

Triangle rectangle

Xcas

```

trirectangle():={
local xA,xB,xC,yA,yB,yC,a,b,c ;
saisir(xA,yA);
saisir(xB,yB);
saisir(xC,yC);
a:=(xB-xA)^2+(yB-yA)^2;
b:=(xC-xB)^2+(yC-yB)^2;
c:=(xC-xA)^2+(yC-yA)^2;
si (a==b+c) alors
  afficher("rectangle en c")
fsi
si (b==a+c) alors
  afficher("rectangle en B")
fsi
si (c==b+a) alors
  afficher ("rectangle en A")
fsi
si (a!=b+c et b!=a+c et c!=a+b) alors
  afficher("pas rectangle")
fsi
}

```

ALGOBOX

```

VARIABLES
- xA EST_DU_TYPE NOMBRE
- xB EST_DU_TYPE NOMBRE
- xC EST_DU_TYPE NOMBRE
- yA EST_DU_TYPE NOMBRE
- yB EST_DU_TYPE NOMBRE
- yC EST_DU_TYPE NOMBRE
- a EST_DU_TYPE NOMBRE
- b EST_DU_TYPE NOMBRE
- c EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
- LIRE xA
- LIRE yA
- LIRE xB
- LIRE yB
- LIRE xC
- LIRE yC
- a PREND_LA_VALEUR pow(xB-xA,2)+pow(yB-yA,2)
- b PREND_LA_VALEUR pow(xC-xA,2)+pow(yC-yA,2)
- c PREND_LA_VALEUR pow(xC-xB,2)+pow(yC-yB,2)
SI (c==a+b) ALORS
  DEBUT_SI
  - AFFICHER "Le triangle est rectangle en A"
  FIN_SI
SI (b==a+c) ALORS
  DEBUT_SI
  - AFFICHER "Le triangle est rectangle en B"
  FIN_SI
SI (a==b+c) ALORS
  DEBUT_SI
  - AFFICHER "Le triangle est rectangle en C"
  FIN_SI
SI (a!=b+c ET b!=a+c ET c!=a+b) ALORS
  DEBUT_SI
  - AFFICHER "Le triangle n'est pas rectangle"
  FIN_SI
FIN_ALGORITHME

```

Équation de droite

Xcas

```

Eqtdroite():={
local xA,yA,xB,yB,a,b;
saisir(xA,yA);
saisir(xB,yB);
si xA!=xB alors
  a:=(yB-yA)/(xB-xA);
  b:=yA-a*xA;
  afficher("(AB) a pour équation y = "+a+"x+ "+b);
sinon
  afficher("(AB) a pour équation x = "+xA);
fsi
}

```

ALGOBOX

```

VARIABLES
- xA EST_DU_TYPE NOMBRE
- yA EST_DU_TYPE NOMBRE
- xB EST_DU_TYPE NOMBRE
- yB EST_DU_TYPE NOMBRE
- a EST_DU_TYPE NOMBRE
- b EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
- LIRE xA
- LIRE yA
- LIRE xB
- LIRE yB
SI (xA!=xB) ALORS
  DEBUT_SI
  - a PREND_LA_VALEUR (yB-yA)/(xB-xA)
  - b PREND_LA_VALEUR yA-a*xA
  - AFFICHER "(AB) a pour équation y = "
  - AFFICHER a
  - AFFICHER "x + "
  - AFFICHER b
  FIN_SI
SINON
  DEBUT_SINON
  - AFFICHER "(AB) a pour équation x = "
  - AFFICHER xA
  FIN_SINON
FIN_ALGORITHME

