

## **tablvar**

Variations d'une fonction

**Calculatrices :** 92 92+ V200  
**Niveaux :** 1ere 1ere S Tale Tale S Sup  
**Descriptif :** Groupe de programmes - Etude des variations d'une fonction et construction de son tableau de variations  
**Mots-clefs :** variations, fonctions  
**Auteur :** Rémi Garoscio  
**Date de dernière révision :** Septembre 2002

### **Présentation :**

Il s'agit d'un groupe dont les deux programmes principaux sont nommés *tablvar* et *tracet*; le premier effectue tous les calculs liés à l'étude d'une fonction, le deuxième de tracer son tableau de variations.

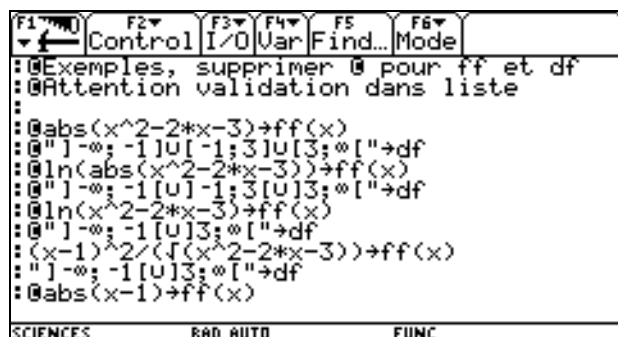
Le groupe contient aussi 3 petits sous-programmes nommés *aon*, *aoff* et *cls*, appelés par les deux précédents.

### **Mode d'emploi :**

#### **Préparatifs** (à effectuer avant exécution) :

Avant tout, vous devez vous placer dans le répertoire *sciences* (ou dans celui dans lequel vous avez copié les fichiers du groupe *tablvar*).

Ensuite, vous devez indiquer quelle est la fonction que vous voulez étudier, et sur quel intervalle : pour cela, vous devez ouvrir le programme *tablvar* en mode édition ([APPS] - 7 - 2 puis choisissez le programme *tablvar* dans la liste). En descendant un peu vous verrez alors quelques lignes mises en commentaires (indiquées par un `:`) : il s'agit d'exemples alors n'hésitez pas à les regarder :



```
F1 F2 F3 F4 F5 F6
Control I/O Var Find... Mode
: @Exemples, supprimer @ pour ff et df
: @Attention validation dans liste
:
: @abs(x^2-2*x-3)+ff(x)
: @ "1-@; -1]U[-1;3]U[3;@[">df
: @ln(abs(x^2-2*x-3))+ff(x)
: @ "1-@; -1]U[-1;3]U[3;@[">df
: @ln(x^2-2*x-3)+ff(x)
: @ "1-@; -1]U[3;@[">df
: (x-1)^2/(sqrt(x^2-2*x-3))+ff(x)
: "1-@; -1]U[3;@[">df
: @abs(x-1)+ff(x)
```

SCIENCES RAD AUTO FUNC

Vous l'avez compris, il suffit donc d'y mettre les deux lignes correspondant aux

affectations de ff (la fonction à étudier) et de df (l'intervalle ou l'union d'intervalles d'étude).

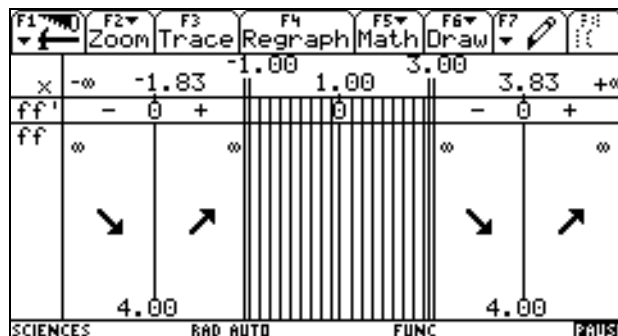
### Exécution :

Enfin, vous pouvez lancer le programme. Il vous suffit de taper :  
`tablvar():tracet()`.

