AoE2 est mon point de référence, et nous en tirerons un minuscule sous-ensemble de règles.

(1) Le jeu se déroule sur une carte, qui est une grille de taille N × M tuiles. La taille

La taille minimale absolue que vous devez pouvoir gérer est de 120 × 120, ce qui correspond à une "minuscule" carte dans AoE2.

(2) Les cartes seront générées de manière aléatoire - un fait que vous devez être en mesure de démontrer.

(3) Vous soutiendrez au moins deux types différents de cartes générées aléatoirement, chacune ayant des implications stratégiques et tactiques : par exemple, une carte avec des ressources généreuses réparties sur la carte (prenez la carte Arabia d'AoE2 comme référence pour voir à quoi cela ressemble) et une carte où tout l'or se trouve au centre de la carte.

(4) Limite de population = nombre maximum d'unités par joueur : 200.

La limite réelle en jeu est déterminée par les maisons et les centres-villes, dans les limites de ce maximum.

(5) Les ressources suivantes :

a. Bois (W), 100 par tuile (arbre)

b. Nourriture (uniquement par les fermes) (F), 300 par ferme

c. Or (G), 800 par tuile

(6) Unités :

a. Villageois : v

Coût 50F, 25 HP, Durée d'entraînement 25s,

2 attaque, vitesse 0.8 tuile/seconde.

peut construire des bâtiments

Le temps de construction nominal t d'un bâtiment donné ci-dessous est le temps nécessaire à un villageois pour construire un bâtiment seul.

villageois pour construire un bâtiment seul.

Si l'on utilise n villageois et que t est le temps de construction nominal restant, le temps de construction réel sera de 3t /(n+2) . Peut collecter des ressources au rythme de 25/minute, peut en transporter 20.

b. Épéiste : s

Coût 50F+20G, Temps d'entraînement 20s,

40HP, 4 attaques, vitesse .9.

c. Cavalier : h

Coût 80F+20G, Temps d'entraînement 30s,

45HP, 4 attaques, vitesse 1.2.

d. Archer : a

Coût 25W+45G, Temps d'entraînement 35s,

30HP, 4 attaque, 4 portée, vitesse 1.

7) Bâtiments :

a. Centre ville : T

Coût 350W, temps de construction 150 secondes,

1000 HP, 4x4, engendre des villageois, point de chute pour les ressources

Population : +5.

b. Maison : H

Coût 25W, Temps de construction 25 secondes,

200 HP, 2x2, Population : +5.

c. Camp : C

Coût 100W, Temps de construction 25 secondes,

200HP, 2x2, Point de chute pour les ressources

d. Ferme : F

Coût 60W, Temps de construction 10 secondes,

100HP, 2x2, Contient 300 vivres.

Note : c'est le seul bâtiment accessible à pied, cf. AoE2.

e. Caserne : B

Coûts 175W, Temps de construction 50 secondes,

500HP, 3x3, génère des épéistes

f. Écurie : S

Coûts 175W, Temps de construction 50 secondes,

500HP, 3x3, engendre des cavaliers

g. Champ de tir à l'arc : A

Coûts 175W, temps de construction 50 secondes,

500HP, 3x3, génère des archers

h. Garder : K (tour)

Coûts 35W, 125G, Temps de construction 80 secondes,

800HP, 1x1,

Tire des flèches : Attaque 5, portée 8

(8) Conditions de départ :

a. Maigre : 50F, 200W, 50G,

Centre ville, 3 villageois

b. Moyenne : 2000(F,W,G)

Centre ville, 3 villageois

c. Marines : 20000(F,W,G)

3 centres-villes, 15 villageois, 2 (casernes, écuries, champs de tir à l'arc)

(9) Mise en œuvre agile.

N'essayez pas d'implémenter toutes les unités et tous les bâtiments en une seule fois, avant de commencer à travailler sur les systèmes qui en dépendent.

Commencez par mettre en place uniquement les villageois et les centres-villes, donnez-leur de grandes ressources de départ et mettez en place des guerres de villageois.

Ensuite, une fois que la boucle du jeu a montré qu'elle fonctionnait, revenez en arrière et mettez en place une plus grande variété de ressources et d'unités.

Si, à la fin de la journée, vous n'avez pas un produit complet dans le sens où certaines unités

et des bâtiments manquent, mais que vous pouvez démontrer que ce que vous avez fonctionne comme prévu, ce ne sera pas l'idéal, mais le résultat sera toujours honorable. Vous aurez en effet créé un jeu d'IA, même s'il est limité.

D'un autre côté, si vous me dites que vous avez techniquement implémenté toutes les unités et tous les bâtiments, mais qu'aucun d'entre eux ne se déplace, ne se construit ou ne combat, alors vous n'avez rien d'intéressant à proposer

(10) Menu principal simple.

Ne passez pas beaucoup de temps à mettre en place un menu de jeu sophistiqué avec un joli arrière-plan, de la musique, etc. Je ne m'en soucie guère.

Ce qui m'importe, c'est la fonctionnalité. Vous devez être en mesure de créer un nouveau jeu avec différents paramètres, de sauvegarder et de charger un jeu de manière efficace. Que vous le fassiez par le biais d'une interface graphique ou entièrement par des options de ligne de commande, cela doit être facile et rapide à utiliser pendant la démonstration.

