...A CIFRA DE SUBSTITUICAO...


```
> restart:
  with(StringTools):
  with(combinat,randperm):
  with(group):
>
```

Algumas ferramentas basicas

Conversao de letras em numeros e vice-versa:

```
> convert("a",bytes);
  convert("a",bytes)[1];
  convert("z",bytes)[1]-97;
  convert([97+20],bytes);
  convert(" ",bytes);
> convert(97,binary);
```

Concatenacao de "strings":

```
> cat("b876","x?=656");
```

Calculo do numero de letras de uma palavra:

```
> palavra:="pois";
  m:=length(palavra);
  palavra[2];
>
```

```
>
```

A Cifra de Substituicao

A Chave e a sua Inversa:

```
> chave:=randperm(26);
  chaveinv:=convert(invperm(convert(chave,'disjcyc')),'permlist',
  26);
```

A Chave disposta numa forma mais simpatica... [para aparecer como matriz, pode ter de esconder o menu que aparece por vezes à esquerda, ou reduzir tamanho da letra!]

```
> mtxb:= array(1..2,1..26):
  for i to 2 do
    for j to 26 do
      if i=1 then letra:=j+64: fi:
      if i=2 then letra:=chave[j]+64: fi:
        mtxb[i,j] := convert(convert([letra],bytes),name):
```

```
    od:
  od:
  print(mtxb);
```

A Chave inversa disposta numa forma mais simpatica...

```
> mtxb:= array(1..2,1..26):
  for i to 2 do
    for j to 26 do
      if i=1 then letra:=j+64: fi:
      if i=2 then letra:=chaveinv[j]+64: fi:
        mtxb[i,j] := convert(convert([letra],bytes),name):
    od:
  od:
  convert(mtxb,matrix);
>
```

A mensagem...

```
> Mg:="eles nao sabem nem sonham que o sonho e uma constante da
  vida tao concreta e definida como outra coisa qualquer como esta
  pedra cinzenta em que me sento e descanso como este ribeiro
  manso em serenos sobressaltos como estes pinheiros altos que em
  verde e oiro se agitam como estas aves que gritam em bebedeiras
  de azul eles nao sabem que o sonho e vinho e espuma e fermento
  bichinho alacre e sedento de focinho pontiagudo que fossa
  atraves de tudo num perpetuo movimento eles nao sabem que o
  sonho e tela e cor e pincel base fuste capitel arco em ogiva
  vitral pinaculo de catedral contraponto sinfonia mascara grega
  magia que e retorta de alquimista mapa do mundo distante rosa
  dos ventos infante caravela quinhentista que e cabo da boa
  esperanca ouro canela marfim florete de espadachim bastidor
  passo de danca colombina e arlequim passarola voadora para raios
  locomotiva barco de proa festiva alto forno geradora cisao do
  atomo radar ultra som televisao desembarque em foguetao na
  superficie lunar eles nao sabem nem sonham que o sonho comanda a
  vida que sempre que um homem sonha o mundo pula e avanca como
  bola colorida entre as maos de uma crianca":
  Mg:=SubstituteAll(Mg," ",""):
  mg:=length(Mg);

>
```

Algoritmo de cifracao, fornecendo o **criptograma**, na forma usual (em blocos de 5) ...

```
> Cr:="":
  for i from 1 to mg do
   cr:=convert(Mg[i],bytes)[1]-97+1: cr:=chave[cr]+64:
   Cr:=cat(Cr,convert([cr],bytes)):
  od:
  #
  bloco:=5: lg:=nops(convert(Cr,bytes)): j:=0:
```

```
   for i from 1 to lg do
    j:=j+1: if j=bloco+1 then j:=1: fi:
    printf("%A",Cr[i]):
    if j=bloco then printf(" "): fi:
    if frac(i/(bloco*10))=0 then printf("\n"): fi:
   od:
```