Alors, une fois que *tablvar* a terminé ses calculs vous obtenez cet écran :



Cet écran disparaît aussitôt pour laisser la place au résultat du programme *tracet* :



### Remarques :

L'auteur apporte quelques remarques **concernant *tablvar*** :

- Ne pas inclure de fonctions trigonométriques dans *tablvar*, les résultats périodiques n'étant pas gérés.
- Certaines variables sont déclarées en données globales et sont exploitées par les deux programmes (tamp2, dimtamp1...).
- Placez-vous en mode Float 2 (dans l'écran MODE) afin d'alléger l'affichage du tableau.
- Ne mettez de signe plus pour +inf dans *tablvar*, mais mettez le signe moins pour -inf.
- Pour la version actuelle, si la dérivée s'annule en dehors du domaine de définition, elle sera tout de même affichée.
- Pour l'utilisation de la valeur absolue, il faut inclure dans le domaine de définition la (les) valeur(s) pour la(les)quelle(s) on souhaite une étude particulière du signe, et du

comportement de la fonction à la(aux) frontières en question(s) : pour plus d'informations, consultez l'exemple 5.

Et **concernant *tracet*** :

- echxy est le facteur de réduction proportionnel du tableau.
- echx est l'échelle de réduction en X.
- echy est l'échelle de réduction en Y.

Vous pouvez les multiplier par un nombre différent de 1 pour peaufiner le dessin.

### Sources :

Aoff ()

Prgm

setGraph("axes","off")

EndPrgm

Aon ()

Prgm

setGraph("axes","on")

EndPrgm

Cls ()

Prgm

ClrDraw

EndPrgm

Tabl var ()

Prgm

FnOff

aoff()

cls()

PxlText "Rémy GAROSCI0 ",10,81

PxlText "Tél./Fax :+377/93.15.02.07",20,41

PxlText "P/Texas Instruments",30,63

PxlText "Tableau de variation",40,60

PxlText "d'une fonction V 1.01",50,57

PxlText "Traitement en cours...",70,46

PxlLine 79,47,79,192

PxlLine 92,47,92,192

PxlLine 79,47,92,47

PxlLine 79,192,92,192

©Exemples, supprimer © pour ff et df

©Attention validation dans liste

©abs(x^2-2\*x-3)→ff(x)

©"]-∞;-1]∪[-1;3]∪[3;∞["→df

©ln(abs(x^2-2\*x-3))→ff(x)

```

@"]^-∞;-1[∪]^-1;3[∪]3;∞["→df
@ln(x^2-2*x-3)→ff(x)
@"]^-∞;-1[∪]3;∞["→df
(x-1)^2/(√(x^2-2*x-3))→ff(x)
"]^-∞;-1[∪]3;∞["→df
@abs(x-1)→ff(x)
@"]^-∞;1[∪]1;∞["→df
@ln(x)→ff(x)
@"]∅;∞["→df
@1/x→ff(x)
@"]^-∞;∅[∪]∅;∞["→df
@√(x)→ff(x)
@"]∅;∞["→df
@ln(x)/(2*x-1)→ff(x)
@"]∅;.5[∪].5;∞["→df
@(-3*x^2+x+10)/(√(x^2-2*x-3))→ff(x)
@"]^-∞;-1[∪]3;∞["→df
@e^(3/(x-1))→ff(x)
@"]^-∞;1[∪]1;∞["→df
@(2*x+1)/(9*x^2-81)→ff(x)
@"]^-∞;-3[∪]^-3;3[∪]3;∞["→df
@(x-1)^2/(√(x^2-2*x-3))→ff(x)
@"]^-∞;-1[∪]3;∞["→df
@-9*x^2+3*x^2-5+√(x+4)→ff(x)
@"]^-∞;-4["→df
@ln(x)/(x-1)→ff(x)
@"]∅;1[∪]1;∞["→df
@(x^4-5*x^3+2)/(x^6-4*x^2-10)→ff(x)
@"]^-∞;-1.66157[∪]^-1.66157;1.66157[∪]1.66157;∞["→df
@(9*x^2-8*x+3)/(x^3-5*x^2+5)→ff(x)
@"]^-∞;-.919[∪]^- .919;1.14[∪]1.14;4.78[∪]4.78;∞["→df
@x^3-5*x^2+5→ff(x)
@"]^-∞;∞["→df
@3*x+1→ff(x)
@"]^-∞;∞["→df
@x^7-x^4+6*x^2-1→ff(x)
@"]^-∞;∞["→df
@2*x^2*ln(x)-1→ff(x)
@"]∅;∞["→df

```

DelVar

dfp,der∅,der1,der2,tamp1,tamp2,crn,dimtamp1,dimder∅,matn11,matder∅,tmp

Local

i,ii,j,k,posc1,nca,ni,crn,n11,matn11,matder∅,dimder∅,der1,der2,llder1,di  
mdfp,precty,prectx,trectx,nrect,eps

d(ff(x),x)→fg(x)

