



PROGRAMMATION OBJET Langage C++

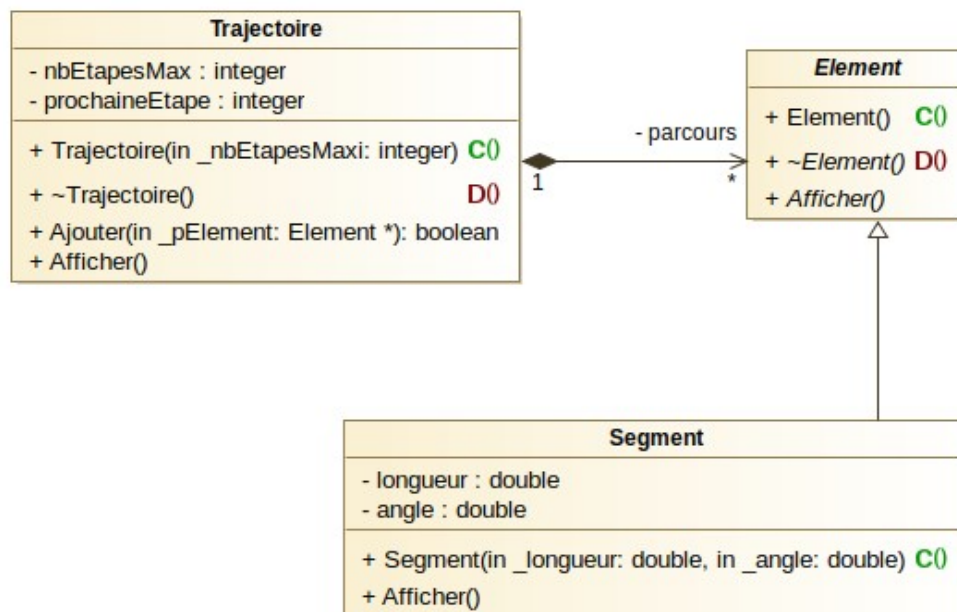
~ TP Polymorphisme ~



L'objectif du programme est de permettre la création, la mémorisation et la modification d'une trajectoire d'un robot mobile au sein d'une entreprise. Cette trajectoire sera constituée d'une liste d'éléments de différents types (segments, arc de cercle...).

Vous déposerez votre projet dans votre Github, en réalisant des sauvegardes intermédiaires à chaque étape.

1- Création des classes définissant la structure du projet



Dans un premier temps, le programme possède 3 classes comme le montre le diagramme de classe ci-après :

Description :

La classe nommée **Element** définit une classe de base pour les autres classes de la collection qui composeront une trajectoire. Elle possède dans un premier temps : un constructeur, un destructeur virtuel et une méthode virtuelle pure **Afficher()**.

Une classe **Segment** dérivée de **Element**. Un **Segment** est caractérisé par sa **longueur** exprimée en unité de longueur et son angle par rapport à l'horizontale exprimé en radian.

Une classe **Trajectoire** représente le conteneur des différents éléments composant la trajectoire. Elle dispose entre autres d'un tableau dynamique de pointeurs pour stocker les adresses de ces éléments.

1. Comment se nomme la relation entre la classe **Trajectoire** et la classe **Element**, sous quelle forme cette relation doit être implémentée en C++ ?
2. Comment peut-on qualifier la classe **Element** ? Justifier votre réponse.
3. Dans un projet de type application C++ sans utiliser la librairie **Qt**, codez ces trois classes.
4. Réalisez un premier programme principal permettant, à titre d'exemple, d'obtenir l'affichage suivant. Vous utiliserez les flux et les manipulateurs pour la mise en forme.

Réalisé dans la
classe Trajectoire

Réalisé dans la
classe Segment

```

Fichier  Editer  Affichage  Rechercher  Terminal  Aide
Trajectoire :
SEGMENT L = 9      A = 0
SEGMENT L = 5      A = 0.927295

Appuyez sur <ENTRÉE> pour fermer cette fenêtre...

