IREM d'Aquitaine
Institut de Recherche sur l'Enseignement des Mathématiques
40, rue Lamartine, 33400 Talence

Triangles isocèles

Fiche élève

On souhaite produire un algorithme permettant de déterminer si un triangle est ou non isocèle.

Travail préparatoire

1. On se place dans un repère orthonormé (O ; I, J). On considère les points A(1;2), B(3;-1) et C(4;0). Le triangle ABC est-il isocèle en A ?

2. L'algorithme suivant (Algobox), incomplet, permet de calculer le carré de la distance entre les points A(xA;yA) et B(xB;yB) :

```
CarreDistance  -  29.01.2015
****************************************
Cet algorithme calcule le carré de la distance entre deux points A(xA;yA)
et B(xB;yB).
****************************************
1   VARIABLES
2     xA EST_DU_TYPE NOMBRE
3     yA EST_DU_TYPE NOMBRE
4     xB EST_DU_TYPE NOMBRE
5     yB EST_DU_TYPE NOMBRE
6     carre_distance EST_DU_TYPE NOMBRE
7   DEBUT_ALGORITHME
8     AFFICHER "Entrez les coordonnées xA et yA du point A"
9     LIRE xA
10    LIRE yA
11    AFFICHER "Entrez les coordonnées xB et yB du point B"
12    LIRE xB
13    LIRE yB
14    carre_distance PREND_LA_VALEUR .....................................
15    AFFICHER "Le carré de la distance AB est "
16    AFFICHER carre_distance
17  FIN_ALGORITHME
```

Créer cet algorithme sous Algobox et le compléter.

Tester l'algorithme avec les points A(1;2) et B(3;-1) et vérifier que le résultat est correct.

Conception d'algorithme

3. Modifier l'algorithme précédent afin qu'à partir des coordonnées de trois points A, B et C, l'algorithme renvoie la valeur des carrés des distances AB, BC et AC.
4. À l'aide de cet algorithme, compléter le tableau suivant (dans chaque cas, calculer les carrés des distances AB, BC et AC, puis déterminer la nature du triangle ABC) :

A(-4;1), B(-1;3) et C(2;1)	AB^2 = ...	BC^2 = ...	AC^2 = ...
Le triangle ABC est :	☐ isocèle en ……	☐ équilatéral	☐ autre
A(-1;1), B(1;5) et C(3;3)	AB^2 = ...	BC^2 = ...	AC^2 = ...
Le triangle ABC est :	☐ isocèle en ……	☐ équilatéral	☐ autre
A(-2;3), B(4;5) et C(2;1)	AB^2 = ...	BC^2 = ...	AC^2 = ...
Le triangle ABC est :	☐ isocèle en ……	☐ équilatéral	☐ autre
A(-3;4), B(-1;6) et C(0;2)	AB^2 = ...	BC^2 = ...	AC^2 = ...
Le triangle ABC est :	☐ isocèle en ……	☐ équilatéral	☐ autre
A(-1;1), B(3;1) et C(1;4,5)	AB^2 = ...	BC^2 = ...	AC^2 = ...
Le triangle ABC est :	☐ isocèle en ……	☐ équilatéral	☐ autre

5. Écrire un algorithme (Algobox) qui vérifie si un triangle ABC est isocèle en A.

Extension de l'algorithme

6. Modifier l'algorithme précédent afin qu'il détermine, à partir des coordonnées de trois points A, B et C, si le triangle est isocèle et, si oui, en quel point.

IREM d'Aquitaine
Institut de Recherche sur l'Enseignement des Mathématiques
40, rue Lamartine, 33400 Talence

Triangles isocèles

Fiche enseignant

Objectifs. Mise en œuvre d'un algorithme de calcul de distance et utilisation de la structure conditionnelle `SI-ALORS-SINON`. Utilisation d'Algobox.

Prérequis. Notions de base d'algorithmique (notion de variable, entrées-sorties), calcul de distance.

On souhaite produire un algorithme permettant de déterminer si un triangle est ou non isocèle.

Travail préparatoire

1. On se place dans un repère orthonormé (O ; I, J). On considère les points A(1;2), B(3;-1) et C(4;0). Le triangle ABC est-il isocèle en A ?

Nous avons :
AB = rac((3-1)²+(-1-2)²) = rac(4+9) = rac(13), et
AC = rac((4-1)²+(0-2)²) = rac(9+4) = rac(13).
Ainsi, AB = AC et le triangle est donc isocèle en A.

Le professeur demande aux élèves s'il est nécessaire de passer à la racine carrée ou si la comparaison des longueurs au carré suffit.

Il induira les élèves sur la comparaison des carrés dans l'algorithme (notamment afin d'éviter les erreurs liées aux arrondis machine).

