



MODULE INF112

TD 2
2012 – 2013



Plan

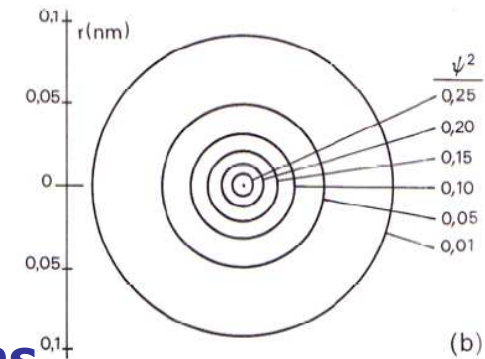
1. Algorithme vs Programme
2. Introduction à l'algorithmique
3. Exercices



1. Algorithme vs programme

Motivations (rappel)

- Pourquoi programmer ?
 - Pour faire des actions répétitives
 - L'ordinateur va plus vite
l'homme s'ennuie et fait des erreurs
- Pourquoi en BIO/SVT/CHBI ?
 - **Création de figures avec des répétitions**
 - **Traitement** et analyse de données
 - **Traitement** et analyse des images



Extraits du livre de Paul Arnaud

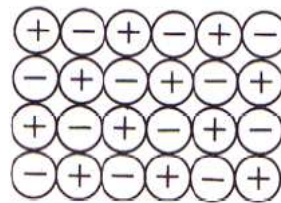
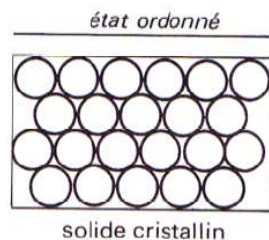
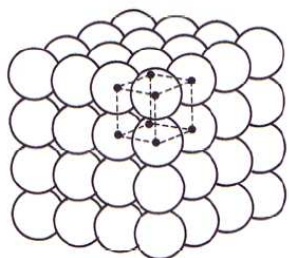


Figure 23.7 – Défauts ponctuels dans un réseau cristallin.

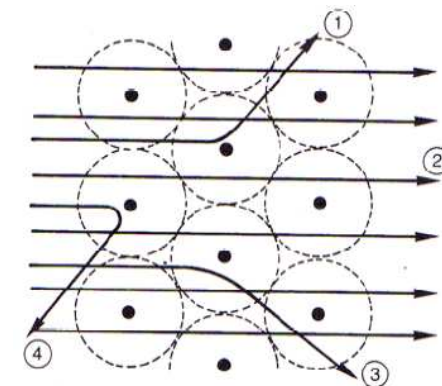


Figure 3.4 – Les trajectoires des particules α

Représentation schématique des états de la matière



1. Algorithme vs programme

Objectifs de INF112

- Il faut apprendre à résoudre les problèmes
 - Soit « a » un problème à résoudre
 - Décomposer « a » en une suite d'actions
- Il faut apprendre à « parler » dans un langage que l'ordinateur comprend
 - Ecrire un programme **format texte**
 - Dans un langage (VBA = Visual Basic Application)
 - Utiliser un outil qui transforme le programme texte en une suite d'octets compréhensible par l'ordinateur (inclus dans Word, Excel et PowerPoint)



1. Algorithmes vs programme

Démarche

Problème complexe



Analyse et décomposition en suite d'opérations élémentaires



ALGORITHME : méthode de résolution d'un problème suivant un enchaînement déterminé de règles

Action « `descriptif_action` » ou encore **Algo** « `nom_action` »

Début

Action 1

Action 2

....

Action n

Fin



PROGRAMME : traduction dans un langage de programmation



1. Algorithmes vs programme

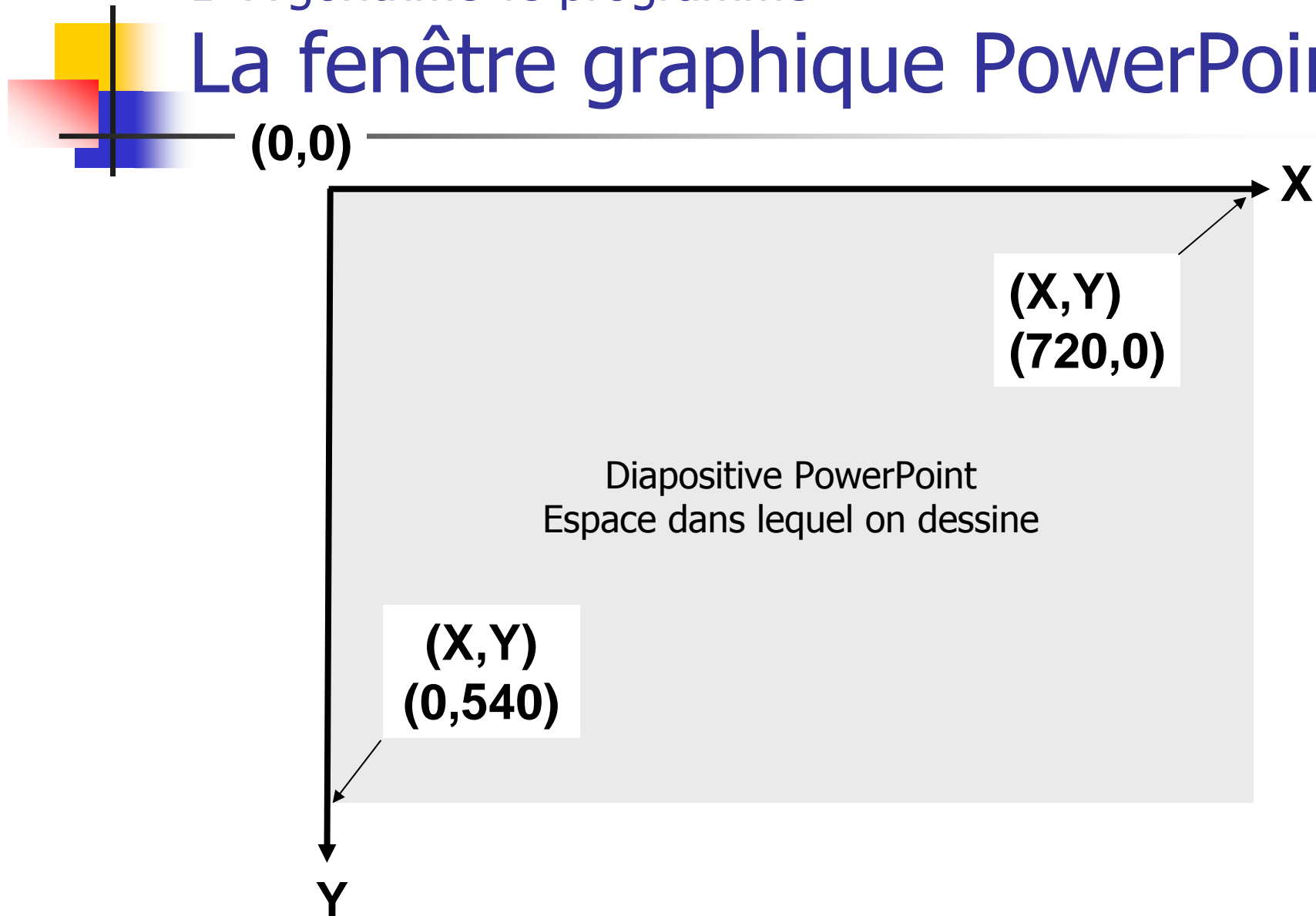
Plan de travail (semaines 2-6)

