**Forme.cs**

**Résumé**

Ce fichier définit la classe abstraite Forme, qui représente une forme géométrique générale. Elle sert de classe de base pour toutes les formes spécifiques (comme les cercles et les rectangles).

**Classe : Forme**

* **Attributs :**
  + Point point : La position d'un point associé à la forme, utilisé pour définir l'emplacement de la forme dans le plan 2D.
* **Constructeur :**
  + Forme(Point point) : Initialise la forme avec un point donné. Ce constructeur est appelé par les classes dérivées.
* **Méthodes abstraites :**
  + abstract public double Perimetre() : Méthode abstraite qui doit être implémentée dans les classes dérivées pour calculer le périmètre de la forme.
  + abstract public double Aire() : Méthode abstraite qui doit être implémentée dans les classes dérivées pour calculer l'aire de la forme.

**Rectangle.cs**

**Résumé**

Ce fichier contient la classe Rectangle, qui hérite de la classe abstraite Forme. Elle représente un rectangle défini par une longueur et une largeur.

**Classe : Rectangle**

* **Attributs :**
  + double Longueur : La longueur du rectangle.
  + double Largeur : La largeur du rectangle.
* **Constructeur :**
  + Rectangle(Point point, double Longueur, double Largeur) : Initialise un rectangle à partir d'un point, d'une longueur et d'une largeur. Appelle le constructeur de la classe parente (Forme).
* **Méthodes redéfinies :**
  + public override double Perimetre() : Calcule et retourne le périmètre du rectangle, qui est égal à 2 \* (Longueur + Largeur).
  + public override double Aire() : Calcule et retourne l'aire du rectangle, qui est égale à Longueur \* Largeur.

**Cercle.cs**

**Résumé**

Ce fichier contient la classe Cercle, qui hérite de Forme. Elle représente un cercle défini par son rayon.

**Classe : Cercle**

* **Attributs :**
  + double Rayon : Le rayon du cercle.
* **Constructeur :**
  + Cercle(Point point, double rayon) : Initialise un cercle à partir d'un point central et d'un rayon. Appelle le constructeur de la classe Forme.
* **Méthodes redéfinies :**
  + public override double Perimetre() : Calcule et retourne le périmètre du cercle, qui est égal à 2 \* Math.PI \* Rayon.
  + public override double Aire() : Calcule et retourne l'aire du cercle, qui est égale à Math.PI \* Rayon \* Rayon.

**Point.cs**

**Résumé**

Ce fichier contient la classe Point, qui représente un point dans un espace 2D, défini par deux coordonnées (x, y).

**Classe : Point**

* **Attributs :**
  + int x : Coordonnée X du point.
  + int y : Coordonnée Y du point.
* **Constructeurs :**
  + Point() : Constructeur par défaut qui n'initialise pas les coordonnées.
  + Point(int x, int y) : Constructeur qui initialise le point avec les coordonnées données (x, y).
* **Méthodes :**
  + public bool Equals(Point point) : Compare ce point avec un autre pour déterminer s'ils sont égaux (même coordonnées).
  + public override string ToString() : Retourne une chaîne représentant les coordonnées du point au format "(x, y)".

**Program.cs**

**Résumé**

Ce fichier contient le point d'entrée du programme. Il crée des instances de Point, Cercle, et Rectangle, et affiche leurs propriétés et méthodes calculées.

**Classe : Program**

* **Méthode : static void Main(string[] args)**
  + La méthode principale du programme.
  + **Étapes :**
    1. Création d'un objet Point avec des coordonnées (3, 4).
    2. Création d'un objet Cercle avec un rayon de 5, en utilisant le point précédemment créé comme centre.
    3. Affichage des informations du cercle, son périmètre, et son aire.
    4. Création d'un objet Rectangle avec une longueur de 8 et une largeur de 6, en utilisant le même point comme coin supérieur gauche.
    5. Affichage des informations du rectangle, son périmètre, et son aire

**Résumé des fonctionnalités**

* **Forme** est une classe abstraite qui représente un modèle pour les formes géométriques, fournissant des méthodes abstraites pour calculer le périmètre et l'aire.
* **Rectangle** et **Cercle** héritent de Forme et implémentent les méthodes pour calculer le périmètre et l'aire de ces formes spécifiques.
* **Point** représente une position dans un plan 2D, utilisée comme référence pour les formes.
* **Program.cs** contient le point d'entrée du programme, où des objets Point, Cercle, et Rectangle sont créés et utilisés.