2. L'algorithme suivant (Algobox), incomplet, permet de calculer le carré de la distance entre les points A(xA;yA) et B(xB;yB) :

```
CarreDistance  -  29.01.2015
****************************************
Cet algorithme calcule le carré de la distance entre deux points A(xA;yA)
et B(xB;yB).
****************************************
1   VARIABLES
2     xA EST_DU_TYPE NOMBRE
3     yA EST_DU_TYPE NOMBRE
4     xB EST_DU_TYPE NOMBRE
5     yB EST_DU_TYPE NOMBRE
6     carre_AB EST_DU_TYPE NOMBRE
7   DEBUT_ALGORITHME
8     AFFICHER "Entrez les coordonnées xA et yA du point A"
9     LIRE xA
10    LIRE yA
11    AFFICHER "Entrez les coordonnées xB et yB du point B"
12    LIRE xB
13    LIRE yB
```

```
14    carre_AB PREND_LA_VALEUR .....................................
15    AFFICHER "Le carré de la distance AB est "
16    AFFICHER carre_AB
17  FIN_ALGORITHME
```

Créer cet algorithme sous Algobox et le compléter.

La ligne à compléter est la suivante :

```
14    carre_distance PREND_LA_VALEUR (xB-xA)*(xB-xA) + (yB-yA)*(yB-yA)
```

Tester l'algorithme avec les points A(1;2) et B(3;-1) et vérifier que le résultat est correct.

Il l'est !...

Conception d'algorithme

3. Modifier l'algorithme précédent afin qu'à partir des coordonnées de trois points A, B et C, l'algorithme renvoie la valeur des carrés des distances AB, BC et AC.

Il suffit d'étendre l'algorithme précédent. On obtient :

```
CarreDistancesAB_BC_AC  -  29.01.2015
****************************************
Cet algorithme calcule le carré des distances entre trois points A(xA;yA),
B(xB;yB) et C(xC;yC).
****************************************
1   VARIABLES
2     xA EST_DU_TYPE NOMBRE
3     yA EST_DU_TYPE NOMBRE
4     xB EST_DU_TYPE NOMBRE
5     yB EST_DU_TYPE NOMBRE
6     xC EST_DU_TYPE NOMBRE
7     yC EST_DU_TYPE NOMBRE
8     carre_AB EST_DU_TYPE NOMBRE
9     carre_BC EST_DU_TYPE NOMBRE
10    carre_AC EST_DU_TYPE NOMBRE
11  DEBUT_ALGORITHME
12    AFFICHER "Entrez les coordonnées xA et yA du point A"
13    LIRE xA
14    LIRE yA
15    AFFICHER "Entrez les coordonnées xB et yB du point B"
16    LIRE xB
17    LIRE yB
18    AFFICHER "Entrez les coordonnées xC et yC du point C"
19    LIRE xC
20    LIRE yC
21    carre_AB PREND_LA_VALEUR (xB-xA)*(xB-xA) + (yB-yA)*(yB-yA)
22    carre_BC PREND_LA_VALEUR (xC-xB)*(xC-xB) + (yC-yB)*(yC-yB)
23    carre_AC PREND_LA_VALEUR (xC-xA)*(xC-xA) + (yC-yA)*(yC-yA)
24    AFFICHER "Le carré de la distance AB est "
25    AFFICHER carre_AB
26    AFFICHER "Le carré de la distance BC est "
27    AFFICHER carre_BC
28    AFFICHER "Le carré de la distance AC est "
```

```
29    AFFICHER carre_AC
30  FIN_ALGORITHME
```

4. À l'aide de cet algorithme, compléter le tableau suivant (dans chaque cas, calculer les carrés des distances AB, BC et AC, puis déterminer la nature du triangle ABC) :

A(-4;1), B(-1;3) et C(2;1)	AB^2 = 13	BC^2 = 13	AC^2 = 36	
Le triangle ABC est :	◆ isocèle en B…	☐ équilatéral	☐ autre	
A(-1;1), B(1;5) et C(3;3)	AB^2 = 20.	BC^2 = 8	AC^2 = 20	
Le triangle ABC est :	◆ isocèle en A…	☐ équilatéral	☐ autre	
A(-2;3), B(4;5) et C(2;1)	AB^2 = 40.	BC^2 = 20	AC^2 = 20	
Le triangle ABC est :	◆ isocèle en C…	☐ équilatéral	☐ autre	
A(-3;4), B(-1;6) et C(0;2)	AB^2 = 8.	BC^2 = 17	AC^2 = 13	
Le triangle ABC est :	☐ isocèle en ……	☐ équilatéral	◆ autre	
A(-1;1), B(3;1) et C(1;4,5)	AB^2 = 16.	BC^2 = 16,25 ..	AC^2 = 16,25 ..	
Le triangle ABC est :	◆ isocèle en C…	☐ équilatéral	☐ autre	

5. Écrire un algorithme (Algobox) qui vérifie si un triangle ABC est isocèle en A.