- Apprendre les principes algorithmiques
 - Itérations, paramétrages
- Application pour des dessins (TP 2-3)
 - PowerPoint
 - Permet de **visualiser le résultat**
 - Auto-évaluation
- Application pour des valeurs numériques (4-..)
 - Excel
 - Traiter d'autres types de problèmes



1. Algorithmes vs programme

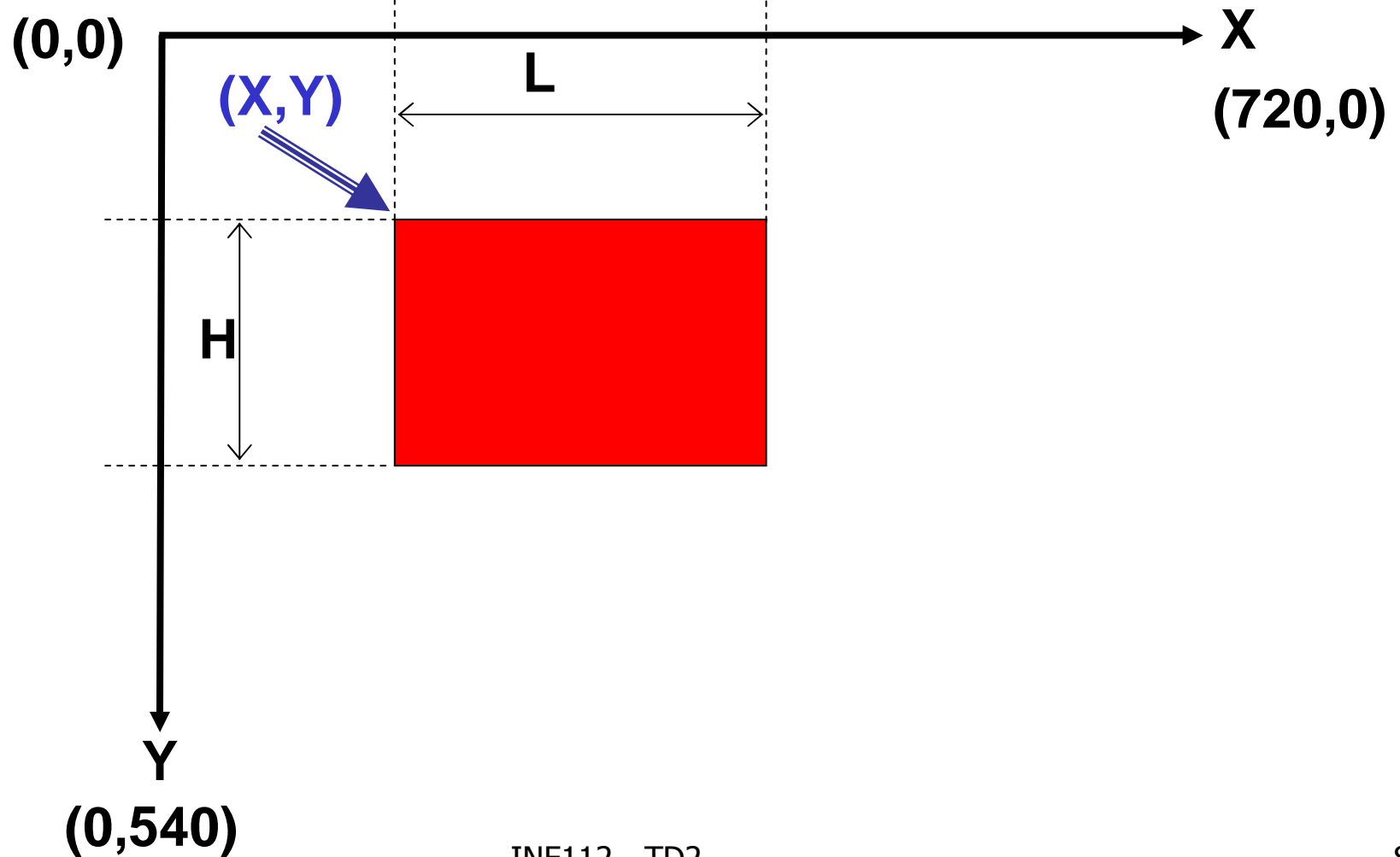
La fenêtre graphique PowerPoint





1. Algorithmes vs programme

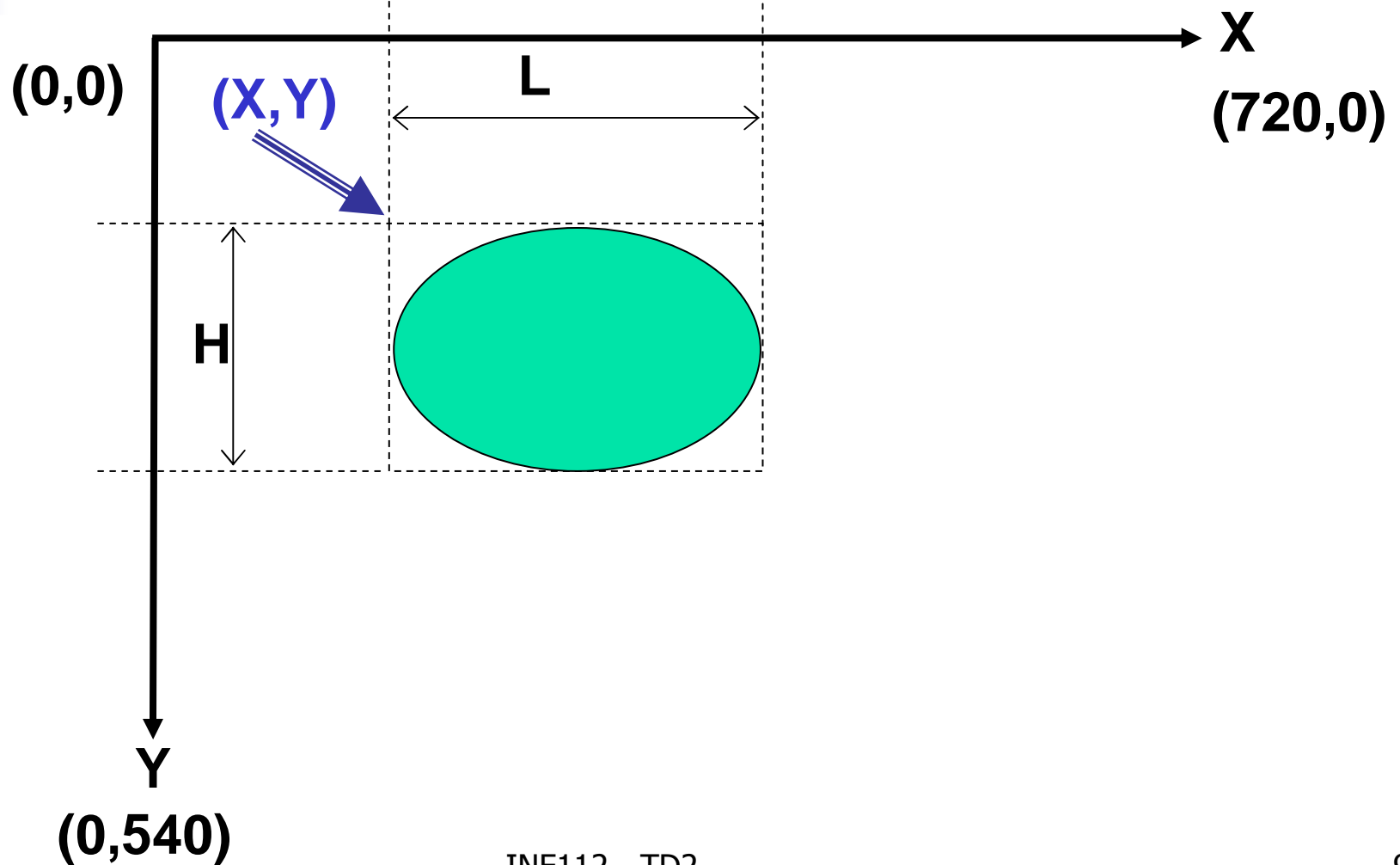
Repérer un objet





1. Algorithmes vs programme

Repérer un objet





1. Algorithmes vs programme

Dessiner un Objet en Algo

Algo UnOvale

