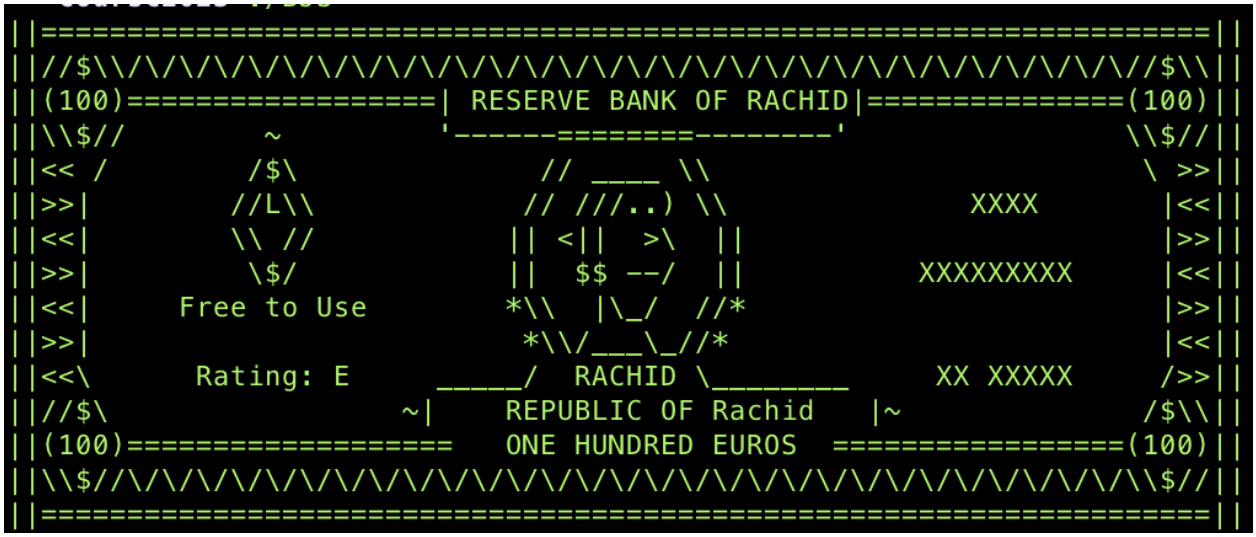
Devoir - BlackJackSimple



Instructions pour l'Évaluation:

- Date: Le devoir doit être rendu le 18 Octobre 2023 avant 13h.
- **Soutenance EN PRESENTIEL**:
  - Vous aurez 10min pour presenter votre code.
  - Vous aurez une question et vous aurez 50min pour implementer la solution en classe entre 13h30 et 17h le 18 Octobre.
  - 10min pour presenter la solution de la question posée.

PARTIE 1. [~14 points]: UN SEUL JOUEUR

Dans la premiere version du BlackJack que vous allez implementer les règles sont simplifiées. Il n'y a qu'un joueur qui joue (pas de banques pour le moment.). Implementer les règles suivantes:

Règles :

Les cartes de 1 à 10 on une valeurs respectives de 1 à 10. Les valeurs des cartes seront contenu dans le tableau suivant:

```
int cartes [52] =
{1,1,1,1,2,2,2,2,3,3,3,3,4,4,4,4,5,5,5,5,6,6,6,6,7,7,7,7,8,8,8,8,9,9,9,9,10,1
0,10,10,10,10,10,10,10,10,10,10,10,10,10,10};
```

- Le joueur à un solde de $1000 au début du jeu.

- Le joueur fait une Mise au debut de la partie. Par exemple il peut miser $50 qui lui seront ajouter si il gagne la partie et déduit de son solde si il perd la partie.
- 2 valeurs de cartes sont tirées au hazard dans le tableau `cartes` décrit ci-dessus. Vous utiliserez la fonction suivante pour le tirage au hazard:

```
int tirerCarte(int min, int max){
      int resultat = (rand() % (max - min + 1)) + min;
      return resultat;
}
```

On pensera à rajouter `#include <time.h>`

- Les valeurs des cartes en main sont additionné et stockée dans une variables `sommeMainJoueur`.
- Si `sommeMainJoueur > 21` alors le joueurs a perdu, la partie est terminé et le solde est décompté de la mise avant d'être affiché.
- Vous utiliserez une variable booléenne `terminer` qui sera initialisé à faux et qui sera vrai lorsque le joueur veux arrêter la partie ou lorsqu'il a perdu.
- A chaque tour du jeu, le joueur pourra choisir de :
  - 1. (HIT) Tirer une nouvelle carte. Dans ce cas la valeur de la carte tirée au hazard sera ajouté `sommeMainJoueur`.
  - Afficher à chaque tour la valeur de `sommeMainJoueur`.
  - 2. (STAND) De ne pas prendre une nouvelle carte et comptabiliser sa mise dans son solde.
  - 3. (QUIT) Quitter le jeu.
  - Ces options de tour apprêteront dans une boucle `while`.

PARTIE 2 [~4 points]: JOUER CONTRE LA BANQUE

Dans la deuxième version du BlackJack que vous allez implementer les règles pour jouer contre la banque. Implementer les règles suivantes:

Règles :

- La Banque reçoit aussi 2 valeurs de cartes aussi.
- Comme pour le joueur le totale des valeurs de sa main est stockée dans `sommeMainBanque`. La banque perd si `sommeMainBanque > 21`. ET le joueur remporte sa mise.
- Lorsque le jouer decide de prendre sa mise (STAND) et de ne pas prendre de nouvelle carte, on vérifie si `sommeMainJoueur > = sommeMainBanque`. Si c'est

vrai alors le joueur remporte la mise sinon c'est la banque qui gagne et la mise est déduite du solde du joueur.

- Lorsque le joueur choisi une carte supplémentaire, la banque aussi reçoit une nouvelle carte

PARTIE 3 [~2 points]: EXTENSIONS ET BONUS POSSIBLES

Implementer une des extensions suivantes.

- Afficher toute les cartes du joueurs tirés en main pendant chaque tour. On pourra utiliser un tableau de `carteMainJoueur` .
- Faite en sorte qu'une meme carte ne peut pas être tiré plus d'une fois.
- Créer une version du tableau de carte avec une struct faisant apparaitre la couleur de la carte (pic, coeur, carreau, trèfle).
- Afficher a chaque tour la probabilité pour que le joueur perde si il decide de rajouter une nouvelle carte à sa main (HIT).