48→prectx:80→precty:16→trectx:0→nrect

@1 nb intervalle

1→ni

1→posc1

dim(df)→nca

```
newMat(nca,1)→crn
```

```
RplcPic rectang,precty,prectx+nrect*trectx
```

```
@2 reperage union
```

```
For i,1,nca
```

```
  If mid(df,i,1)="∪" Then
```

```
    ni+1→ni
```

```
  EndIf
```

```
  If mid(df,i,1)="]" Then
```

```
    i→crn[posc1,1]
```

```
    posc1+1→posc1
```

```
  EndIf
```

```
  If mid(df,i,1)="[" Then
```

```
    i→crn[posc1,1]
```

```
    posc1+1→posc1
```

```
  EndIf
```

```
EndFor
```

```
2*ni→n11
```

```
newMat(ni,11)→dfp
```

```
nrect+1→nrect
```

```
RplcPic rectang,precty,prectx+nrect*trectx
```

```
@3 archivage donnees dont limites
```

```
For i,1,ni
```

```
  mid(df,crn[i*2-1,1],crn[i*2,1]-crn[i*2-1,1]+1)→dfp[i,1]
```

```
  left(df[i,1],1)→dfp[i,2]
```

```
  right(df[i,1],1)→dfp[i,5]
```

```
  inString(dfp[i,1],";",2)→posc1
```

```
  expr(mid(dfp[i,1],2,posc1-2))→dfp[i,3]
```

```
  expr(mid(dfp[i,1],posc1+1,dim(dfp[i,1])-posc1-1))→dfp[i,4]
```

```
  If dfp[i,2]="]" Then
```

```
    1→dfp[i,6]
```

```
    limit(ff(x),x,dfp[i,3],1)→dfp[i,8]
```

```
  Else
```

```
    ff(dfp[i,3])→dfp[i,8]
```

```
  EndIf
```

```
  If dfp[i,5]="[" Then
```

```
    1→dfp[i,7]
```

```
    limit(ff(x),x,dfp[i,4],-1)→dfp[i,9]
```

```
  Else
```

```
    ff(dfp[i,4])→dfp[i,9]
```

```
  EndIf
```

```
EndFor
```

```
nrect+1→nrect
```

```
RplcPic rectang,precty,prectx+nrect*trectx
```

```
@4 reperage nb de lignes
```

```
For i,1,ni
```

```
  If i<ni Then
```

```
    If i=1 Then
```

```

    If dfp[i,3]=-∞ Then
        n11-1→n11
    Else
        1→dfp[i,10]
    EndIf
EndIf
If dfp[i+1,4]=∞ Then
    n11-1→n11
EndIf
If dfp[i+1,3]=dfp[i,4] Then
    n11-1→n11
    1→dfp[i,11]
Else
    1→dfp[i+1,10]
    1→dfp[i,11]
EndIf
EndIf
If ni=1 Then
    2→n11
    If dfp[i,3]=-∞ Then
        n11-1→n11
    Else
        1→dfp[i,10]
    EndIf
    If dfp[i,4]=∞ Then
        n11-1→n11
    Else
        1→dfp[i,11]
    EndIf
EndIf
EndFor

```

nrect+1→nrect

RplcPic rectang,precty,prectx+nrect\*trectx

©5 reperage des valeurs fg(x)=0

© archivage bornes

```

0→matn11
0→matder0
zeros(fg(x),x)→der0
CopyVar der0,der
dim(der0)→dimder0
dimder0→j
If dimder0>0 Then
    0→k
    For i,1,dimder0
        If string(der0[i])="-∞" or string(der0[i])="∞" Then
            j-1→j
        Else
            1+k→k
            der0[i]→der[k]
        EndIf
    EndFor
DelVar der0