Cette fois, il suffit de calculer les carrés des distances AB et AC et de vérifier leur égalité. On obtient l'algorithme suivant :

```
TriangleIsoceleEnA  -  29.01.2015
****************************************
Cet algorithme détermine si un triangle ABC est isocèle en A.
****************************************
1   VARIABLES
2     xA EST_DU_TYPE NOMBRE
3     yA EST_DU_TYPE NOMBRE
4     xB EST_DU_TYPE NOMBRE
5     yB EST_DU_TYPE NOMBRE
6     xC EST_DU_TYPE NOMBRE
7     yC EST_DU_TYPE NOMBRE
8     carre_AB EST_DU_TYPE NOMBRE
9     carre_AC EST_DU_TYPE NOMBRE
10  DEBUT_ALGORITHME
11    AFFICHER "Entrez les coordonnées xA et yA du point A"
12    LIRE xA
13    LIRE yA
14    AFFICHER "Entrez les coordonnées xB et yB du point B"
15    LIRE xB
16    LIRE yB
17    AFFICHER "Entrez les coordonnées xC et yC du point C"
18    LIRE xC
19    LIRE yC
20    carre_AB PREND_LA_VALEUR (xB-xA)*(xB-xA) + (yB-yA)*(yB-yA)
21    carre_AC PREND_LA_VALEUR (xC-xA)*(xC-xA) + (yC-yA)*(yC-yA)
22    SI (carre_AB == carre_AC) ALORS
23      DEBUT_SI
```

```
24      AFFICHER "Le triangle est isocèle en A"
25      FIN_SI
26      SINON
27        DEBUT_SINON
28        AFFICHER "Le triangle n'est pas isocèle en A"
29        FIN_SINON
30  FIN_ALGORITHME
```

Extension de l'algorithme

6. Modifier l'algorithme précédent afin qu'il détermine, à partir des coordonnées de trois points A, B et C, si le triangle est isocèle et, si oui, en quel point.

Il est cette fois nécessaire d'utiliser des structures conditionnelles SI-ALORS-SINON imbriquées : on ne tester le cas « isocèle en B » que lorsque le triangle n'est pas isocèle en A et le cas « isocèle en C » que lorsque le triangle n'est ni isocèle en A ni isocèle en B…

On obtient alors :

```
TriangleIsocele  -  29.01.2015
****************************************
Cet algorithme détermine si un triangle est isocèle.
****************************************
1   VARIABLES
2     xA EST_DU_TYPE NOMBRE
3     yA EST_DU_TYPE NOMBRE
4     xB EST_DU_TYPE NOMBRE
5     yB EST_DU_TYPE NOMBRE
6     xC EST_DU_TYPE NOMBRE
7     yC EST_DU_TYPE NOMBRE
8     carre_AB EST_DU_TYPE NOMBRE
9     carre_BC EST_DU_TYPE NOMBRE
10    carre_AC EST_DU_TYPE NOMBRE
11  DEBUT_ALGORITHME
12    AFFICHER "Entrez les coordonnées xA et yA du point A"
13    LIRE xA
14    LIRE yA
15    AFFICHER "Entrez les coordonnées xB et yB du point B"
16    LIRE xB
17    LIRE yB
18    AFFICHER "Entrez les coordonnées xC et yC du point C"
19    LIRE xC
20    LIRE yC
21    carre_AB PREND_LA_VALEUR (xB-xA)*(xB-xA) + (yB-yA)*(yB-yA)
22    carre_BC PREND_LA_VALEUR (xC-xB)*(xC-xB) + (yC-yB)*(yC-yB)
23    carre_AC PREND_LA_VALEUR (xC-xA)*(xC-xA) + (yC-yA)*(yC-yA)
24    SI (carre_AB == carre_AC) ALORS
25      DEBUT_SI
26      AFFICHER "Le triangle est isocèle en A"
27      FIN_SI
28      SINON
29        DEBUT_SINON
30        SI (carre_AB == carre_BC) ALORS
31          DEBUT_SI
32          AFFICHER "Le triangle est isocèle en B"
33          FIN_SI
```

```
34          SINON
35            DEBUT_SINON
36            SI (carre_BC == carre_AC) ALORS
37              DEBUT_SI
38              AFFICHER "Le triangle est isocèle en C"
39              FIN_SI
40              SINON
41                DEBUT_SINON
42                AFFICHER "Le triangle n'est pas isocèle"
43                FIN_SINON
44            FIN_SINON
45        FIN_SINON
46  FIN_ALGORITHME
```