1. CADRE DU PROJET

Vous avez le choix parmi la liste des projets proposés. Seul l'aspect fonctionnel sera pris en compte et non l'aspect graphique. L'utilisation de librairies est autorisée et encouragée afin de gagner en rapidité de développement.

2. OBJECTIFS PEDAGO

PROGRAMMATION

- créer un logiciel complet de A à Z
- utiliser des librairies tierces
- mettre en œuvre la persistance des données
- implémenter des actions CRUD

ALGORITHMES AVANCES

- répondre à un besoin fonctionnel par de l'algorithmie
- mettre en place des algorithmes complexes
- optimiser le code couteux

POO

- maîtriser la programmation orientée objet
- mettre en place un algorithme complexe

ARCHITECTURE LOGICIELLE

- effectuer des choix technologiques et respecter leurs conventions
- maîtriser la mise en place d'actions CRUD
- mettre en place une architecture de code

UML

- traduire un besoin fonctionnel en modèle de donnée
- concevoir une architecture logicielle en amont du code

IOT / PROGRAMMATION BAS NIVEAU

- créer une communication entre son logiciel et l'extérieur (machine to machine ou machine to reality)

MATHEMATIQUE

- maîtriser la logique booléenne
- manipuler des matrices

3. LIVRABLES

- dépôt GIT de votre logiciel à jour => bejaoui+livrables@gmail.com
- exécutable de votre logiciel
- un document de présentation de votre projet (rôles de chacun, technologies utilisées, structure algorithmique, fonctionnalités majeures, captures d'écran…)
- Slides de votre présentation (optionnel)

4. MODALITÉS D'ÉVALUATION DU PROJET

Vous serez évalué sur l'ensemble des productions. L'évaluation prendra aussi la forme d'une présentation orale de synthèse d'environ 10 minutes accompagnée d'un support de présentation et d'une démonstration des fonctionnalités mises en place.

Des évaluations intermédiaires auront également lieu au cours du déroulement du Projet.

RÉCAPITULATIF DE LA GRILLE DE NOTATION

Selon le nombre de points de difficulté total mis en place sur votre projet, des points bonus seront accordés selon les paliers suivants :

Points de difficulté : Entre 23 et 28	*0 point bonus*
Points de difficulté : Entre 28 et 35	*1 point bonus*
Points de difficulté : Plus de 35	*2 points bonus*

Les projets proposés devront être réalisés dans leur intégralité pour obtenir la totalité des points.

LISTE DES PROJETS AU CHOIX :

PROJET : CARD GAME

Présentation

Projet inspiré du jeu “Castle Siege: Fantasy Card Game”. Le but est d’augmenter ses données tout en réduisant celles de l’adversaire afin de gagner la partie.

Fonctionnalités

- cinq écrans : *Difficulté : 2*
  - menu principal avec accès aux autres écrans
  - écran de jeu avec possibilité de retour
  - écran de fin de partie
  - écran d’instructions avec possibilité de retour
  - écran d’options avec possibilité de retour
- l’écran d’instructions devra présenter à l’utilisateur toutes les données nécessaires à la bonne utilisation du logiciel *Difficulté : 1*
- modèle de données :
  - deux camps opposés, qui possèdent chacun : *Difficulté : 2*
    - un index de joueur, premier ou deuxième
    - des points de vie, par défaut 100, minimum 0, maximum 100
    - des points de bouclier, par défaut 30, minimum 0, maximum 100
    - 3 ressources différentes, qui possèdent chacune :
      - un fournisseur de ressource, minimum 1
      - un stock de ressource, par défaut 10, minimum 0
    - une main de 7 cartes
  - une carte possède : *Difficulté : 2*
    - un nom
    - un type de ressource à dépenser
    - un coût
    - un type de donnée de camp à modifier (vie, bouclier, fournisseurs et stock)
    - une valeur
    - si elle affecte l’ennemi ou soi-même
    - une probabilité d’apparition
- Déroulement d’un tour :
  - le stock des ressources du joueur se voient incrémenté de 1 par fournisseur qu’il possède *Difficulté : 1*
  - si la main du joueur possède moins de 7 cartes, une carte est piochée en tenant compte des probabilités d’apparition des cartes. *Difficulté : 2*
  - le Joueur a deux possibilités :
    - jouer une carte s'il peut payer le coût en ressource *Difficulté : 2*
    - jeter une des cartes de sa main *Difficulté : 1*
  - fin du tour, début du tour adverse *Difficulté : 1*