{algorithme réalisant le tracé d'un ovale}

Début

Ovale(150, 192, 168, 126)

Fin

X,Y L, H

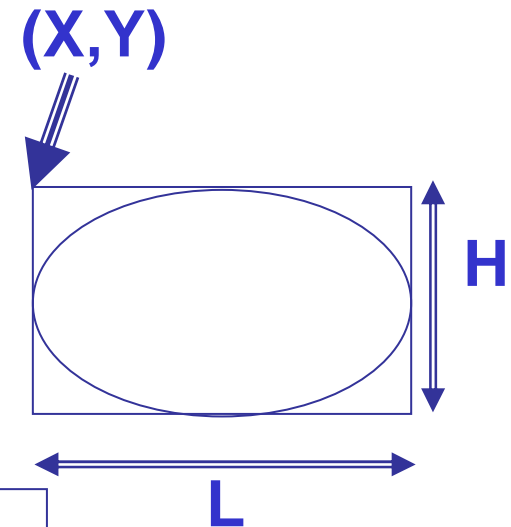
Algo UnRectangle

{algorithme réalisant le tracé d'un rectangle}

Début

Rectangle(150, 192, 168, 126)

Fin





1. Algorithmme vs programme

Dessiner un Objet en VBA

```
Sub UnOvale()
```

```
ActiveWindow.Selection.SlideRange.Shapes.AddShape(msoShapeOval,  
150#, 192#, 168#, 126#).Select
```

```
End Sub
```

```
Sub UnRectangle()
```

```
ActiveWindow.Selection.SlideRange.Shapes.AddShape(msoShapeRectangle,  
150#, 192#, 168#, 126#).Select
```

```
End Sub
```