```

```

left(der,k)→derØ
dim(derØ)→dimderØ
EndIf
If dimderØ>Ø Then
  newMat(dimderØ,1)→der1
  newMat(dimderØ,5)→der2
  1→matderØ   © flag de creation
EndIf

If nll>Ø Then
  newMat(nll,5)→tamp1
  1→matnll   © flag de creation
  1→j
  For i,1,ni
    If dfp[i,1Ø]=1 Then
      dfp[i,3]→tamp1[j,1]
      approx(dfp[i,3])→tamp1[j,2]
      i→tamp1[j,3]
      3→tamp1[j,4]
      If dfp[i,6]=1 Then
        1→tamp1[j,5]
      EndIf
      j+1→j
    EndIf
    If dfp[i,11]=1 Then
      dfp[i,4]→tamp1[j,1]
      approx(dfp[i,4])→tamp1[j,2]
      i→tamp1[j,3]
      4→tamp1[j,4]
      If dfp[i,7]=1 Then
        1→tamp1[j,5]
      EndIf
      j+1→j
    EndIf
  EndFor
Else
  Ø→dimtamp1
EndIf
tamp1→tmp

nrect+1→nrect
RplcPic rectang,precty,prectx+nrect*trectx

©6 test valeurs communes ff et fg
If dimderØ>Ø Then
  Ø→k
  Ø→lder1
  For i,1,dimderØ
    If matnll=1 Then
      For j,1,nll
        If tamp1[j,2]=approx(derØ[i]) Then
          © nota string pour expr trigo
          1→k   © flag
        EndIf
      EndFor
    EndIf
    If k=Ø Then

```

```

        1+lder1→lder1
        der0[i]→der1[lder1,1]
    EndIf
    0→k
Else
    1+lder1→lder1
    der0[i]→der1[lder1,1]
EndIf
EndFor
© reorganisation dans der2
For i,1,lder1
    der1[i,1]→der2[i,1]
    approx(der1[i,1])→der2[i,2]
    Try
        ff(der2[i,1])
    Else
        1→der2[i,5]
    EndTry
EndFor
EndIf

nrect+1→nrect
RplcPic rectang,precty,prectx+nrect*trectx

©7 rajout val selon conditions

If matder0=1 Then
    If matn11=1 Then
        (augment(tamp1T,der2T))T→tamp1
    Else
        der2→tamp1
        colDim(tamp1T)→dimtamp1
    EndIf
EndIf

nrect+1→nrect
RplcPic rectang,precty,prectx+nrect*trectx

©8 tri des valeurs

If matn11=1 Then
    colDim(tamp1T)→dimtamp1
    If dimtamp1≥2 Then
        0→k
        Loop
            For i,1,dimtamp1-1
                If tamp1[i,2]>tamp1[i+1,2] Then
                    rowSwap(tamp1,i,i+1)→tamp1
                    1→k
                EndIf
            EndFor
            If k=0
                Exit
            0→k
        EndLoop
    EndIf
EndIf

```



```

EndIf

nrect+1→nrect
RplcPic rectang,precty,prectx+nrect*trectx

@9 creation zones tableau

If dimtamp1>0 Then
  augment(augment([[ $-\infty$ ][ $-\infty$ ][ $-2$ ][0][1]],tamp1T),[[ $\infty$ ][ $\infty$ ][ $-1$ ][0][1]])→tamp1
  dimtamp1+2→dimtamp1
Else
  2→dimtamp1
  newMat(5,2)→tamp1
  [[ $-\infty$ , $\infty$ ][ $-\infty$ , $\infty$ ][ $-2$ , $-1$ ][0,0][1,1]]→tamp1
EndIf
@colDim(dfT)→dimdfp
newMat(24,dimtamp1-1)→tamp2
For i,1,dimtamp1-1
  If dimtamp1>2 Then

    @ etude borne gauche

    If tamp2[2,i]=0 Then
      i→tamp2[1,i]
      1→tamp2[2,i]
      tamp1[3,i]→tamp2[3,i]
      tamp1[1,i]→tamp2[4,i]
      tamp1[2,i]→tamp2[5,i]
      tamp1[5,i]→tamp2[7,i]
      Try
        fg(tamp2[4,i])→tamp2[8,i]
        approx(tamp2[8,i])→tamp2[9,i]
        If tamp2[8,i]=undef or tamp2[8,i]= $-\infty$  or tamp2[8,i]= $\infty$  Then
          1→tamp2[6,i]
        EndIf
      Else
        1→tamp2[6,i]
        If errornum=800 Then
          "NRR"→tamp2[8,i]
          "NRR"→tamp2[9,i]
          ClrErr
        EndIf
      EndTry
      If tamp2[7,i]=1 Then
        If tamp1[3,i]>0 Then
          If ni≥1 and i>1 Then @ni> ou≥ ?
            If tamp1[3,i]≤ni Then
              If tamp1[4,i]=4 Then
                dfp[tamp1[3,i]+1,8]→tamp2[10,i]
                approx(dfp[tamp1[3,i]+1,8])→tamp2[11,i]
              Else
                dfp[tamp1[3,i],8]→tamp2[10,i]
                approx(dfp[tamp1[3,i],8])→tamp2[11,i]
              EndIf
            EndIf
          EndIf
        EndIf
      EndIf
    EndIf
  EndIf
EndFor