Algoritmo de decifracao:

```
> Dr:="":
   mg:=nops(convert(Cr,bytes)):
   for i from 1 to mg do
    dr:=convert(Cr[i],bytes)[1]-64: dr:=chaveinv[dr]+64:
    Dr:=cat(Dr,convert([dr],bytes)):
   od:
   #
   bloco:=5: lg:=nops(convert(Dr,bytes)): j:=0:
   for i from 1 to lg do
    j:=j+1: if j=bloco+1 then j:=1: fi:
    printf("%A",Dr[i]):
    if j=bloco then printf(" "): fi:
    if frac(i/(bloco*10))=0 then printf("\n"): fi:
   od:
>
>
```

Cripto-analise da cifra de Substituição

```
> length(Cr);
   convert([65+2],bytes);
```

Estudo da frequencia das letras:

```
> Digits:=3:
   mg:=length(Cr):
   for j from 0 to 25 do:
   ct:=0:
   for i from 1 to mg do:
    if Cr[i]=convert([65+j],bytes) then ct:=ct+1:fi:
   od:
   printf("%A  %A (%A)\n",convert([65+j],bytes),ct,evalf(ct/mg)*
   100):
   od:
>
```

Estudo de digramas da forma x- e -x :

```
> letraX:="A":
   printf("(.)  %A- ; -%A\n",letraX,letraX):
   for j from 0 to 25 do:
   c:=0: d:=0:
   for i from 1 to mg do:
```

```
    if Cr[i]=letraX then
     if Cr[i+1]=convert([65+j],bytes) then c:=c+1:fi:
     if i<>1 then if Cr[i-1]=convert([65+j],bytes) then d:=d+1:fi:
   fi: fi:
   od:
   printf("(%A) %A ; %A\n",convert([65+j],bytes),evalf(c/mg)*100,
   evalf(d/mg)*100):
   od:
>
```

Estudo de trigramas da forma xx-...

```
> letra1:="H":
   letra2:="A":
   for j from 0 to 25 do:
   c:=0:
   for i from 1 to mg do:
    if Cr[i]=letra1 then
     if Cr[i+1]=letra2 then
      if Cr[i+2]=convert([65+j],bytes) then c:=c+1:fi:
   fi: fi:
   od:
   printf("(%A) %A\n",convert([65+j],bytes),evalf(c)):
   od:
>
```

Algoritmo de decifracao parcial:

```
> Hipotese1:="HBARL":
   Hipotese2:="AEOSR":
   hip:=nops(convert(Hipotese1,bytes)):
   Dr:="":
   mg:=nops(convert(Cr,bytes)):
   for i from 1 to mg do
      ct:=0:
        for j from 1 to hip do
          if Cr[i]=Hipotese1[j] then dr:=Hipotese2[j]: ct:=1: fi:
        od:
     if ct=0 then dr:="-":fi:
    Dr:=cat(Dr,dr):
   od:
   #
   bloco:=5: lg:=nops(convert(Dr,bytes)): j:=0:
   for i from 1 to lg do
    j:=j+1: if j=bloco+1 then j:=1: fi:
    printf("%A",Dr[i]):
    if j=bloco then printf(" "): fi:
    if frac(i/(bloco*10))=0 then printf("\n"): fi:
   od:
```

Novamente... (e util ter pelo menos dois...)

```
> Hipotese1:="IPQNXT":
```

```
    Hipotese2:="AEOSRM":
    hip:=nops(convert(Hipotese1,bytes)):
    Dr:="":
    mg:=nops(convert(Cr,bytes)):
    for i from 1 to mg do
       ct:=0:
         for j from 1 to hip do
           if Cr[i]=Hipotese1[j] then dr:=Hipotese2[j]: ct:=1: fi:
         od:
      if ct=0 then dr:="-":fi:
     Dr:=cat(Dr,dr):
    od:
    #
    bloco:=5: lg:=nops(convert(Dr,bytes)): j:=0:
    for i from 1 to lg do
     j:=j+1: if j=bloco+1 then j:=1: fi:
     printf("%A",Dr[i]):
     if j=bloco then printf(" "): fi:
     if frac(i/(bloco*10))=0 then printf("\n"): fi:
    od:
  >
>
```