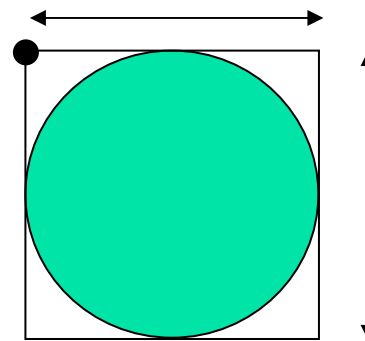
1. Algorithmme vs programme

Dessiner un objet en Algo

(PositionX, PositionY)

Cercle (PositionX, PositionY, Côté)

Exemple : Cercle(252, 216, 174)

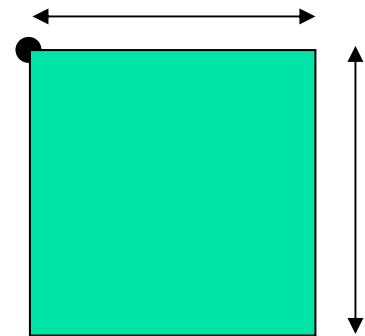


CôtéX = CôtéY

(PositionX, PositionY)

Carré (PositionX, PositionY, côté)

Exemple : Carré(252, 216, 174).



CôtéX = CôtéY



1. Algorithmme vs programme

Dessiner un Objet en VBA

```
Sub UnCercle()
```

```
ActiveWindow.Selection.SlideRange.Shapes.AddShape(msoShapeOval,  
252#, 216#, 174#, 174#).Select
```

```
End Sub
```

```
Sub UnCarré()
```

```
ActiveWindow.Selection.SlideRange.Shapes.AddShape(msoShapeRectangle,  
252#, 216#, 174#, 174#).Select
```

```
End Sub
```



1. Algorithme vs programme

Notion de programme

Un programme est constitué d'une suite d'actions

Algorithmique	Programmation VBA
Algo « nom_action » Début {commentaire} Action 1 Action 2 Action n Fin	Sub nom_action 'commentaire : description de l'action Action 1 Action 2 Action n End Sub



1. Algorithmes vs programme

Notion de programme

Algo "OvaleRouge"

Début

Ovale(276, 120, 132, 312)

Couleur(rouge)

Fin

Sub OvalRouge()

**ActiveWindow.Selection.SlideRange.Shapes.AddShape
(msoShapeOval, 276#, 120#, 132#, 312#).Select**

With ActiveWindow.Selection.ShapeRange

.Fill.Visible = msoTrue

.Fill.ForeColor.RGB = RGB(255, 0, 0)

End With

End Sub



1. Algorithmes vs programme

Notion de programme (OpenOffice)

```
sub UnRond
  Dim Doc As Object
  Dim Page As Object
  Dim Circle As Object
  Dim Point As New com.sun.star.awt.Point
  Dim Size As New com.sun.star.awt.Size

  Doc = ThisComponent
  Page = Doc.DrawPages(0)
  Size.Width = 1320
  Size.Height = 3120
  Point.x = 2760
  Point.Y = 1200
  Circle = Doc.CreateInstance("com.sun.star.drawing.EllipseShape")
  Circle.Size = Size
  Circle.Position = Point
  Circle.FillColor = RGB(255,0,0)
  Page.add(Circle)
end sub
```

Algo "OvaleRouge"

Début

Ovale(276, 120, 132, 312)

Couleur(rouge)

Fin



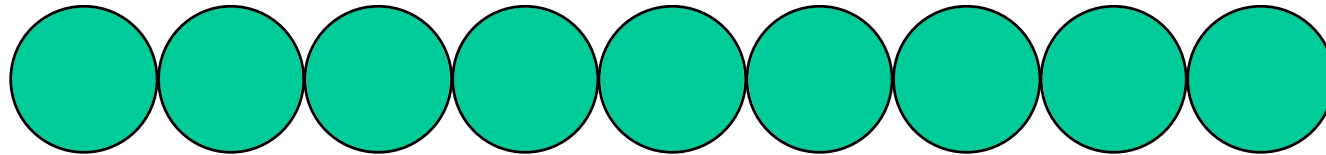
Plan

1. Algorithme vs Programme
2. Introduction à l'algorithmique
3. Exercices



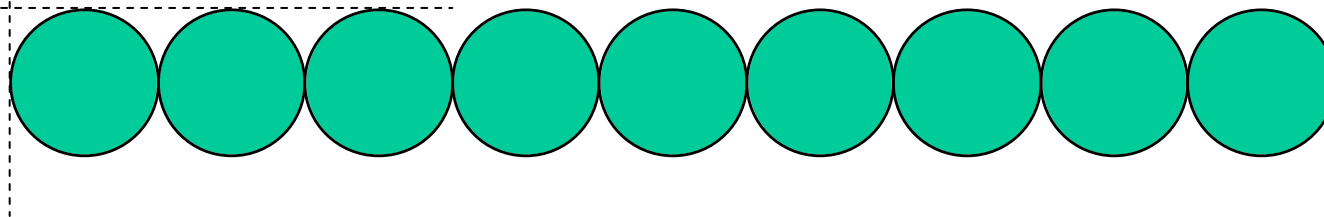
2. Introduction à l'algorithmique

« Créer une chaîne »



Problème : dessiner une chaîne de cercles de même taille

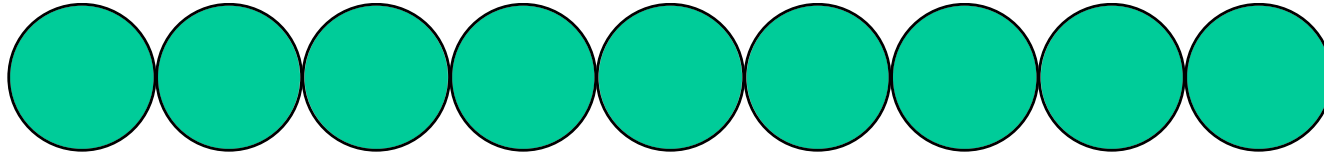
(X,Y)





2. Introduction à l'algorithmique

« Créer une chaîne »



Action “ créer une chaîne ”

Début

Cercle (x, y, c)

Cercle (x+c, y, c)

Cercle (x+2*c, y, c)

Cercle (x+3*c, y, c)

Cercle (x+4*c, y, c)

....