```

2- Complément apporté aux trois classes précédentes

5. Complétez la classe **Element** en lui ajoutant un attribut protégé nommé **numero** initialisé à 0 par le constructeur. Il sera incrémenté à chaque ajout d'élément dans la trajectoire. Pour manipuler cet attribut, on prévoit les fonctions « Getter » et « Setter » associées. Un deuxième attribut protégé sera également ajouté, il portera le nom de **vitesse**. Cet attribut sera initialisé par le constructeur avec une valeur par défaut égale à 1.
6. Complétez le constructeur de la classe **Segment** afin qu'il tienne compte des ajouts précédents. Modifiez la fonction d'affichage de cette classe pour obtenir la figure suivante :
7. Complétez le code de la méthode **Ajouter** de la classe **Trajectoire** afin de gérer l'incrémentation des éléments de la trajectoire.
8. Complétez le code de la fonction main pour obtenir l'affichage suivant

Affichage du
numéro dans la
trajectoire

```

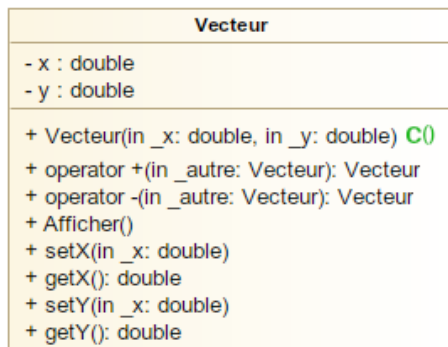
Fichier  Editer  Affichage  Rechercher  Terminal  Aide
Trajectoire :
(1) SEGMENT L = 9      A = 0      V = 3
(2) SEGMENT L = 5      A = 0.927295  V = 1

Appuyez sur <ENTRÉE> pour fermer cette fenêtre...

```

Affichage de la
vitesse

9. Réalisez la classe **Vecteur** comme représentée sur la figure ci-après :



- ← Attributs
- ← Constructeur
- ← Surcharge des opérateurs
- ← Affichage (x,y)
- ← Getter et Setter

10. Ajoutez les fonctions virtuelles pures suivantes dans la classe **Element**, elles serviront de modèle pour les classes dérivées :

| | |
|--|--|
| double ObtenirLongueur() | Retourne la longueur de l'élément. |
| double ObtenirDuree() | Retourne le temps mis pour réaliser le mouvement de l'élément. |
| Vecteur ObtenirVecteurArrivee() | Retourne le vecteur résultant de l'élément. |

11. Modifiez le constructeur de la classe **Segment** pour tenir compte de l'initialisation des nouveaux attributs de la classe **Element**, et ajoutez la surcharge des nouvelles méthodes.

Pour rappel :

| | |
|--|--|
| le calcul du vecteur d'arrivée d'un segment est donné par les formules suivantes | |
| $x = \text{longueur} * \cos(\text{angle})$ | $y = \text{longueur} * \sin(\text{angle})$ |
| le temps pour réaliser le mouvement est donné par la formule | |
| $\text{durée} = \text{longueur} / \text{vitesse}$ | |

12. Complétez l'affichage de la trajectoire pour faire apparaître les nouveaux éléments comme le montre la figure ci-après, on suppose le point de départ en (0,0) :

```

Fichier  Editer  Affichage  Rechercher  Terminal  Aide
Trajectoire :

(1) SEGMENT L = 9      A = 0      V = 3
Vecteur en (9,0)
(2) SEGMENT L = 5      A = 0.927295  V = 1
Vecteur en (12,4)

Durée totale du parcours = 8
Longueur totale du parcours = 14

Appuyez sur <ENTRÉE> pour fermer cette fenêtre...

```

13. Complétez pour la classe **Trajectoire** avec un nouvel attribut **depart** de type **Vecteur**, il contiendra les coordonnées du point de départ et sera initialisé par le constructeur.

