }

     return $word;

 }


//.............................................................

 function encipher($key,$ptDigits,$nrOfBits)
    {
      $nrOfBytes = $nrOfBits/8;  // no bytes in key

      $keySchedule = getKeySchedule($key);

 //....pad the pt to be exact number of blocks......
      $lengthPt = count($ptDigits);
      $requiredLength=  ceil($lengthPt/16) *16;
      $pad= $requiredLength - $lengthPt;
      for($j=0;$j<$pad;$j++) $ptDigits[$lengthPt+$j] = 65;

   //..put amount of padding as initial cipher character.....
      $ciphertext="";
      $ciphertext=$ciphertext.chr($pad);

      for ($j=0; $j<$requiredLength; $j+=16)
        {
         for($k=0;$k<16;$k++) $ptBlock[$k]=$ptDigits[$j+$k];
         $encryption =  Rijndael($ptBlock, $keySchedule);
         for($k=0;$k<16;$k++) $ciphertext.=chr($encryption[$k]);
        }

    $ct = base64_encode($ciphertext);
    return $ct;
  }

//.............................................................
  function decipher($ciphertext, $key, $nrOfBits)
  {
      //..note: ciphertext has been decoded earlier from base64......
      $nrOfBytes = $nrOfBits/8;  // no bytes in key

      $keySchedule = getKeySchedule($key);
```

```
      $pad=ord(substr($ciphertext,0,1));
      $plainex=array(); $t=0;
      for ($j=1; $j<strlen($ciphertext); $j+=16)
        {
         for($k=0;$k<16;$k++)
$ctBlock[$k]=ord(substr($ciphertext,($j+$k),1));
         $decryptedBlock =  RijndaelInverse($ctBlock, $keySchedule);

         for($k=0;$k<16;$k++)
            {
             $plainex[$t]=$decryptedBlock[$k];
             $t++;
            }
        }
      //....drop the padding.......
      $redex=array();
      for($j=0;$j<$t-$pad;$j++)
         $redex[$j]=$plainex[$j];
      return $redex;
  }



//---------------------------------------------------------------
//.....MAIN........


   if ($_SERVER["REQUEST_METHOD"] == "POST")
  {
     if (empty($_POST["password"]))
         {
          $passwordErr = "Password is required";
         }
     else {
           $pw = test_input($_POST["password"]);
          }


     if (empty($_POST["bits"]))
         {
          $bitsErr = "bits is required";
         }
     else {
          $nrOfBits = test_input($_POST["bits"]);
         }


     if (empty($_POST["text"]))
        {
         $textErr = "text is required";
        }
    else
       {
        $txt = test_input($_POST["text"]);
       }

    if (empty($_POST["transform"]))
         {
          $transformErr = "transform is required";
         }
```

```
     else {
          $trans = test_input($_POST["transform"]);
         }

  }

function test_input($data)
   {
     $data = trim($data);
     $data = stripslashes($data);
     $data = htmlspecialchars($data,ENT_IGNORE);
     return $data;
   }

 function digitsFromText($txt)
   {
     $digits=array(); $t=0;
     for($j=0;$j<strlen($txt);$j++)
       if(substr($txt,$j,1)=='&')
        {
         if(substr($txt,$j,5)==="&amp;")
{$digits[$t]=38;$t++;$jump=4; }
         if(substr($txt,$j,4)=="&lt;")   {$digits[$t]=60; $t++;
$digits[$t]=32;$t++;$jump=3; }
         if(substr($txt,$j,4)=="&gt;")
{$digits[$t]=62;$t++;$jump=3; }
         if(substr($txt,$j,6)=="&quot;")
{$digits[$t]=34;$t++;$jump=5;}
         if(substr($txt,$j,6)=="&#039;")  {$tt[$t]=39;$t++;$jump=5;}
         $j=$j+$jump;
        }
      else {$digits[$t]= ord(substr($txt,$j,1));$t++;}
      return $digits;
   }

 function extendKey($key,$nrOfBits)
    {

       //...ensure password digits to required number of bytes..
      $len_password=count($key);
      $nrOfBytes= $nrOfBits/8;
      $diff=$nrOfBytes-$len_password;
      for($j=0;$j<$diff;$j++)
            {
             $x=$j%$len_password;
             $key[$len_password+$j]=$key[$j];
            }
       return $key;
    }