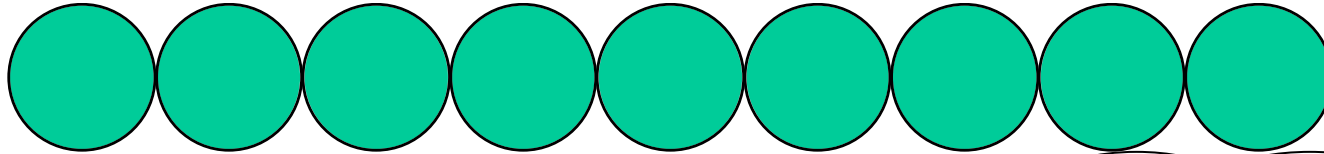
Et ainsi de suite jusqu'au 9^{ième} cercle

Fin



2. Introduction à l'algorithmique

« Créer une chaîne »



Action “ créer une chaîne ”

Début

Cercle (x, y, c)

Cercle (x+c, y, c)

Cercle (x+2*c, y, c)

Cercle (x+3*c, y, c)

Cercle (x+4*c, y, c)

....

Et ainsi de suite jusqu'au 9^{ième} cercle

Fin

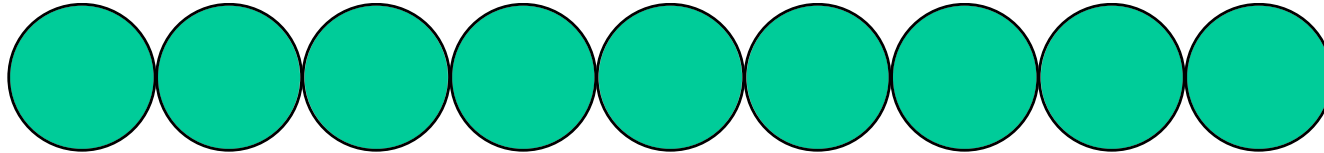
Peut-on éviter d'écrire 9 fois
l'instruction « cercle » ?





2. Introduction à l'algorithmique

Notion de répétition



Action " créer une chaîne"

Début

Cercle (x, y, c)

Cercle (x+c, y, c)

Cercle (x+2*c, y, c)

Cercle (x+3*c, y, c)

Cercle (x+4*c, y, c)

....

Fin



Action " créer une chaîne"

Début

Répéter 9 fois :

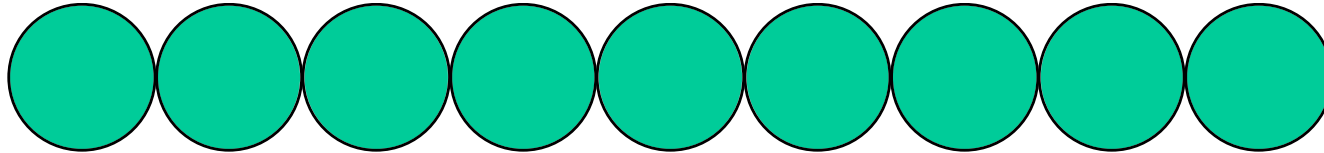
Cercle (_____, y, c)

Fin



2. Introduction à l'algorithmique

Notion de répétition



Action " créer une chaîne"

Début

Cercle (x, y, c)
Cercle (x+c, y, c)
Cercle (x+2*c, y, c)
Cercle (x+3*c, y, c)
Cercle (x+4*c, y, c)
....

Fin



Introduction d'une variable Pour « compter »

Action " créer une chaîne"

Début

t ← 0

Répéter 9 fois :

 Cercle (x+t*c, y, c)

 t ← t+1

Fin répéter

Fin



2. Introduction à l'algorithmique

Notion de répétition

Il existe deux façons de répéter une action :

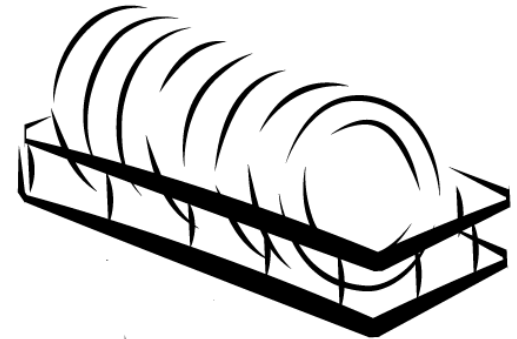
- Tant qu'une condition est vraie, faire...
- Pour $i = \text{val_initiale}$ jusqu'à val_finale , faire...