```

Points alignés

En utilisant les coefficients directeurs

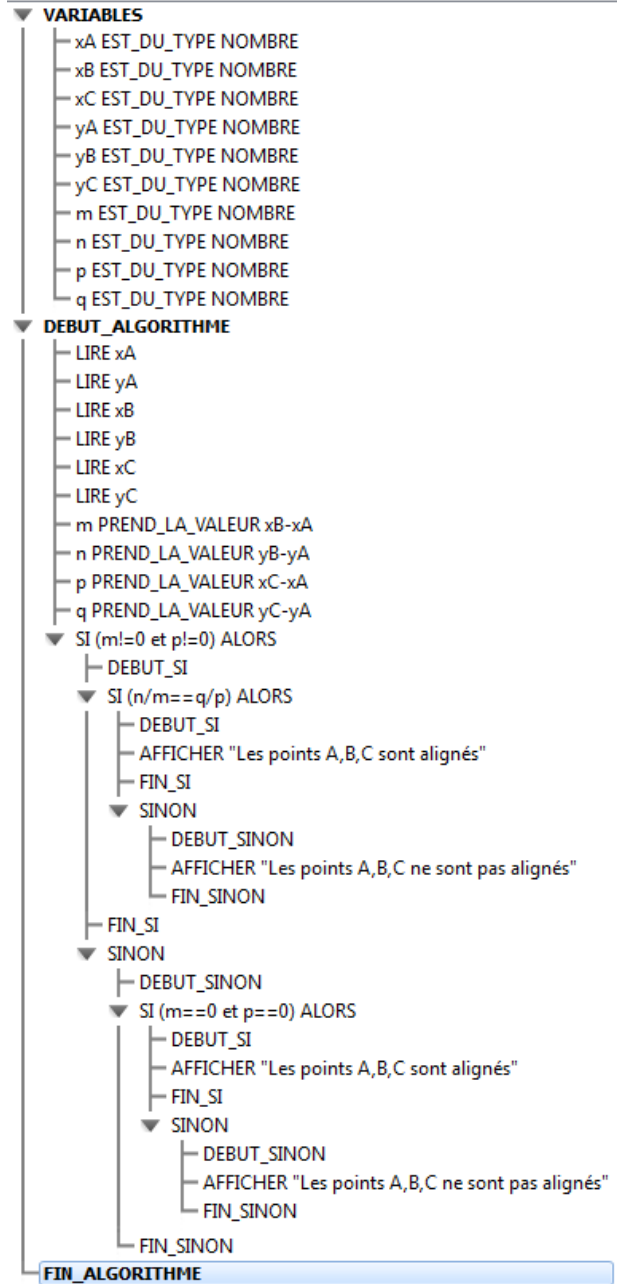
Xcas

```

points_aignés():= {
local xA,xB,xC,yA,yB,yC,m,n,p,q ;
saisir(xA,yA);
saisir(xB,yB);
saisir(xC,yC);
m:=xB-xA;
n:=yB-yA;
p:=xC-xA;
q:=yC-yA;
si m!=0 et p!=0 alors
  si n/m==q/p alors
    afficher("Les points A, B et C sont
alignés")
  sinon
    afficher("Les points A, B et C ne sont pas
alignés")
  fsi
sinon
  si m==0 et p==0 alors
    afficher("Les points A, B et C sont
alignés")
  sinon
    afficher("Les points A, B et C ne sont pas
alignés")
  fsi
fsi
}
;;

```

ALGOBOX



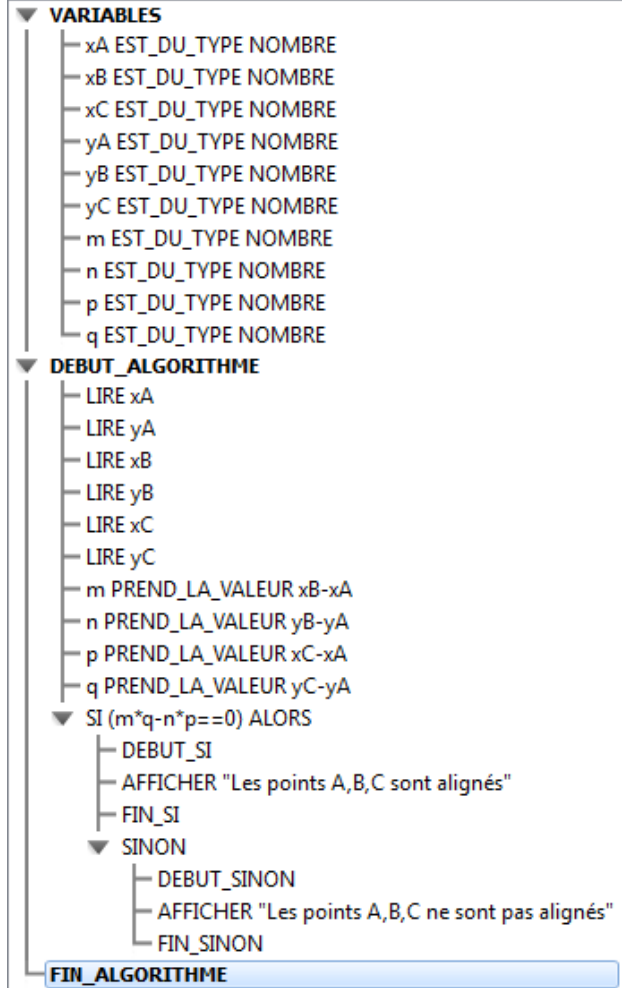
Points alignés

En utilisant les vecteurs

Xcas

```
points_alignés():={
local xA,xB,xC,yA,yB,yC,m,n,p,q ;
saisir(xA,yA);
saisir(xB,yB);
saisir(xC,yC);
m:=xB-xA;
n:=yB-yA;
p:=xC-xA;
q:=yC-yA;
si m*q-n*p ==0 alors
  afficher("Les points A, B et C sont alignés")
sinon
  afficher("Les points A, B et C ne sont pas alignés")
fsi
}
::
```

ALGOBOX



Parallélogramme

Xcas

```
para():={  
  local xA, yA, xB, yB, xC, yC, xD, yD;  
  saisir(xA,yA);  
  saisir(xB,yB);  
  saisir(xC,yC);  
  si (xD:=xC-xB+xA) et (yD:=yC-yB+yA) alors  
    afficher("ABCD est un parallélogramme")  
  sinon  
    afficher("ABCD n'est pas un  
              parallélogramme")  
  fsi  
}  
::
```

ALGOBOX

