Université Bachir El Ibrahimi de Bordj Bou Arreridj
Faculté des Mathématiques et Informatique
Département d'informatique

Examen n°01 d'Algorithmique et Structures de données avancées

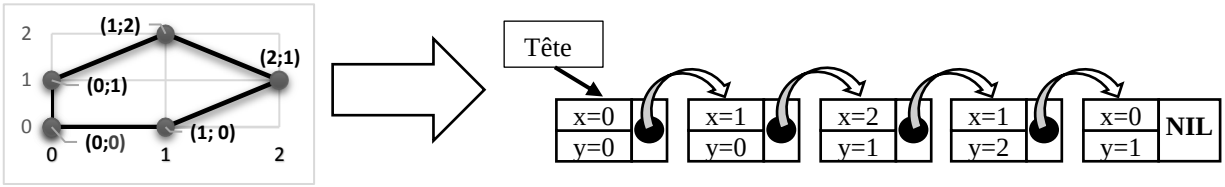
2ème année licence

Durée : 01 h 30 min

La propreté de la feuille de réponses est exigée et notée (1 point)

L'objectif de cet examen consiste à manipuler un polygone dans un plan. Un polygone est représenté par un ensemble de points stockés dans une liste linéaire chainée (LLC). La figure suivante illustre le principe de construction d'une liste à partir d'un polygone.

Exemple :



Pour ce faire, nous divisons le travail en quatre parties :

Partie 1 : Remplir la liste par des points d'un polygone (5 points)

Chaque maillon de LLC contient les cordonnées cartésiennes (x, y) d'un point dans un plan. Par conséquent, Nous utilisons deux structures :

1. Point : structure qui représente un point. Cette structure contient deux champs : **x** et **y**.
2. Maillon : structure qui forme le maillon de la liste, contient deux champs :
   - **Valeur :** un point du polygone.
   - **Suivant**: un pointeur sur le maillon suivant.

```
Type Point=Structure
     x: Réel
     y: Réel
Fin
```

```
Type Maillon=Structure
     Valeur: Point
     Suivant: Pointeur(Maillon)
Fin
```

Pour accéder/modifier les cordonnées d'un point stocké dans un maillon, nous utilisons les opérations **Valeur** et **Aff_Val** comme suit :

```
P : Pointeur(Maillon)
V : Point
/*Accès à un point*/
V=Valeur(P)
x= V.x
y= V.y
```

```
P : Pointeur(Maillon)
V : Point
/*Modifier un point*/
V.x=2
V.y=3
Aff_val(P,V)
```

1. Ecrire, en langage algorithmique, la procédure qui permet de créer une liste qui représente un polygone à partir des cordonnées (x,y) des points fournis par l'utilisateur. **(3 points)**

```
Procédure CréerListePolygone(Var Tête : Pointeur(Maillon), N :entier)
Var P,Q :Pointeur(Maillon)
     I :Entier                          0. 25
     V :Point
Début
     Tête ← Nil                         0.25
     Pour i allant de 1 à N faire
     Dpour
          Lire(V.x)                     0.5
          Lire(V.y)

          Allouer(P)                    0.75
          Aff_val(P,V)
          Aff_adr(P,Nil)

          Si(Tête = Nil) Alors
               Tête ← P                 0.5
          Sinon
               Aff_adr(P,Q)             0.5
          Q ← P                         0.25
     Fpour
Fin
```

2. Ecrire, en langage algorithmique, la procédure qui affiche les cordonnées des points d'un polygone donné. (**2 points**)

```
Procédure AfficherPolygone(Tête : Pointur(Maillon))
Var P : Pointeur(Maillon)                    0.25
    V : Point
Début
     P ← Tête  0.25
     Tant que (P <> Nil) Faire  0.5
     Dtq
          V ← Valeur(P)   0.25
          Ecrire(V.x,V.y)  0.5
          P ← Suivant(P)  0.25
     Ftq
Fin
```

Partie 2 : Périmètre du polygone (5 points)

Dans cette partie, nous allons déterminer le périmètre d'un polygone stocké dans une liste. Pour ce faire, nous avons besoin de deux fonctions (Distance, Périmètre):

1. Ecrire, en langage algorithmique, une fonction qui calcule la distance entre deux points $\boldsymbol{P_1 = (x_1, y_1)}$ et $\boldsymbol{P_2 = (x_2, y_2)}$. (**2 points**)

```
Fonction Distance (P1:Point, P2:Point):Réel
Var Difx,Dify :réel;  0.25
Début
     Difx=(P1.x-P2.x)*(P1.x-P2.x)  0.5
     Dify=(P1.y-P2.y)*(P1.y-P2.y)  0.5
     Distance = Racine(Difx+Dify)
Fin          0.25  0.25     0.25
```