2. Introduction à l'algorithmique

Exemple de la vie courante

- Soit une pile de vaisselle à laver
- Tant que la pile est « non vide »
 - Prendre l'assiette au dessus de la pile
 - Laver l'assiette, la rincer, la poser ailleurs
- Il y a 8 assiettes sur l'égouttoir
- Pour $i=1$ jusqu'à 8
 - Prendre l'assiette au dessus de la pile
 - Laver l'assiette, la rincer, la poser ailleurs





2. Introduction à l'algorithmique

Chaîne avec un « tant que »

Algo « chaineTantQue »

Début

x, c, t : entier {Déclaration}

c \leftarrow 50

x \leftarrow 100

t \leftarrow 0

tant que $t < 9$ faire

 Cercle(**$x + t * c$** , 200, **c**)

t \leftarrow **$t + 1$**

Fin tant que

Fin



2. Introduction à l'algorithmique

Chaîne avec un « pour »

Algo « chainePour »

Début

x, c, t : entier {Déclaration}

c ← 50

x ← 100

Pour $t=0$ jusqu'à 8 **faire**

 Cercle(**$x + t*c$** , 200, **c**)

Fin Pour

Fin



2. Introduction à l'algorithmique

Vocabulaire : variable

Algo « chaineTantQue »

Début

x, c, t : entier
{Déclaration}

$c \leftarrow 50$

$x \leftarrow 100$

$t \leftarrow 0$

tant que $t < 9$ faire

Cercle($x + t * c$, 200, c)

$t \leftarrow t + 1$

Fin tant que

Fin

Variable :

Nom générique d'une donnée

Type :

Manière dont est codée
l'information contenue dans la
variable

(entier, réel, caractères...)

Valeur :

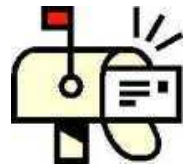
Contenu de la variable



2. Introduction à l'algorithmique

Notion de variable

- Une variable dans un programme, c'est un endroit pour y ranger des données
- Analogie de la vie courante : boîte à lettres



- Déclarer une variable, c'est donner :
 - un nom (pour pouvoir manipuler les données)
 - une dimension (parce que la mémoire n'est pas infinie)
- Déclarer une variable, c'est réserver de la place dans la mémoire de l'ordinateur



2. Introduction à l'algorithmique

Déclaration de variable en VBA

Algorithmique	Programmation VBA
Algo « nom_action » Début t : entier {Déclaration d'un entier} x : réel {Déclaration d'un réel} ch : chaîne {Déclaration d'une chaîne de caractères} Action1 Action 2	Sub nom_action Dim t As Integer <i>'Déclaration d'un entier</i> Dim x As real <i>'Déclaration d'un réel</i> Dim ch As string <i>'Déclaration</i> <i>d'une chaîne de caractères</i> Action 1 Action 2
Fin	End Sub



2. Introduction à l'algorithmique

Affectation de variable

$u \leftarrow 0$ { u prend pour valeur zéro}

$R \leftarrow 3.14$

$ch \leftarrow \text{"bonjour"}$

$t \leftarrow u$

à gauche :

Le nom de la variable
(contenant à remplir)

à droite :

le nouveau contenu
valeur, variable ou expression



2. Introduction à l'algorithmique

Affectation de variable

- **Initialisation :**
première affectation
d'une variable

$t \leftarrow 0$

$t \leftarrow 12$

$u \leftarrow 50$

$u \leftarrow 20$

$t \leftarrow u+4$

$ch \leftarrow \text{"bonjour"}$

$ch \leftarrow \text{"au revoir"}$

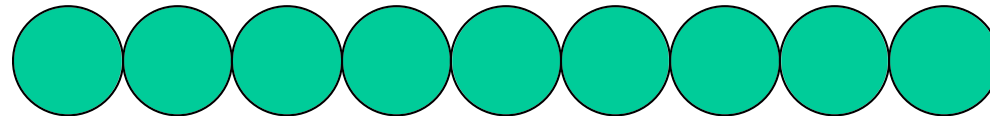
- **Incrémentation :**
augmentation de
la valeur d'une variable

$t \leftarrow t+1$



2. Introduction à l'algorithmique

Retour sur les itérations

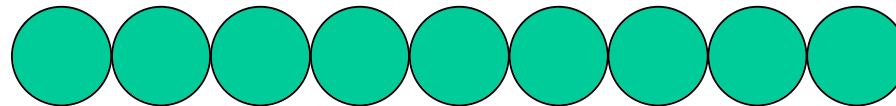


Algorithmique	Programmation VBA
Algo « chaineTantQue »	Sub chaineWhile()
Début	
t : entier {Déclaration}	Dim t As Integer
$t \leftarrow 0$ {initialisation}	t = 0
Tant Que $t < 9$ faire {condition d'arrêt}	Do While t < 9
Cercle(100 + t * 50, 200, 50)	ActiveWindow.Selection.SlideRange.Shapes.AddShape(msoShapeOval, 100# + t * 50#, 200#, 50#, 50#).Select
$t \leftarrow t + 1$ {Incrémentation}	t = t + 1
Fin Tant Que	Loop
Fin	End Sub



2. Introduction à l'algorithmique

Retour sur les itérations



Algorithmique	Programmation VBA
<p>Algo « chainePour »</p> <p>Début</p> <p>t : entier {Déclaration}</p> <p>Pour $t=0$ jusqu'à 8 faire</p> <p>{condition d'arrêt initialisation, incrémentation inclus dans le Pour}</p> <p>Cercle($100 + t * 50$, 200, 50)</p> <p>Fin Pour</p> <p>Fin</p>	<pre>Sub chaineFor() Dim t As Integer For t = 0 To 8 ActiveWindow.Selection.SlideRange.Shapes.AddShape(msoShapeOval, 100# + t * 50#, 200#, 50#, 50#).Select Next End Sub</pre> <p style="text-align: right;">22</p>



2. Introduction à l'algorithmique

Retour sur les variables

Algo « chaineTantQue »

Début

x, c, t : entier {Déclaration}

c \leftarrow 50

x \leftarrow 100

t \leftarrow 0 {initialisation}

Tant Que **$t < 9$** faire

Cercle(**x** , 200, **c**)

x \leftarrow **$x + c$**

t \leftarrow **$t + 1$**
{Incrémentation}

Fin Tant Que

Fin

Algo « chainePour »