Je n'accepterais pas, par exemple, que votre jeu ne puisse avoir qu'un (ou trois, ou cinq) fichier(s) de sauvegarde à cause des limitations de votre interface graphique de menu. Si vous devez avoir une interface graphique de menu, utilisez les dialogues d'ouverture de fichiers standard à cette fin. TkInter est parfait pour ce type de tâches.

(11) Visualisation de cartes : terminal

Tout comme le jeu d'échecs est indépendant de l'échiquier que vous utilisez - pièces en bois ou en verre ? sur la table ou sur un ordinateur ou purement par le biais de la notation d'échecs par courrier électronique ou postal ou ... . - la logique de votre jeu doit être indépendante de sa visualisation graphique. Il est extrêmement important de comprendre ce point, qui a des conséquences considérables sur le temps de développement, la qualité et la fiabilité du code, etc.

Pour vous obliger à séparer la logique du jeu (les règles, son état dans l'abstrait) de la façon dont il est visualisé, je vous demande de fournir deux visualisations.

La première est une vue terminale. Il n'est pas nécessaire qu'elle montre tout, mais elle devrait être suffisante pour donner une idée générale de ce qui se passe pour les petits jeux sur de très petites cartes.

Vous utiliserez les lettres données pour chaque bâtiment / tuile pour représenter la carte. Par exemple :

s s W WWWWWW

WWWW WWWW

vFTTTT W W WWWW

FFTTTT

TTTT GGv CC

TTTT GGvCC K

Hv v

H

représente un petit village avec un centre ville, une mine d'or à l'est, gardée par un donjon, des bois au nord-est, deux soldats au nord-ouest, une ferme à l'ouest, avec un villageois marchant dessus (probablement en train de la construire ou de la cultiver), et deux autres villageois au sud-ouest, à côté de maisons (probablement en train de construire les maisons). Il faut pouvoir mettre le jeu en pause avec P, faire défiler la carte en utilisant ZQSD et les touches directionnelles (+Maj pour aller vite).

En appuyant sur TAB, on met le jeu en pause et on ouvre une page web (fichier HTML généré) listant toutes les unités du jeu et leurs statistiques (HP, position, etc.) et les tâches en cours, ainsi que toute donnée pertinente sur l'état des IA des joueurs.

Par exemple, on peut apprendre que les soldats ont perdu des HP, et que le fermier est en train de construire sa ferme.

Vous ne passerez pas trop de temps à rendre la page attrayante, mais pensez à la rendre lisible et consultable (sections pliables), etc.

(12) Visualisation cartographique : 2.5D

Indépendamment de la visualisation du terminal, vous devez fournir une visualisation graphique 2.5D (isométrique, basée sur des sprites) de haut en bas, dans le style d'AoE.

Cela signifie que vous pouvez démarrer ou charger un jeu soit en mode terminal, soit en mode GUI, ou même passer de l'un à l'autre à la volée, en utilisant le bouton F12.

Vous pouvez utiliser les sprites d'AoE ou d'autres jeux, si vous pouvez les extraire ou les télécharger. Il s'agit d'un projet de programmation, pas d'un cours d'art.

Vous aurez besoin d'un cadre graphique pour cette tâche. Plusieurs possibilités s'offrent à vous :

◊ PyGame, https://www.pygame.org est le choix le plus courant parmi les étudiants pour ce type de projets.

◊ la bibliothèque Arcade https://api.arcade.academy

https://learn.arcade.academy.

Très récente, mais active ; quelques groupes l'ont utilisée et ont eu une bonne expérience.

◊ TkInter,

◊ PySimpleGUI (avec le backend TkInter uniquement ; plus simple pour commencer)

◊ PyQt5 ou PyQt6 (plus puissant, plus complexe, exigences externes (bn))

◊ wxPython, https://www.wxpython.org

Liaisons avec wxWidgets, similaires à PyQt.

◊ PursuedPyBear, https://ppb.dev

Celui-ci semble tout juste sorti du four, et n'est pas documenté.

◊ Kivy, https://kivy.org

◊ ou tout ce qui fonctionne avec Python, vraiment, je ne suis pas difficile, ce qui compte c'est le résultat.

Testez les différentes possibilités et choisissez judicieusement.

N'essayez pas de faire cela en 3D

(13)Sauvegarde et chargement.

Ces jeux peuvent être longs. Vous devez être en mesure de sauvegarder l'état du jeu quand vous le souhaitez et de le charger sans perte d'informations. Notez que, si vous avez une IA, cela inclut ce que l'IA sait du monde, et plus généralement son état d'esprit (planification d'une attaque, plan de jeu, etc.) Vous devez être capable de gérer un nombre arbitraire de sauvegardes, en les manipulant comme des fichiers standards.

Cela sera extrêmement important pour la défense, car vous n'aurez pas le temps de jouer plusieurs parties complètes pendant la démonstration. Au lieu de cela, vous chargerez des parties sauvegardées, prises à des moments intéressants de diverses parties, pour montrer les grandes batailles, le plan de jeu de l'IA, etc.