➢ ***Indication :*** *La distance euclidienne entre deux points $P_1 = (x_1, y_1)$ et $P_2 = (x_2, y_2)$ est donnée par la formule suivante : $Distance(P_1, P_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Vous pouvez appeler la fonction standard* ***Racine (x)*** *dans la fonction distance.*

2. Ecrire, en langage algorithmique, une fonction qui calcule le périmètre d'un polygone en se basant sur la fonction **Distance**. (**3 points**)

```
Fonction Périmètre (Tete : Pointeur(Maillon)):Réel
Var P :Pointeur(Maillon)
    D,S :réel
Début
     P ← Tête  0.25
     S ← 0     0.25
     Tant que (Suivant(P) <> Nil) Faire  0.25
     Dtq
          D ← Distance(Valeur(P),Valeur(Suivant(P)))  0.75
          S ← S + D      0.25
          P ← Suivant(P) 0.25
     Ftq
     S ← S + Distance (Valeur(P), Valeur(Tête))  0.75
     Périmètre ← S  0.25
Fin
```

Indication : *Périmètre du polygone (**Figure 1**)* $= D1 + D2 + D3 + D4 + D5$

$$= 1 + \sqrt{2} + \sqrt{2} + \sqrt{2} + 1 \approx 6.242$$

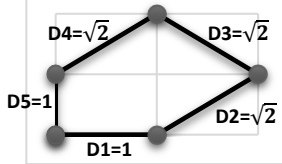


Figure 1

Partie 3 : Déplacement du polygone (4 points)

Dans cette partie, nous voulons déplacer le polygone par un vecteur de déplacement $D(d_x, d_y)$. Pour ce faire, nous divisons le travail en deux modules:

1. Ecrire, en langage algorithmique, la fonction de déplacement d'un point ***P*** par le vecteur de déplacement $D(d_x, d_y)$. Cette fonction retourne le point après le déplacement. (**2 points**)

```
Fonction DéplacerPoint(P:Point,dx:Réel,dy:Réel):Point
Var V :Point                                          0.25
Début
     V.x ← P.x+dx                                     0.75
     V.y ← P.y+dy                                     0.75
     DépacerPoint ← V                                 0.25
Fin
```

- ***Indication :*** *Le déplacement d'un point est effectué comme suit :*
  $\begin{cases} P.x = P.x + d_x \\ P.y = P.y + d_y \end{cases}$

2. Ecrire, en langage algorithmique, la procédure qui déplace un polygone par le vecteur de déplacement $D(d_x, d_y)$. (**2 points**)

```
Procédure DéplacerPolygone(Tete :Pointeur(maillon),dx:Réel,dy:Réel)
Var P :Pointeur(Maillon)                              0.25
    V :Point
Début
     P ← Tête                                         0.25
     Tant que (P <> Nil) Faire                        0.25
     Dtq
          V ← DéplacerPoint(P,dx,dy)                  0.5
          Aff_val(P,V)                                0.5
          P ← Suivant(P)                              0.25
     Ftq
Fin
```

- ***Indication :*** *Le déplacement d'un polygone implique le déplacement de tous ces points par le même vecteur de déplacement (**Figure 2**).*

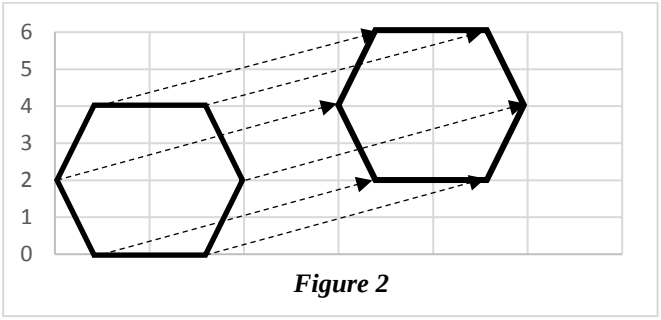


Figure 2

Partie 4 : Programme principal (5 points)

En utilisant des appels aux modules des parties précédentes, écrire le programme principal (Sans la partie déclaration de variables, fonctions) qui permet de :

1. Créer une liste linéaire chainée d'un polygone fourni par l'utilisateur.
2. Afficher les cordonnés des points du polygone créé.
3. Calculer et afficher le périmètre du polygone créé.
4. Déplacer le polygone par un vecteur de déplacement $\mathbf{D} = (\mathbf{3}, \mathbf{2})$.
5. Afficher les cordonnés du polygone après le déplacement précèdent.

```
Début
     Ecrire('Donner le nombre de points')
     Lire(N)                                0.5
     CréerListePolygone(Tête,N)             0.5
     AfficherPolygone(Tête)                 1
     Ecrire(Périmètre(Tete))                1
     DéplacerPolygone(Tête,3,2)             1
     AfficherPolygone(Tête)                 1
Fin
```

Bonne chance