Exercice - C++

GTech 2 - 2024/2025

Exercice 1

Faire la classe:

Vector2

Elle doit stocker une position x, y dans les floats.
Elle doit être capable de getter et setter ces positions.

Exercice 2

Faire la classe:

Entity

Elle doit stocker une position dans un Vector2.
Elle doit avoir un constructeur pour set la position.
Elle doit avoir un getter/setter virtuel pour la position.

Exercice 3

Faire la classe abstraite:

Pour rappel:
Une classe abstraite est une classe qui a au moins une fonction membre virtuel pur.

AMovable
Elle doit stocker un vecteur unitaire de direction dans un Vector2.
Elle doit stocker un float pour la vitesse.
Elle doit avoir un constructeur pour set la direction et la vitesse.
Elle doit avoir une fonction membre public virtuel pour setter la direction.
Elle doit avoir une fonction membre public virtuel pour setter la speed.
Elle doit avoir une fonction membre public virtuel pur pour se déplacer.

Exercice 4

Faire la classe:

Alive

Elle doit stocker dans un float le maximum de vie.
Elle doit stocker dans un float la vie actuel.
Elle doit avoir un constructeur pour set la vie.
Elle doit avoir des fonctions membres public virtuel pour:

- Récupérer le maximum de vie.
- Récupérer la vie actuel.
- Recevoir des dégâts / perdre de la vie.

Exercice 5

Faire l’interface:

Pour rappel:
Une interface est une classe qui contient que des fonctions membres virtuel pur.

IAttacker
Elle doit avoir la fonctions membres virtuel public pur pour attaquer un pointeur de Alive.

Exercice 6

Faire la classe:

StaticObject
Elle doit hérité de Entity.
Elle doit faire à sa création:

- Un set le x et y. (Prendre les valeurs en paramètre)
- Afficher “Static Object just created at x = ici-la-position-x and y = ici-la-position-y”

Exercice 7

Faire la classe:

BreakableObject
Elle doit hérité de Entity et de Alive.

Elle doit faire a sa création:

- Un set le x et y. (Prendre les valeurs en paramètre)
- Set le maximum de vie et la vie actuel. (Prendre les valeurs en paramètre)
- Afficher “Breakable Object just created at x = ici-la-position-x and y = ici-la-position-y with ici-la-maxlife life”

Elle doit override la fonction membre virtuel “Recevoir des dégâts” pour afficher:

- “Breakable Object just broke”

Ne pas oublier de d'appeler la fonction parent dans l’override.

Exercice 8

Faire la classe:

Mob
Elle doit hérité de Entity, de Alive et de AMovable.

Elle doit faire a sa création:

- Set le x et y de la position. (Prendre les valeurs en paramètre)
- Set le maximum de vie et la vie actuel. (Prendre les valeurs en paramètre)
- Set le vector déplacement. (Prendre les valeurs en paramètre)
- Afficher “Mob just created at x = ici-la-position-x and y = ici-la-position-y with ici-la-maxlife life with direction x = ici-la-direction-x and y = ici-la-direction-y”

Elle doit override la fonction membre virtuel “Recevoir des dégâts” pour afficher:

- “Mob just die”

Elle doit override la fonction membre virtual de déplacement:

- “Mob move to x = ici-la-position-x and y = ici-la-position-y

Ne pas oublier de d'appeler la fonction parent dans les overrides.

Exercice 9

Faire la classe:

Player
Elle doit hérité de Entity, de Alive, de AMovable et IAttacker.

Elle doit faire a sa création:

- Set le vector position. (Prendre les valeurs en paramètre)
- Set le maximum de vie et la vie actuel. (Prendre les valeurs en paramètre)
- Set le vector déplacement. (Prendre les valeurs en paramètre)
- Afficher “Player just created at x = ici-la-position-x and y = ici-la-position-y with ici-la-maxlife life with direction x = ici-la-direction-x and y = ici-la-direction-y”

Elle doit override la fonction membre virtuel “Recevoir des dégâts” pour afficher:

- “Player just died”

Elle doit override la fonction membre virtual de déplacement:

- “Player moved to x = ici-la-position-x and y = ici-la-position-y”

Elle doit implémenter la fonction de IAttacker:

- Enlever 10 point de vie du pointeur de Alive pris en paramètre
- Afficher “Player just attacked.”

Ne pas oublier de d'appeler la fonction parent dans les overrides.

Exercice 10

Dans une fonction main.

Initialiser un StaticObject, un BreakableObject, un Mob, un Player.

Tester que tout les fonctions marches correctement.

Exercice 11

Faire la classe:

World

Elle doit stocker des pointeur sur Entity dans un Vector.

Elle doit avoir une fonction membre Init:

- Qui va initialiser un StaticObject, un BreakableObject avec 1 point de vie, un Mob avec 20 point de vie, un Player avec 10 point de vie
  - Mettez les positions et vitesses que vous voulez.
- Qui va stocker leurs pointeurs dans le Vector

Elle doit avoir une fonction membre “Step” qui doit faire:

- Parcourir tous les pointeurs Entity du Vector et les faires les choses suivante en fonction de leur type:
  - Mob:Il doit se deplacer vers le BreakableObject.
  - Player: Si le Mob est en Vie: Il doit se déplacer vers un Mob et l’attaquer si il est à moins de 1 de distance du joueur. Si le Mob est mort, il doit se déplacer vers un BreakableObject et l’attaquer si il est à moins de 1 de distance du joueur.
- Si tous les Mobs et BreakableObjects sont morts, écrire “Simulation Finished”

Pour la vérification de type, regardez comment marche dynamic_cast.

Exercice 12

Dans la fonction main.

Initialiser un World.

Appeler la fonction Init et des Steps jusqu’à la fin de la simulation.