14. Complétez à nouveau la fonction d'affichage et le programme principal pour faire apparaître par exemple une trajectoire

```

Fichier  Editer  Affichage  Rechercher  Terminal  Aide
Trajectoire :
(1) SEGMENT L = 9      A = 0      V = 3
Vecteur en (14,10)
(2) SEGMENT L = 5      A = 0.927295  V = 1
Vecteur en (17,14)
(3) SEGMENT L = 6      A = 1.5708    V = 2
Vecteur en (17,20)

Durée totale du parcours = 11
Longueur totale du parcours = 20

Vecteur de départ = (5,10)
Vecteur d'arrivée = (17,20)
Appuyez sur <ENTRÉE> pour fermer cette fenêtre...

```

3- Création de nouveaux éléments

15. Ajoutez une classe Pause, elle représente un arrêt du mobile dans la trajectoire et possède une donnée membre **tempsAttente**, initialisée par le constructeur de cette classe.

Affichage pour la classe Pause avec un temps de pause égale à 4.

```

Fichier  Editer  Affichage  Rechercher  Terminal  Aide
(1) SEGMENT L = 9      A = 0      V = 3
Vecteur en (14,10)
(2) SEGMENT L = 5      A = 0.927295  V = 1
Vecteur en (17,14)
(3) SEGMENT L = 6      A = 1.5708    V = 2
Vecteur en (17,20)
(4) Pause D = 4
Vecteur en (17,20)

Durée totale du parcours = 15
Longueur totale du parcours = 20

Vecteur de départ = (5,10)
Vecteur d'arrivée = (17,20)
Appuyez sur <ENTRÉE> pour fermer cette fenêtre...

```

16. Ajouter une classe Arc, elle possède les données membres **rayon**, **angleDebut**, **angleFin**, 3 réelles doubles précisions. Les angles sont définis par rapport au cercle trigonométrique centré sur le centre de l'arc, ils sont exprimés également en radian.

Pour la longueur de l'arc, on donne la formule suivante

$\text{longueur} = |\text{angleDebut} - \text{angleFin}| * \text{rayon}$

La valeur absolue est obtenue avec la fonction fabs() de la librairie math.h

On donne la méthode permettant d'obtenir le vecteur résultant à la fin de l'arc

Vecteur Arc::ObtenirVecteurArrivee()

```

{
    Vecteur ptCentre(rayon * cos(angleDebut), rayon * sin(angleDebut));
    Vecteur ptArrivee(rayon * cos(angleFin), rayon * sin(angleFin));

    return (ptArrivee - ptCentre) ;
}

