

**Blackjack**

Programmation III

420-P45

# Avant-propos

Ce travail pratique consiste à produire une application graphique en utilisant le langage C#, en appliquant une approche de programmation orientée objet et en implémentant divers concepts vu en classe pendant la session.

Vous devrez coder un jeu de blackjack permettant à plusieurs joueurs de s’affronter à distance via un réseau IP.

De plus, vous devrez programmer un joueur virtuel capable de participer efficacement à une partie et même peut-être de gagner contre des humains.



# Objectifs

* Programmer une application en utilisant l’approche orientée objet
  + En utilisant le langage C#
  + En respectant les règles de programmation d’une classe
  + En utilisant la librairie Winform
* **Implémenter une communication réseau TCP/IP**
  + **En utilisant les objets .NET existants**
  + **En créant un protocol de communication de niveau 7**
  + **En utilisant la programmation multi-fil**
* Créer une libraire liée dynamique (DLL)
  + En évaluant l’ordre de complexité de votre algorithme
* Écrire du code robuste et général

# GUI

Vous devrez construire une interface graphique en utilisant la librairie WinForm ainsi que les objets graphiques fournis par la plateforme .NET permettant de gérer et de jouer des parties de poker multijoueurs.

* Créer une nouvelle partie

Un menu devra permettre de créer une nouvelle partie, l’on doit ensuite préciser le nombre de joueurs, le montant initial de chaque joueur et la mise minimale.

* Joindre une partie

Une autre entrée de menu doit permettre de joindre une partie distante en précisant l’adresse IP à laquelle l’on désire se connecter.

* Charger joueur virtuel

Cette option permet de choisir un joueur virtuel en sélectionnant le fichier .dll correspondant

* Zone de jeu

Une aire de jeu doit afficher la partie en utilisant des images de cartes et divers contrôles visuels pour afficher les mises, le pot, les mains des joueurs, la dernière action et autres messages destinés aux joueurs.

Vous êtes libre de construire votre interface comme vous le désirez, vous pouvez vous inspirer d’interfaces existantes et utiliser des images que vous trouvez sur le net.

L’interface doit contenir les éléments suivants :

* + La main de chaque joueur (cachée ou révélée selon le cas)
  + Le montant du pot
  + Le montant du joueur
  + Le tour du joueur actuel
  + La dernière action jouée

# Joueur virtuel

Votre joueur virtuel sera encapsulé dans le code d’une .dll qui vous sera possible de charger dans le jeu à la place d’un joueur humain. Il doit être en mesure de décider de la prochaine action à prendre en fonction des cartes présentent dans sa main ainsi que dans celle du « dealer ».

Essentiellement il décidera entre les deux choix suivants : « HIT » ou « STAND »

# Communication réseau

Lorsque le créateur de la partie est prêt à recevoir les autres joueurs, il attend leur connexion et la partie de démarre pas tant que tous les joueurs ne sont pas connectés.

Au moment de la connexion le joueur se voit attribuer un numéro de joueur, le créateur étant le joueur 1.

Lorsque vient le temps à un joueur de jouer son tour, il doit envoyer son action au serveur qui lui va retransmettre l’action aux autres.

Le protocole de communication réseau est le même que pour les actions du joueur virtuel.

# Blackjack battle

Si tous est implanter comme prévu il devrait être possible de faire combattre plusieurs joueurs virtuels ensemble afin de déterminer lequel est le meilleur!!!

# Échéancier

La démonstration du programme aura lieu le \_\_\_\_\_\_\_\_\_\_\_

Vous avez jusqu’au \_\_\_\_\_\_\_\_\_ pour la remise des autres documents.

# Objets de remise

* Le code source de votre programme
* Un document texte décrivant chacune de vos classes en précisant :
  + Son nom
  + Courte description
  + Ses données membres (nom, type et description)
  + Ses méthodes (signature, description)
* Un petit document texte décrivant l’analyse de vos algorithmes
* Le fichier DLL de votre joueur virtuel.
* Démonstration en personne du fonctionnement de votre application