```

```

Else
  If tamp1[3,i]<0 Then
    dfp[1,8]→tamp2[10,i]
    approx(dfp[1,8])→tamp2[11,i]
  EndIf
EndIf
Else
  ff(tamp2[4,i])→tamp2[10,i]
  approx(tamp2[10,i])→tamp2[11,i]
EndIf
EndIf

© etude borne droite

i+1→ii
If tamp2[12,i]=0 Then
  1→tamp2[12,i]
  tamp1[3,ii]→tamp2[13,i]
  tamp1[1,ii]→tamp2[14,i]
  tamp1[2,ii]→tamp2[15,i]
  tamp1[5,ii]→tamp2[17,i]
Try
  fg(tamp2[14,i])→tamp2[18,i]
  approx(tamp2[18,i])→tamp2[19,i]
  If tamp2[18,i]=undef or tamp2[18,i]=-∞ or tamp2[18,i]=∞ Then
    1→tamp2[16,i]
  EndIf
Else
  1→tamp2[16,i]
  If errornum=8000 Then
    "NRR"→tamp2[18,i]
    "NRR"→tamp2[19,i]
    ClrErr
  EndIf
EndTry
EndTry
If tamp2[17,i]=1 Then
  If tamp1[3,ii]>0 Then
    dfp[tamp1[3,ii],tamp1[4,ii]+5]→tamp2[20,i]
    approx(dfp[tamp1[3,ii],tamp1[4,ii]+5])→tamp2[21,i]
  Else
    If tamp1[3,ii]<0 Then
      dfp[ni,9]→tamp2[20,i]
      approx(dfp[ni,9])→tamp2[21,i]
    EndIf
  EndIf
Else
  ff(tamp2[14,i])→tamp2[20,i]
  approx(tamp2[20,i])→tamp2[21,i]
EndIf
EndIf
Try
  .01→eps
  If i=dimtamp1-1 Then
    ff(tamp2[4,i]+eps)
    sign(fg(tamp2[4,i]+eps))→tamp2[24,i]
    char(abs(tamp2[24,i]-44))→tamp2[23,i]
  EndIf
EndTry

```

```

Else
  ff(tamp2[14,i]-eps)
  sign(fg(tamp2[14,i]-eps))→tamp2[24,i]
  char(abs(tamp2[24,i]-44))→tamp2[23,i]
EndIf
Else
  1→tamp2[22,i]
EndTry
Else
  1→tamp2[1,1]
  1→tamp2[2,1]
  tamp1[3,1]→tamp2[3,1]
  tamp1[1,1]→tamp2[4,1]
  tamp1[2,1]→tamp2[5,1]
  tamp1[5,1]→tamp2[7,1]
  1→tamp2[6,1]
  dfp[1,8]→tamp2[10,1]
  dfp[1,8]→tamp2[11,1]
  tamp1[3,2]→tamp2[13,1]
  tamp1[1,2]→tamp2[14,1]
  tamp1[2,2]→tamp2[15,1]
  tamp1[5,2]→tamp2[17,1]
  1→tamp2[16,1]
  dfp[1,9]→tamp2[20,1]
  dfp[1,9]→tamp2[21,1]
  sign(d(ff(x),x))→tamp2[24,1]
  char(abs(tamp2[24,1]-44))→tamp2[23,1]
EndIf
EndFor

Px|Text "Succès des opérations !",70,46
nrect+1→nrect
RplcPic rectang,precty,prectx+nrect*trectx
EndPrgm

Tracet ()
Prgm
Local
i,j,postx,posty,echxy,echx,echy,lvx1,lhy1,lhy2,lontx,lonty,lonutty,lonu
ttx,taintx,const,deply

Ø→postx:Ø→posty
1→echxy
1*echxy→echx  @Ø.49 p/left/right  .98 p/top/bottom

1*echxy→echy  @Ø.47 p/top/bottom Ø.98 p/left/right
postx+20→lvx1
posty+16→lhy1
posty+26→lhy2
int(238*echx)→lontx @238 max si postx=Ø
int(102*echy)→lonty @102 max si posty=Ø
lontx-lvx1→lonuttx
lonty-lhy2→lonutty
int(lonuttx/(dimp1-1))→taintx
cls()
aoff()