Début

x, c, t : entier {Déclaration}

c \leftarrow 50

x \leftarrow 100

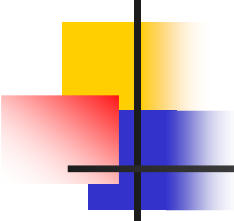
Pour **$t=0$** jusqu'à **8** faire

Cercle(**x** , 200, **c**)

x \leftarrow **$x + c$**

Fin Pour

Fin

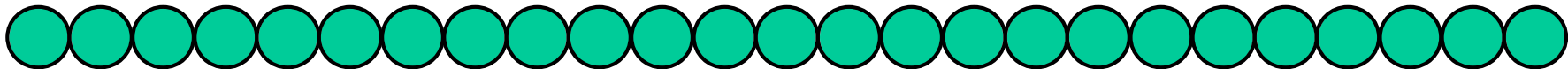


Plan

1. Algorithme vs Programme
2. Introduction à l'algorithmique
3. **Exercices**



Exercice 1 :



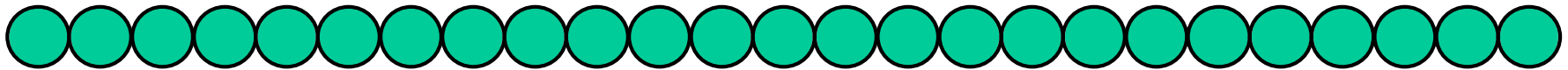
Chaîne de 25 cercles de 20 pixels de diamètre

Utiliser une boucle « Pour »

(dans la chaîne précédente le diamètres des cercles était de 50 pixels)



Exercice 2 :



Même chose avec la boucle « TantQue »



Exercice 3 :

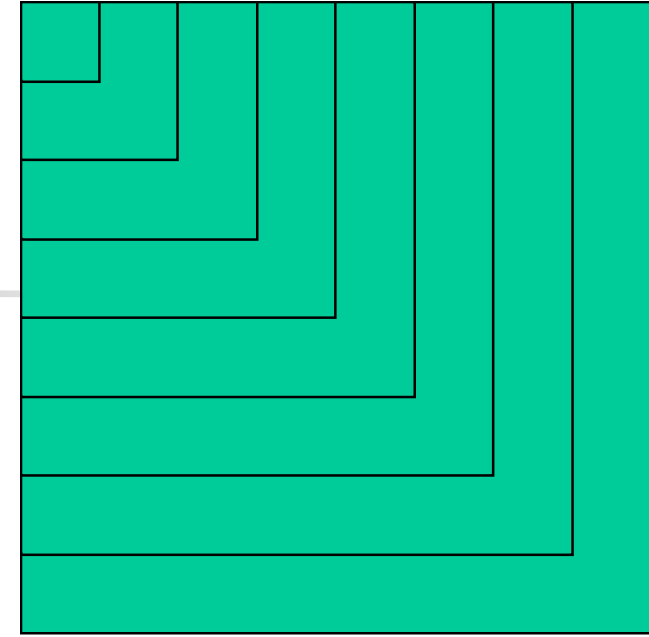


Frise de 12 carrés de 30 pixels de côté espacés de 30 pixels



Exercice 4 : Carrés emboîtés

La taille du plus petit carré correspond à l'espacement entre les carrés.

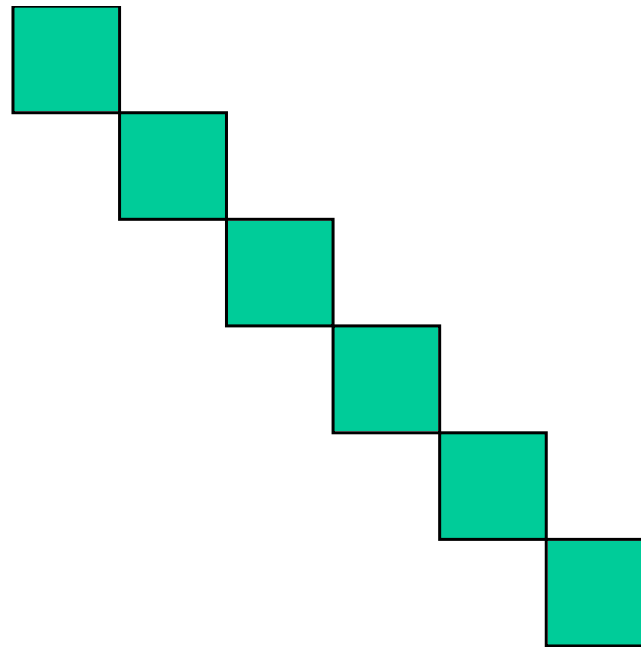


La valeur prise pour le dessin est de 30 pixels, mais cette valeur doit être facilement modifiable.

Si l'on veut que le plus grand carré ne cache pas tous les autres, il faut tracer les carrés du plus grand au plus petit.



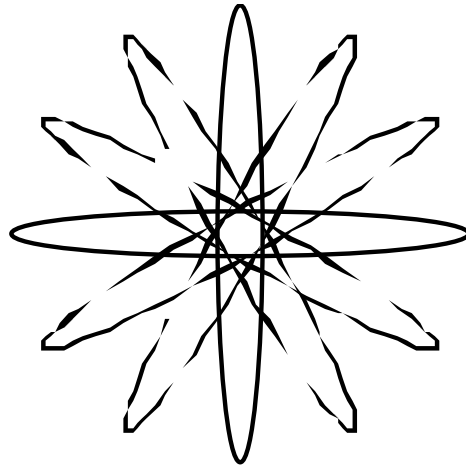
Exercice 5 :



Escalier de 6 carrés de 40 pixels de côté.



Exercice 6 :



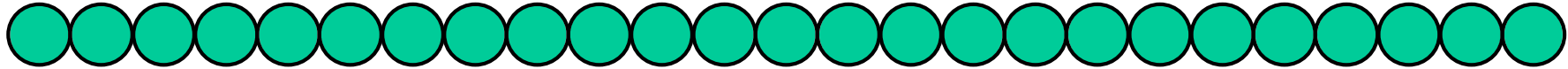
Rosace

Chaque élément fait 10 pixels dans sa plus petite dimension.



