```
>
```

Initial setup and utility functions.

```
> with(StringTools):
```

PrintRuler is just to make a little ruler under the Alphabet so we can easily see what character has which position.

```
> PrintRuler:=proc(n)
    local j;
    for j from 0 to n-1 do
     if  (j mod 10 = 0) then printf("%d",trunc( modp(j,100)/10));

     elif (j mod 10 = 5) then printf("+");
     else printf(".");
      fi;
    od;
  end:
```

Let's define our Alphabet by selecting all printable characters from the ASCII sequence.

```
> Alphabet := Select(IsPrintable, convert([seq(i,i=1..127)],
  bytes)):
  printf("Our %d-character Alphabet is \n%s\n",length(Alphabet),
  Alphabet); PrintRuler(length(Alphabet));
Our 95-character Alphabet is
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]
^_`abcdefghijklmnopqrstuvwxyz{|}~
0....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+
```

StringToList converts a string into a list of numbers representing the position of each character in the Alphabet.

ListToString converts such a list back into a text string.

```
> StringToList := proc (str::string)
    local L,M;
    global Alphabet;
    L:=[seq(SearchText(str[i], Alphabet)-1, i = 1 .. length(str))
  ];
    return(L);
  end:
  ListToString := proc (l::list(nonnegint))
    global Alphabet;
    return(cat(seq(Alphabet[l[i]+1], i = 1 .. nops(l))))
  end:
```

We can also define versions of these which work on k-graphs, that is, multi-byte "characters".
StringToKgraph converts a string into a list of numbers representing the position of each character in the Alphabet taken k at a time.

KgraphToString converts such a list back into a text string.

Note that StringToKgraph(text,1) is the same as StringToList(text).

```
> StringToKgraph:=proc(text::string, k::posint)
    local numlist, p;
    global Alphabet;
    p:=length(Alphabet);
    numlist:=StringToList(text);
    return(convert(numlist,base, p, p^k));
  end:

  KgraphToString:=proc(numlist::list(nonnegint), k::posint)
    local  p;
    global Alphabet;

    p:=length(Alphabet);
    ListToString(convert(numlist,base, p^k, p));
  end:
```

```
> text := "Who put the bop in bop-bop-shu-bop?";
```

$$text := \text{"Who put the bop in bop-bop-shu-bop?"} \tag{1}$$

```
> StringToList(text);
```

$$[55, 72, 79, 0, 80, 85, 84, 0, 84, 72, 69, 0, 66, 79, 80, 0, 73, 78, 0, 66, 79, 80, 13, 66, 79, 80, 13, 83, 72, 85, 13, 66, 79, 80, 31] \tag{2}$$

```
> StringToKgraph(text, 4);
```

$$[719870, 766255, 629649, 729571, 56594233, 56711754, 71287129, 56712222, 287454] \tag{3}$$

```
>
```

About vectors and matrices

```
> v := <1, 2, 3>;
```

$$v := \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \tag{4}$$

```
> w := Vector([1, 0]);
```

$$w := \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{5}$$

```
> v:=<1,2,3>;
```

$$v := \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \tag{6}$$

```
> A:=Matrix( [ [1,2,3], [4,5,6], [7, 8,9]]);
```

$$A := \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \tag{7}$$

```
> B := <<1, 2>|<3, 4>>;
```

$$B := \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix} \tag{8}$$

```
> A.v;
```

(9)

$$\begin{bmatrix} 14 \\ 32 \\ 50 \end{bmatrix} \tag{9}$$

```
> A. A;
```

$$\begin{bmatrix} 30 & 36 & 42 \\ 66 & 81 & 96 \\ 102 & 126 & 150 \end{bmatrix} \tag{10}$$

The scalar product of v with itself.

```
> v.v;
```

$$14 \tag{11}$$

```
> A*v;
Error, (in rtable/Product) invalid arguments
> MatrixInverse(A);
```

$$\mathit{MatrixInverse}\left(\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}\right) \tag{12}$$

```
> with(LinearAlgebra) :
> MatrixInverse(A);
Error, (in MatrixInverse) singular matrix
> MatrixInverse(B);
```

$$\begin{bmatrix} -2 & \frac{3}{2} \\ 1 & -\frac{1}{2} \end{bmatrix} \tag{13}$$

```
> C := Matrix( [[1, 2, 3], [4, 5, 6], [0, 0, 1]]);
```

$$C := \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 0 & 0 & 1 \end{bmatrix} \tag{14}$$

```
> Matrix([[a, b, c]]);
```

$$\begin{bmatrix} a & b & c \end{bmatrix} \tag{15}$$

```
> Transpose(⟨1, 2, 3⟩);
```

$$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \tag{16}$$

```
> JordanForm(C);
```

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 3-2\sqrt{3} & 0 \\ 0 & 0 & 3+2\sqrt{3} \end{bmatrix} \tag{17}$$

```
> JordanForm(A);
```

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{15}{2} - \frac{3}{2}\sqrt{33} & 0 \\ 0 & 0 & \frac{15}{2} + \frac{3}{2}\sqrt{33} \end{bmatrix} \tag{18}$$

> *Eigenvalues*(*A*);

$$\begin{bmatrix} 0 \\ \frac{15}{2} + \frac{3}{2}\sqrt{33} \\ \frac{15}{2} - \frac{3}{2}\sqrt{33} \end{bmatrix} \tag{19}$$

> *CharacteristicPolynomial*(*A*, *x*);

$$x^3 - 15\,x^2 - 18\,x \tag{20}$$

> *StringToList*(*text*);

$$[55, 72, 79, 0, 80, 85, 84, 0, 84, 72, 69, 0, 66, 79, 80, 0, 73, 78, 0, 66, 79, 80, 13, 66, 79, 80, 13, 83, 72, 85, 13, 66, 79, 80, 31] \tag{21}$$