```

```

Px1Line lhy1,postx,lhy1,lontx
Px1Line lhy2,postx,lhy2,lontx
Px1Line posty,lvx1,lonty,lvx1
Px1Text "x",lhy1-8,lvx1-10
Px1Text "ff'",lhy2-8,lvx1-18
Px1Text "ff ",lhy2+2,lvx1-18
Px1Text "-∞",lhy1-8,lvx1+1
Px1Text "+∞",lhy1-8,postx+lontx-11

For i,1,dimtamp1-1
    lvx1+taintx*i→const
    If i≠dimtamp1-1 Then
        If tamp2[13,i]≠0 Then
            lhy1-7→deply
            If tamp2[17,i]=1 Then
                Px1Line deply,const-1,lonty,const-1
                Px1Line deply,const+1,lonty,const+1
            Else
                Px1Line deply,const,lonty,const
            EndIf
            Px1Text string(tamp2[15,i]),posty,const-
int(dim(string(tamp2[15,i]))/2)*5
        Else
            lhy1-1→deply
            Px1Line deply,const,lhy2,const
            Px1Text "0",lhy1+2,const-3
            If tamp2[17,i]=1 Then
                Px1Line lhy2,const-1,lonty,const-1
                Px1Line lhy2,const+1,lonty,const+1
            Else
                Px1Line lhy2,const,lonty,const
            EndIf
            Px1Text string(tamp2[15,i]),deply-8,const-
int(dim(string(tamp2[15,i]))/2)*5
        EndIf
    EndIf
    If tamp2[22,i]=1 Then
        For j,lvx1+taintx*(i-1)+5,const-3,4
            Px1Line lhy1,j,lonty,j
        EndFor
    Else
        Px1Text tamp2[23,i],lhy1+2,int(const-taintx/2-3)
        If tamp2[23,i]="+" Then
            RplcPic flht,int(lonutty/2+lhy2-5),int(const-taintx/2-5)
        Else
            RplcPic flbs,int(lonutty/2+lhy2-5),int(const-taintx/2-5)
        EndIf
        If tamp2[7,i]=1 Then
            Px1Text
string(tamp2[11,i]),int(lonutty/2+lhy2+tamp2[24,i]*(lonutty/2-
7)),int(const-taintx+3)
        Else
            Px1Text
string(tamp2[11,i]),int(lonutty/2+lhy2+tamp2[24,i]*(lonutty/2-
7)),int(const-taintx-dim(string(tamp2[11,i]))*3.75)
        EndIf

```

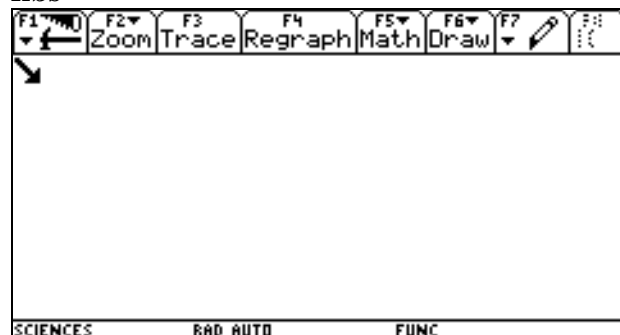
```

If tamp2[17,i]=1 Then
  PxlText string(tamp2[21,i]),int(lonutty/2+1hy2-
tamp2[24,i]*(lonutty/2-7)),int(const-dim(string(tamp2[21,i]))*7.5)
Else
  EndIf
EndIf
EndFor
Pause
EndPrgm

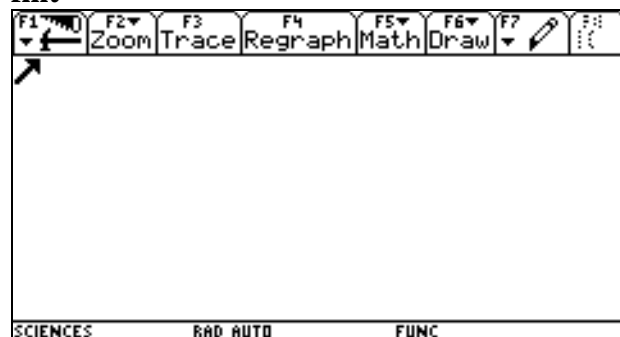
```

**Attention : vous devez posséder ces 3 images :**

**flbs**



**flht**



**rectang**

