

```

> with(StringTools):
> Alphabet := Select(IsPrintable, convert([seq(i,i=1..127)], bytes)
);
Alphabet:=
"!#$%&'()*+,-./0123456789:<=>?@ABCDEFGHIJKLMNPOQRSTUVWXYZ[\]
^_`abcdefghijklmnopqrstuvwxy{~"
(1)
=
> StringToList := proc (str::string)
global Alphabet;
return([seq(SearchText(str[i], Alphabet)-1, i = 1 .. length
(str))]);
end:
ListToString := proc (l::list(nonnegint))
global Alphabet;
return(cat(seq(Alphabet[l[i]+1], i = 1 .. nops(l))))
end:
> StringToList("Some string.");
[51, 79, 77, 69, 0, 83, 84, 82, 73, 78, 71, 14]
(2)
=
> ListToString(%);
"Some string."
(3)
=
> ListToString([-1, 5]);
Error, invalid input: ListToString expects its 1st argument, l,
to be of type list(nonnegint), but received [-1, 5]
> Spot := "Sally said, "See Spot run. Run, Spot, run!"";
Error, recursive assignment
> Spot := "Sally said, \"See Spot run. Run, Spot, run!\"";
Spot:= "Sally said, "See Spot run. Run, Spot, run!""
(4)
=
> length(Spot);
43
(5)
=
> Spot[42..43];
"!""
(6)
=
> Prufrock := " I have heard the mermaids singing, each to each.
I do not think they will sing to me.";
Prufrock:= " I have heard the mermaids singing, each to each.
I do not think they will sing to me."
(7)
=
> Newline := "\n";
Newline:= "
"
(8)
=
> x:=123\
456;
x:= 123456
(9)
=
> Caesar:=proc(plain::string, shift::integer)

```

```

local L, S, len;
global Alphabet;
len:=length(Alphabet);
L:=StringToList(plain); # convert to numbers
## S:=[seq( (L[i]+shift) mod len, i=1..nops(L))];
S:=map(n->(n+shift) mod len, L); # same as above.
return(ListToString(S));
end:

```

```

> Caesar("Salad.", 5);
                                     "Xfqfi3"

```

(10)

```

> Caesar(%, -5);
                                     "Salad."

```

(11)

```

> L := [0,  $\frac{\text{Pi}}{4}$ ,  $\frac{\text{Pi}}{3}$ ,  $\frac{\text{Pi}}{2}$ ];
                                     L := [0,  $\frac{1}{4}\pi$ ,  $\frac{1}{3}\pi$ ,  $\frac{1}{2}\pi$ ]

```

(12)

```

> map(sin, L);
                                     [0,  $\frac{1}{2}\sqrt{2}$ ,  $\frac{1}{2}\sqrt{3}$ , 1]

```

(13)

```

> StringToList("Salad");
                                     [51, 65, 76, 65, 68]

```

(14)

```

> StringToList("cow");
                                     [67, 79, 87]

```

(15)

```

> Vignere:=proc(plain::string, key::string)
local L, shifts, S, len;
global Alphabet;
len:=length(Alphabet);
shifts:=StringToList(key); #list of shifts.
L:=StringToList(plain); # convert to numbers
S:=[seq( (L[i]+shifts[i]) mod len, i=1..nops(L))];
return(ListToString(S));
end:

```

This will break, kinda.

```

> Vignere("Cow", "Dogs and Cats");
                                     "g__"

```

(16)

```

> Vignere("Dogs and Cats", "cow");
Error. (in Vignere) invalid subscript selector

```

```

> Vignere:=proc(plain::string, key::string)
local L, shifts, S, len, keylen;
global Alphabet;
len:=length(Alphabet);
keylen:=length(key);
shifts:=StringToList(key); #list of shifts.

```

```

L:=StringToList(plain); # convert to numbers
S:=[seq( (L[i]+shifts[((i-1) mod keylen)+1]) mod len, i=1..nops
(L))];
return(ListToString(S));
end:

```

```

> mess := Vignere("Dogs and Cats", "cow");
      mess := ("_WoYRTw'QIW"
(17)

```

```

> deVignere:=proc(plain::string, key::string)
local L, shifts, S, len, keylen;
global Alphabet;
len:=length(Alphabet);
keylen:=length(key);
shifts:=StringToList(key); #list of shifts.
L:=StringToList(plain); # convert to numbers
S:=[seq( (L[i]-shifts[((i-1) mod keylen)+1]) mod len, i=1..nops
(L))];
return(ListToString(S));
end:

```

```

> deVignere(mess, "cow");
      "Dogs and Cats"
(18)

```

```

> StringToList("cow");
      [67, 79, 87]
(19)

```

```

> map(x→ -x mod length(Alphabet), %);
      [28, 16, 8]
(20)

```

```

> ListToString(%);
      "! 0("
(21)

```

```

> Vignere(mess, "<0(");
      "Dogs and Cats"
(22)

```

```

> Vignere(" I have heard the mermaids singing",
      "Dare I to eat the peach? I don't remember this stupid poem");
      "DC rNa@etXeGTY iQK ^KTQJ)d= XY]n^nZ"
(23)

```

```

> rand( );
      395718860534
(24)

```

```

> rand( );
      193139816415
(25)

```

```

> rand( );
      22424170465
(26)

```

```

>
> randomize(3);
      3
(27)

```

```

> rand( ); rand( ); rand( );
      455570294424
      790795084744

```

```
|
|
|> rand( ); rand( ); rand( );
|
|
|> randomize(3); rand( ); rand( ); rand( );
|
|
|>
|
```

2445173525 (28)

867663915883

593770158997

511502908838 (29)

3

455570294424

790795084744

2445173525 (30)