```
>
```

Today's goal: implement RSA from input file to list of numbers, & decrypt.

1) get your text somehow.
2) set up RSA: (secret part) pick 2 big primes p, q let n=p*q
and and exponent e rel prime to phi(n)
(secret) determine d=1/e mod phi(n)
publish (n, e). [encrypting key]
remember (n, d) [decrypting key]

```
> text:= "now is the time and this is the record of the time.";
```

text := "now is the time and this is the record of the time." **(1)**

Use as our alphabet just anything a computer can represent, ie bytes 0 .. 255.

```
> rand();
```

395718860534 **(2)**

```
> dice:=rand(1..6);
```

dice := **proc**( ) **(3)**

proc( ) **option** *builtin* = *RandNumberInterface*; **end proc**(6, 6, 3) + 1

end proc

```
> dice();
```

6 **(4)**

```
> randomize();
```

1364410584 **(5)**

```
> SetUpRSA:=proc(ndigits::posint)
    local rnum, Randy;
    randomize();
    Randy:=rand(10^(ndigits-1)..10^(ndigits));
    print(Randy());
  end:


> SetUpRSA(3);
```

623 **(6)**

```
> SetUpRSA:=proc(ndigits::posint)
    local rnum, Randy, Mandy,p, q, n, e, phi, d;
    randomize();
    Randy:=rand(10^(ndigits-1)..10^(ndigits));
    p:=nextprime(Randy());
    q:=nextprime(Randy());
    phi:=(p-1)*(q-1);
    n:=p*q;
    Mandy:=rand(3..p);
    e:=Mandy();
    while ( gcd(e,phi) <> 1) do
      print(e,"gcd of e, phi is", gcd(e,phi));
      e:=Mandy();
    od;

    #  need gcd(e,phi)=1
    print(p,q, n, phi, e);
  end:
```

```
> SetUpRSA(10);
```

1313581029, "gcd of e, phi is", 3
5229974463, "gcd of e, phi is", 21
7704763107, "gcd of e, phi is", 3
5388129080, "gcd of e, phi is", 8
163150998, "gcd of e, phi is", 6
8628415867, 9682861337, 83547754398131634179, 83547754379820356976, 21131975 **(7)**

```
> SetUpRSA:=proc(ndigits::posint)
    local rnum, Randy, Mandy,p, q, n, e, phi, d;
    randomize();
    Randy:=rand(10^(ndigits-1)..10^(ndigits));
    p:=nextprime(Randy());
    q:=nextprime(Randy());
    phi:=(p-1)*(q-1);
    n:=p*q;
    Mandy:=rand(3..p);
    e:=Mandy();
    while ( gcd(e,phi) <> 1) do #  need gcd(e,phi)=1
      ## print(e,"gcd of e, phi is", gcd(e,phi));
      e:=Mandy();
    od;
    d:= 1/e mod phi;
    printf("public key is (%d,%d), private is (%d,%d)",
          n,e, n, d);
    return ( [n,e], [n,d]);
  end:
> pub,priv:=SetUpRSA(4);
public key is (91644551,4799), private is (91644551,75625799)
```

pub, *priv* := [91644551, 4799], [91644551, 75625799] **(8)**

Still need something to convert my plaintext into BIG numbers,
Something to do the exponentiation stuff.

```
> n:=pub[1]; e:=pub[2]; d:=priv[2];
```

$n := 91644551$
$e := 4799$
$d := 75625799$ **(9)**

```
> text;
```

"now is the time and this is the record of the time." **(10)**

```
> with(StringTools):
> Explode(text);
```

["n", "o", "w", " ", "i", "s", " ", "t", "h", "e", " ", "t", "i", "m", "e", " ", "a", "n", "d", " ", "t", "h", "i", "s", " ", "i", "s", " ", "t", "h", "e", " ", "r", "e", "c", "o", "r", "d", " ", "o", "f", " ", "t", "h", "e", " ", "t", "i", "m", "e", "."] **(11)**

```
> map(Ord,%);
```

[110, 111, 119, 32, 105, 115, 32, 116, 104, 101, 32, 116, 105, 109, 101, 32, 97, 110, 100, 32, 116, 104, 105, 115, 32, 105, 115, 32, 116, 104, 101, 32, 114, 101, 99, 111, 114, 100, 32, 111, 102, 32, 116, 104, 101, 32, 116, 105, 109, 101, 46] **(12)**

```
> tryit:=map(Ord,Explode(text));
```

tryit := [110, 111, 119, 32, 105, 115, 32, 116, 104, 101, 32, 116, 105, 109, 101, 32, 97, 110, **(13)**

```
    100, 32, 116, 104, 105, 115, 32, 105, 115, 32, 116, 104, 101, 32, 114, 101, 99, 111, 114,
    100, 32, 111, 102, 32, 116, 104, 101, 32, 116, 105, 109, 101, 46]
> convert(tryit, base, 256, n);
[38996666, 84723777, 20818592, 59814376, 73632811, 88902933, 81165642, 60154584,                (14)
    50255446, 63023882, 53175057, 29274220, 87625992, 68224596, 44637795, 443]
> convert(%,base,n,256);
[110, 111, 119, 32, 105, 115, 32, 116, 104, 101, 32, 116, 105, 109, 101, 32, 97, 110, 100, 32,  (15)
    116, 104, 105, 115, 32, 105, 115, 32, 116, 104, 101, 32, 114, 101, 99, 111, 114, 100, 32,
    111, 102, 32, 116, 104, 101, 32, 116, 105, 109, 101, 46]
> StringToBaseN:=proc(text::string,n::posint)
    convert(
      map(Ord,Explode(text)),
      base, 256, n);
  end:
  BaseNToStr:=proc(baseN::list(posint), n::posint)
     cat(op(map(Char,
           convert(baseN,base,n,256))));
  end:
> StringToBaseN("stuff stuff stuffy stuiff",456);
[67, 118, 80, 55, 19, 425, 435, 46, 303, 140, 137, 2, 332, 23, 57, 58, 286, 384, 3, 8, 89, 208, 20]  (16)
> BaseNToStr(%,456);
                                "stuff stuff stuffy stuiff"                                      (17)
> ApplyRSA:=proc(baseN::list(posint), key::list(posint))
    local n, e;
    n:=key[1]; e:=key[2];
    map( x-> modp(x&^e, n), baseN);
  end:
> ApplyRSA( StringToBaseN("yippity zippity", n), [n,e]);
                  [20563300, 84620337, 75919031, 14611267, 54026771]                             (18)
> ApplyRSA( %, [n,d]);
                     [2178771, 79006203, 51529508, 18584495, 8940]                               (19)
> BaseNToStr(%,n);
                                  "yippity zippity"                                              (20)
> DoRSA := proc( text, key)
    ApplyRSA( StringToBaseN(text, key[1]), key)
    end:
  UndoRSA:=proc (numlist, decr)
    BaseNToStr( ApplyRSA ( numlist, decr), decr[1]);
  end:
> DoRSA(text, pub);
[66133642, 77891432, 52233131, 54491971, 16013079, 35949331, 17262914, 6490804,                 (21)
    8237599, 37631224, 58913759, 61938827, 72850462, 18763363, 68891515, 46743878]
> UndoRSA(%, priv);
                  "now is the time and this is the record of the time."                          (22)
```