Corrigé des exercices

Ne pas imprimer sur le polycopié
étudiant.



Exercice 1 :

Algo “ chaine25”

Début

x, y : entier {position de la chaîne}
c : entier {diamètre d'un cercle}
n : entier {nombre de cercles}
t : entier {compteur d'itérations}

x ← 100

y ← 200

c ← 20

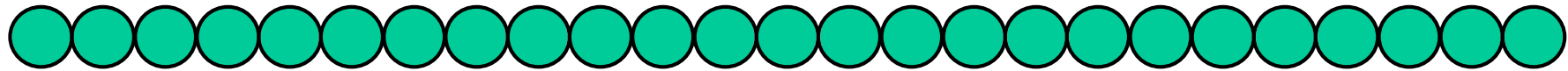
n ← 25

Pour *t=0* jusqu'à *n-1* **faire**

 Cercle(*x + t * c, y, c*)

Fin Pour

Fin



Exercice 2 :

Algo “ chaine25”

Début

x*, *y : entier {position de la chaîne}
c : entier {diamètre d'un cercle}
n : entier {nombre de cercles}
t : entier {compteur d'itérations}

***x* ← 100**

***y* ← 200**

***c* ← 20**

***n* ← 25**

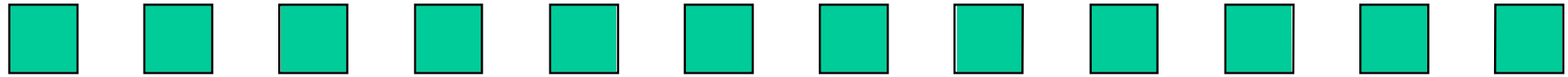
***t* ← 0**

Tant que *t* < *n* faire

Cercle(*x* + *t* * *c*, *y*, *c*)

***t* ← *t* + 1**

Fin Tant Que



Exercice 3 :

Algo “ Frise12Carres”

Début

x*, *y : entier {position de la chaîne}
c : entier {côté d'un carré}
n : entier {nombre de carrés}
t : entier {compteur d'itérations}

x ← 10

y ← 300

c ← 30

n ← 12

Pour *t*=0 jusqu'à *n*-1 faire

 Carré($x + 2 * t * c$, *y*, *c*)

Fin Pour

Fin



Exercice 4 :

Algo “ CarresEmboites”

Début

x, y, n, e, c : entier

t : entier {compteur d'itérations}

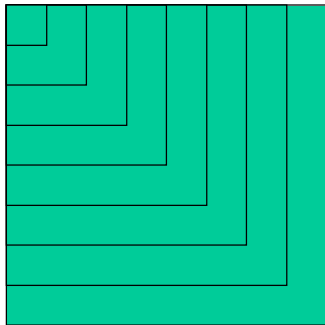
$x \leftarrow 50$ {position des carrés}

$y \leftarrow 50$

$n \leftarrow 8$ {nombre de carrés}

$e \leftarrow 30$ {Espace entre les carrés}

$c \leftarrow e * n$ {Côté du grand carré}



Pour $t=0$ jusqu'à $n-1$ faire

Carré(x, y, c)

$c = c - e$

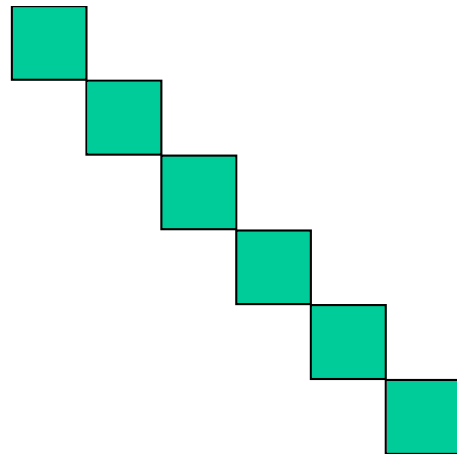
Fin Pour



Exercice 5 :

Algo “ CarresEscalier”

Début



x, y, n, e, c : entier

t : entier {compteur d'itérations}

$x \leftarrow 50$ {position des carrés}

$y \leftarrow 50$

$n \leftarrow 6$ {nombre de carrés }

$c \leftarrow 30$ {côté des carrés}

Pour $t=0$ jusqu'à $n-1$ faire

Carré(x, y, c)

$x = x + c$

$y = y + c$

Fin Pour

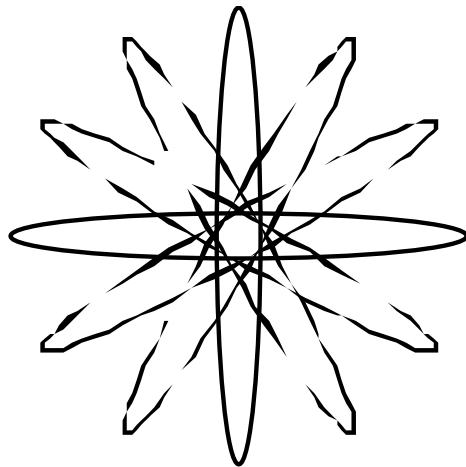
Fin



Exercice 6 :

Algo “Rosace”

Début



x, y, c, n, a : entier

t : entier {compteur d'itérations}

$x \leftarrow 100$

$y \leftarrow 100$ {position des ovals }

$n \leftarrow 6$ {nombre d'ovals}

$c \leftarrow 10$ {petit diamètre de l'ovale}

$a \leftarrow 180 / n$ {Angle de rotation des ovals}

Pour $t= 1$ jusqu'à n faire

Ovale($x, y, c, 10*c$)

Couleur (transparente)

Rotation ($a * t$)

Fin Pour

Fin