> *M* := *Matrix*([[1, 2], [3, 4]]);

$$M := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \tag{22}$$

> *M*.⟨55, 72⟩;

$$\begin{bmatrix} 199 \\ 453 \end{bmatrix} \tag{23}$$

> % **mod** 95;

$$\begin{bmatrix} 9 \\ 73 \end{bmatrix} \tag{24}$$

> *ListToString*([9, 73]);

")i" **(25)**

> *M*.⟨79, 0⟩ **mod** 95;

$$\begin{bmatrix} 79 \\ 47 \end{bmatrix} \tag{26}$$

> *ListToString*([79, 47]);

"oO" **(27)**

> *M*;

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \tag{28}$$

> *Dimensions*(*M*);

$$2, 2 \tag{29}$$

```
>
```

Need a StringToVects   and VectsToString

```
> l := [a, b, c, d, e, f];
```

$$l := [a, b, c, d, e, f] \qquad \textbf{(30)}$$

```
> k := 1;
  for i from 1 to nops(l)/2 do
     for j from 1 to 2 do
      print(k, l[k], "goes to", i, j);
     k := k + 1;
    od;
   od;
```

$$k := 1$$

$$1, a, \text{"goes to"}, 1, 1$$

$$2, b, \text{"goes to"}, 1, 2$$

$$3, c, \text{"goes to"}, 2, 1$$

$$4, d, \text{"goes to"}, 2, 2$$

$$5, e, \text{"goes to"}, 3, 1$$

$$6, f, \text{"goes to"}, 3, 2 \qquad \textbf{(31)}$$

```
> k := 1; VL := [ ]; n := 2;
  for i from 1 to nops(l)/2 do
    v := ⟨seq(l[j], j = k ... k + 1)⟩;
    print(v);
    k := k + 2;
   od;
```

$$k := 1$$

$$VL := [\ ]$$

$$n := 2$$

$$v := \begin{bmatrix} a \\ b \end{bmatrix}$$

$$\begin{bmatrix} a \\ b \end{bmatrix}$$

$$k := 3$$

$$v := \begin{bmatrix} c \\ d \end{bmatrix}$$

$$\begin{bmatrix} c \\ d \end{bmatrix}$$

$$k := 5$$

$$v := \begin{bmatrix} e \\ f \end{bmatrix}$$

$$\begin{bmatrix} e \\ f \end{bmatrix}$$

$$k := 7 \tag{32}$$

> $k := 1;\ VL := [\ ];\ n := 2;$
>
> **for** i **from** 1 **to** $\dfrac{nops(l)}{2}$ **do**
>
> $v := \langle seq(l[j], j = k\,...\,k+1) \rangle;$
>
> $\mathrm{VL} := [\mathrm{op}(\mathrm{VL}), v];$
>
> $k := k + 2;$
>
> **od**;

$$k := 1$$

$$VL := [\ ]$$

$$n := 2$$

$$v := \begin{bmatrix} a \\ b \end{bmatrix}$$

$$VL := \left[\begin{bmatrix} a \\ b \end{bmatrix}\right]$$

$$k := 3$$

$$v := \begin{bmatrix} c \\ d \end{bmatrix}$$

$$VL := \left[\begin{bmatrix} a \\ b \end{bmatrix}, \begin{bmatrix} c \\ d \end{bmatrix}\right]$$

$$k := 5$$

$$v := \begin{bmatrix} e \\ f \end{bmatrix}$$

$$VL := \left[\begin{bmatrix} a \\ b \end{bmatrix}, \begin{bmatrix} c \\ d \end{bmatrix}, \begin{bmatrix} e \\ f \end{bmatrix}\right]$$

$$k := 7 \tag{33}$$

```
> StringToVects:=proc(text::string, n::posint)
    local L, i, k, v, VL;
    L:=StringToList(text);
     k := 1;
     VL := [];
    for i to nops(L)/n do
      v := <seq(L[j], j = k .. k+1)>;
      VL := [op(VL), v];
      k := k+n;
    end do;
    return(VL);
  end:
```

> $StringToVects(\text{"SomeThing"}, 3);$

$$\left[\left[\begin{array}{c}51\\79\end{array}\right],\left[\begin{array}{c}69\\52\end{array}\right],\left[\begin{array}{c}73\\78\end{array}\right]\right] \qquad \textbf{(34)}$$

```
> StringToVects("SomeThing", 2);
```

$$\left[\left[\begin{array}{c}51\\79\end{array}\right],\left[\begin{array}{c}77\\69\end{array}\right],\left[\begin{array}{c}52\\72\end{array}\right],\left[\begin{array}{c}73\\78\end{array}\right]\right] \qquad \textbf{(35)}$$

```
> StringToVects:=proc(intext::string, n::posint)
   local L, i, k, v, VL, text;

   text:=intext;
   while (length(text) mod n <>0) do
     text:=cat(text,Alphabet[-1]);
   end;
   L:=StringToList(text);
   k := 1;
   VL := [];
   for i to nops(L)/n do
     v := <seq(L[j], j = k .. k+n-1)>;
     VL := [op(VL), v];
     k := k+n;
   end do;
   return(VL);
 end:
```

```
> StringToVects("SomeThing", 2);
```

$$\left[\left[\begin{array}{c}51\\79\end{array}\right],\left[\begin{array}{c}77\\69\end{array}\right],\left[\begin{array}{c}52\\72\end{array}\right],\left[\begin{array}{c}73\\78\end{array}\right],\left[\begin{array}{c}71\\94\end{array}\right]\right] \qquad \textbf{(36)}$$

```
>
```