?>

<h2>AES ECB MODE</h2>
<p><span class="error">* required field.</span></p>
<form method="post" action="<?php echo
htmlspecialchars($_SERVER["PHP_SELF"]);?>">

   password:
   <input type="text" name="password" value="<?php echo $pw;?>">
   <span class="error">* <?php echo $passwordErr;?></span>
   <br><br>
```

```
    Number of Bits:
   <input type="radio" name="bits" <?php if (isset($nrOfBits) &&
$nrOfBits=="128") echo "checked";?>  value="128">128
   <input type="radio" name="bits" <?php if (isset($nrOfBits) &&
$nrOfBits=="192") echo "checked";?>  value="192">192
   <input type="radio" name="bits" <?php if (isset($nrOfBits) &&
$nrOfBits=="256") echo "checked";?>  value="256">256
   <span class="error">* <?php echo $bitsErr;?></span>
   <br><br>

   input: <br><textarea name="text" rows="10" cols="80"><?php echo
$txt;?></textarea>
   <span class="error">* <?php echo $textErr;?></span>
   <br><br>

   transform:
   <input type="radio" name="transform" <?php if (isset($trans) &&
$trans=="encipher") echo "checked";?>  value="encipher">encipher
   <input type="radio" name="transform" <?php if (isset($trans) &&
$trans=="decipher") echo "checked";?>  value="decipher">decipher
   <span class="error">* <?php echo $transformErr;?></span>
   <br><br>


   <input type="submit" name="submit" value="Submit">
</form>

<?php

   if(strlen($pw)==0 || strlen($txt)==0) $bad=1;
   else $bad=0;

        if($trans=='encipher' && $bad==0)
           {
            $key = array();
            $key=digitsFromText($pw);
            $key =extendKey($key,$nrOfBits);
            $ptDigits=digitsFromText($txt);
            $ct=encipher($key,$ptDigits,$nrOfBits);
            $out=$ct;
         }

  if($trans=='decipher' && $bad==0)
      {
        //...check that ciphertext has only permitted characters...
        $good = "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789+/=";
        for($j=0;$j<strlen($txt);$j++)
           {
            $ch=substr($txt,$j,1);
            $ch=strtoupper($ch);
            $bad=1;
            for($k=0;$k<39;$k++)
                if($ch==substr($good,$k,1)) $bad=0;
             if(ord($ch) ==10 || ord($ch)==13) $bad=0;
            if($bad==1)
               {
                 print "<font color = 'red' font face= 'arial'
size='5' >";
                 print "<br> Bad character found in Ciphertext, not
present in base64 encoding <br>";
```

```
                print" which uses only letters of alphabet, numbers
0 to 9 and + or / <br>";
                print"ciphertext may end in = or == <br>";
                print " I am aborting this run <br>";
                break;
             }
          }
        //...convert from base64......
        $ciphertext = base64_decode($txt);
        $x=strlen($ciphertext)%16;
        if($x!=1)
             {
              $bad=1;
              print "<font color = 'red' font face= 'arial'  size='5'
Error in  Ciphertext <br>";
              print "length of decoded Base64 text is incorrect
<br>";
              print " I am aborting this run <br>";
             }

        if($bad==0)
            {

              $key = array();
              $key=digitsFromText($pw);
              $key =extendKey($key,$nrOfBits);
              $plainex = decipher($ciphertext, $key, $nrOfBits);
              $out="";
              for($j=0;$j<count($plainex);$j++)
                 if($plainex[$j]==10) $out.="\n";
                 else $out.=chr($plainex[$j]);


            }
    }


print "<font color = 'black' font face= 'arial'  size='4' Error in
Ciphertext <br>";
?>

output:
 <br><textarea name="text" rows="10" cols="80"><?php echo
$out;?></textarea>
   <br><br>
 <h5>
[validated with NIST input & output data, see:<br>
http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf]<br><br>
M J Cowan 2014    </h5>

</body>
</html>
```