- fin d'une partie : *Difficulté : 2*
    - la partie se termine si un joueur à ses points de vie à 0 lors du début de son tour
    - le logiciel bascule alors sur l'écran de fin de partie, où diverses informations de la partie seront décrites
- écran d'option :
    - il devra permettre à l'utilisateur de créer, consulter, modifier et supprimer (CRUD) toutes les cartes utilisées dans le logiciel *Difficulté : 5*
    - ces données devront être stockées dans un fichier afin de pouvoir les réutilisées même après fermeture du logiciel *Difficulté : 3*
- deux modes de jeux devront être proposés :
    - mode deux joueurs en local *Difficulté : 1*
    - mode jouer contre une IA *Difficulté : 5*

Degré de difficulté total : 30 points

PROJET : BRICK SHOOTER

Présentation

Projet inspiré d'un mini-jeu compris dans les vieilles consoles All in one. https://youtu.be/RPTanMNGmek?t=155

https://www.youtube.com/watch?v=dvjapcHsqXY

Fonctionnalités

- cinq écrans : *Difficulté : 2*
    - menu principal avec accès aux autres écrans
    - écran de jeu avec possibilité de retour
    - écran de fin de partie
    - écran d'instructions avec possibilité de retour
    - écran d'équipement avec possibilité de retour
- l'écran d'instructions devra présenter à l'utilisateur toutes les données nécessaires à la bonne utilisation du logiciel *Difficulté : 1*
- modèle de données :
    - un joueur possédant : *Difficulté : 2*
        - des crédits
        - une vitesse de tir
        - une vitesse de déplacement
        - une vitesse de défilement
    - des types de bloc possédant : *Difficulté : 2*
        - un type de donnée du Joueur, peut être nul
        - une valeur, peut être positive ou négative ou nul
        - une probabilité d'apparition (il vaut mieux que le bloc n'affectant pas le joueur soit celui de plus forte probabilité)

- pour les types de bloc, avoir au minimum :
    - un bloc neutre à forte probabilité
    - un bloc d'augmentation de vitesse de défilement à faible probabilité
    - un bloc d'augmentation de vitesse de déplacement à faible probabilité
    - un bloc de diminution de la vitesse de tir à faible probabilité
- déroulement d'une partie :
    - le Joueur apparait au milieu en bas de l'écran, il peut se déplacer de gauche à droite jusqu'aux bords de l'écran *Difficulté :* 2
    - des blocs apparaissent aléatoirement en haut de l'écran, faisant défiler vers le bas ceux déjà apparues *Difficulté :* 3
    - le Joueur peut tirer un projectile qui, une fois un bloc heurté, l'active et le casse.
      Un bloc activé peut modifier une donnée du joueur *Difficulté :* 3
- fin d'une partie : *Difficulté :* 2
    - la partie se termine quand le joueur le décide OU quand un bloc descendu trop bas touche le joueur
    - le logiciel bascule alors sur l'écran de fin de partie, où diverses informations de la partie seront décrites
    - si le joueur a décidé lui-même de quitter, le nombre de blocs qu'il a cassé s'ajoute à ses crédits
    - si le joueur a heurté un bloc, il ne gagne pas de crédits
- écran d'équipement :
    - il devra permettre à l'utilisateur de modifier (moins ou plus) toutes ses données (vitesse de tir, de déplacement, de défilement) monnayant les crédits qu'il a accumulé *Difficulté :* 5
    - ces données devront être stockées dans un fichier afin de pouvoir les réutilisées même après fermeture du logiciel *Difficulté :* 3

Degré de difficulté total : 23 points

PROJET : SERVEUR DE MATCHMAKING

Présentation

Ce projet contient 3 composantes : un serveur de Matchmaking, un logiciel client et une base de données.

Fonctionnalités

- modèle de données :
    - une file d'attente, contenant pour chaque attendant : *Difficulté:* 1
        - le moyen de communiquer avec lui (IP et port par exemple)
        - un pseudo
        - la date à laquelle il est entré dans la file
    - des matchs, contenant pour chacun : *Difficulté :* 1
        - le moyen de communiquer avec le joueur 1