```

17. Complétez l'affichage et le programme principal pour obtenir le résultat final suivant :

| Quelques valeurs de PI | |
|------------------------|---------|
| M_PI | 3,14159 |
| M_PI/2 | 1,5708 |
| 3 * M_PI/2 | 4,71239 |

```

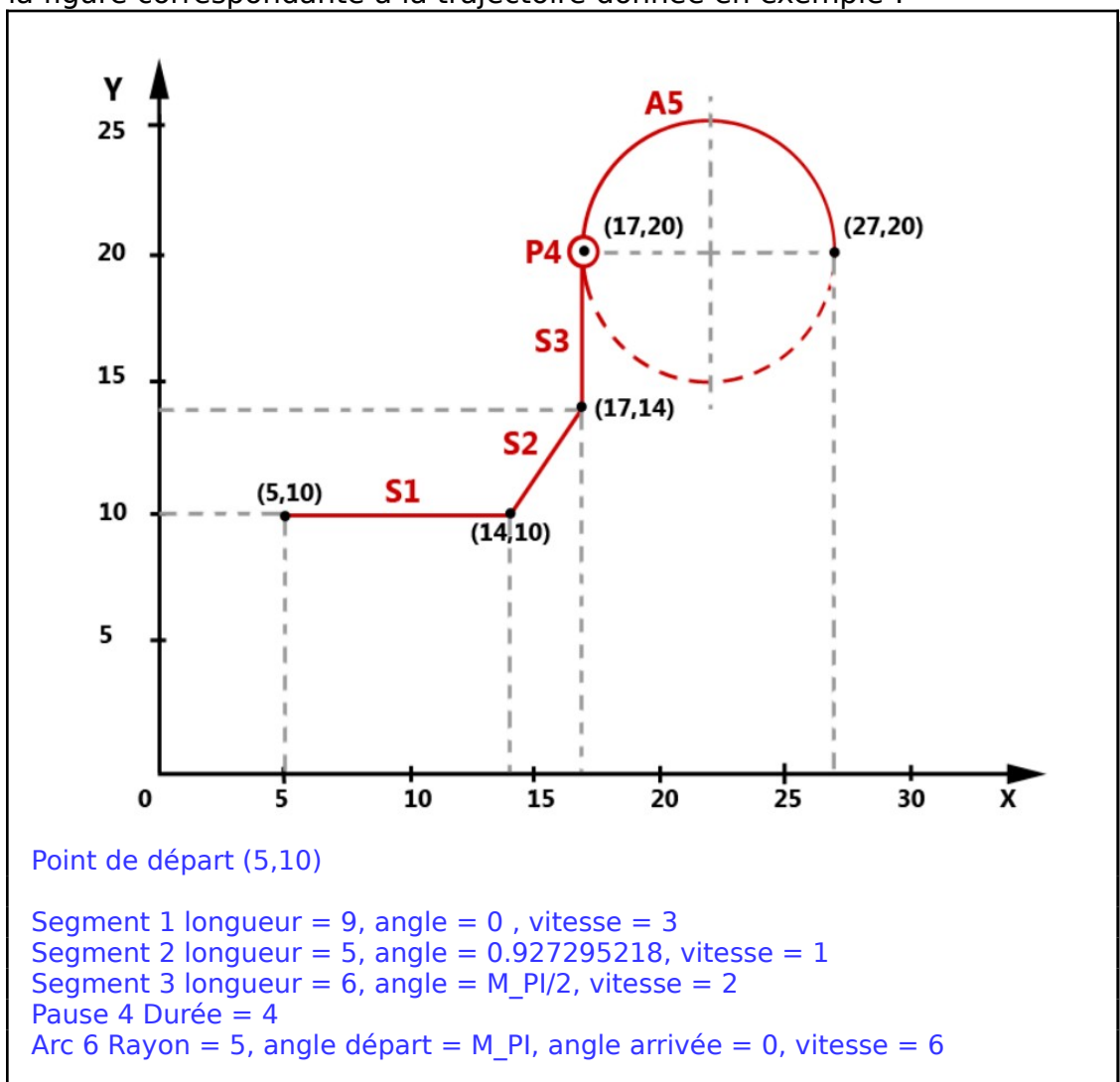
Fichier  Editer  Affichage  Rechercher  Terminal  Aide
Trajectoire :
(1) SEGMENT L = 9      A = 0      V = 3
Vecteur en (14,10)
(2) SEGMENT L = 5      A = 0.927295  V = 1
Vecteur en (17,14)
(3) SEGMENT L = 6      A = 1.5708    V = 2
Vecteur en (17,20)
(4) Pause D = 4
Vecteur en (17,20)
(5) ARC R = 5      Angle Debut = 3.14159  Angle Fin = 0      V = 6
Vecteur en (27,20)

Durée totale du parcours = 17.618
Longueur totale du parcours = 35.708

Vecteur de départ = (5,10)
Vecteur d'arrivée = (27,20)
Appuyez sur <ENTRÉE> pour fermer cette fenêtre...

```

Voici la figure correspondante à la trajectoire donnée en exemple :



18. Pour terminer, réalisez sous **Modelio** le diagramme de classe de votre application, enregistrez-le sous la forme d'une image **png** et insérez-la dans le fichier **Readme** de votre **GitHub**. Faites la mise à jour de votre dépôt **GitHub** en indiquant comme message d'indexation « **version finale** ».