            - le moyen de communiquer avec le joueur 2
        - le plateau de jeu
        - si le match est fini
        - s'il y a eu victoire du joueur 1, du joueur 2 ou égalité
    - des tours, contenant pour chacun : *Difficulté : 1*
        - la liaison avec le match
        - qui a joué : le joueur 1 ou le joueur 2
        - l'information du coup joué (en fonction du jeu choisi)
- le serveur de Matchmaking contient :
    - le lien avec la base de données *Difficulté : 3*
    - un système de socket avec les actions suivantes : *Difficulté : 5*
        - arrivé d'un client dans la file d'attente (réception)
        - début d'un match (envoie)
        - réception d'un tour (réception puis envoie)
        - fin d'un match (envoie)
    - une vérification constante de la file d'attente et création de matchs en fonction
      Difficulté : 2
    - une logique de jeu. Vous êtes libre quant au choix du jeu. Cela doit cependant être un jeu de plateau au tour par tour (Ex : puissance 4, dames, morpion, ...) *Difficulté : 5*

- le logiciel client contient :
    - un système de socket avec les actions suivantes : *Difficulté : 5*
        - entrée en file d'attente (envoie)
        - début d'un match (réception)
        - jouer un coup (envoie)
        - prendre en compte le coup adverse (réception)
        - fin d'un match (réception)
    - une partie de la logique du jeu choisis. *Difficulté : 2*
    - soit : *Difficulté : 3*
        - une IHM pour pouvoir jouer
        - une CLI avec IA

Degré de difficulté total : 28 points

PROJET : BORNE D'ARCADE

Présentation

Ce projet consiste à créer une mini borne d'arcade contenant un jeu simple de 1 VS 1, à l'aide d'une Raspberry pi et de boutons/joystick à assembler.

Fonctionnalités

- cinq écrans : *Difficulté : 3*
    - menu principal avec accès aux autres écrans et présentation des meilleurs scores

    - écran de jeu
    - écran de fin de partie
    - écran d'instructions avec possibilité de retour
    - écran d'option avec possibilité de retour
- l'écran d'instructions devra présenter à l'utilisateur toutes les données nécessaires à
  la bonne utilisation du logiciel *Difficulté :* 1
- modèle de données :
    - deux joueurs possédant : *Difficulté :* 2
        - une différenciation joueur 1 / joueur 2
        - une vitesse de rotation
        - une vitesse de déplacement
        - des points de vie
        - une puissance de tir
        - un délai de tir
        - une vitesse de projectile
    - des Scores possédants : *Difficulté :* 1
        - un pseudo
        - un score

- branchement des capteurs d'action et réception de leurs informations : *Difficulté :* 5
    - deux Joystick : un par joueur
    - quatre boutons poussoir : deux par joueur
- toute la navigation UI, le jeu ainsi que le remplissage des champs texte doivent pouvoir se faire par le biais des capteurs d'action *Difficulté :*5
- déroulement d'une partie : *Difficulté :* 5
    - les deux joueurs apparaissent de part et d'autre de l'écran
    - l'espace de jeu se limite à l'écran, pas de physique, la vue caméra est dite "top- down"
    - le Joystick doit permettre au joueur de pivoter à 360° et d'avancer vers l'avant
    - un des boutons doit permettre au joueur de tirer un projectile
    - les données de la base doivent être utilisées pour gérer les capacités des joueurs
    - un Joueur perd des points de vie quand un projectile le touche
- fin d'une partie : *Difficulté :* 3
    - la partie se termine quand un joueur n'a plus de points de vie
    - le logiciel bascule alors sur l'écran de fin de partie, où le score du gagnant s'affiche ainsi que sa position dans les meilleurs scores
    - pour le score, vous devez créer une formule prenant en compte la différence de points de vie, le temps passé sur la partie ainsi que la différence sur les données de BDD du joueur 1 et 2 (un joueur peut gagner un plus gros score en baissant sa vitesse de rotation par rapport à son adversaire par exemple)
- écran d'option :

- il devra permettre aux joueurs de modifier toutes leurs données (puissance de tir, vitesses ...) *Difficulté : 3*
- ces données devront être stockées dans une base de données, afin de pouvoir les réutilisées même après fermeture du logiciel *Difficulté : 2*

Degré de difficulté total : 30 points

RESSOURCES COMPLEMENTAIRES

Exemple de boutons/joystick pour le projet Borne d'arcade : https://www.amazon.fr/Joystick- Boutons- Ensemble-Console-Encodeur/dp/B07PMCHM63

BESOINS MATERIELS ET LOGICIELS

- un IDE
- un langage de programmation orienté objet
- GIT
- un serveur (pour le projet n°1)
- un Raspberry et des boutons/joystick (pour le projet n°2)