CAMEL UP! UP! UP!

Castillo Esteban, Collette Paul, 2022

Eléments d'analyse

Utilisateurs visés:
Camel up est un jeu de 2 à 8 joueurs, les règles sont facilement et rapidement compréhensible par tout le monde, le jeu est donc destiné aux enfants de 8 ans et plus ainsi qu'au adulte souhaitant se divertir entre amis autour d'un jeu.
Fonctionnalités (services /fonctions offerts par l'application):
-Le programme respecte les règles du jeu et la partie se déroule comme une vrai partie.
-Afficher le plateau et donner les informations nécessaires au déroulement du jeu aux joueurs.
-Demander au joueur quand c'est son tour ce qu'il veut faire et faire en sorte que le programme réagisse en fonction.
Donnés nécessaires au programme:
-Nombre de joueur (saisie utilisateur)
-Choix de ce que veut faire le joueur à chaque tour (saisie utilisateur)
-Sous programmes qui permettent de simuler un lancé de dé (au hasard), gérer le temps (attendre quelques secondes), (fichier code blocks)
-Nous créerons les autres sous programmes nécessaires.
Résultats :
-La position des chameaux est donnée grâce à un plateau de 17 cases
-Les données (argent, carte désert, pari manche, carte pyramide) sont données à gauche du plateau pour chaque joueur.
-Les paris partie ne sont pas visibles conformément aux règles du jeu (il est seulement marqué en quel position le joueur à parier mais pas sur quel chameau), le classement est affiché à la fin de la partie.
Méthode:
Le programme est découpé en 3 partie (début/initialisation et collecte de données initial, Partie avec déroulement des manches ainsi que les choix de chaque joueur, fin de partie avec affichage du classement)
Faisabilité :
Toutes les données du jeu sont facilement retranscriptible en langage informatique car elles sont stockable sous forme d'entiers (case, hauteur du chameau, argent, nombre de carte pyramide, position des cartes déserts)
Limites :
Tant que l'utilisateur ne saisie pas des données qui n'ont aucun sens comme des caractère à la place de chiffre ou autre, le programme peut le gérer.
Scénarios d'utilisation :
La position des chameaux est déterminé aléatoirement en début de partie (case entre 1 et 3), les joueurs jouent chacun leur tour et ont 4 possibilités: (parier sur la manche, parier sur la partie, tirer un dé, placer une case de désert), une fois que tous les dés ont été lancés la manche se termine (on remet les dés dans la pyramide, réparti l'argent des paris), une fois qu'un chameau arrive à la fin la partie se termine (affichage classement).

Schéma des Structures de données utilisées

Types structurés

```
struct mise_manche{
        int mise_rouge[3];
        int mise_bleu[3];
        int mise_blanc[3];
        int mise_jaune[3];
        int mise_vert[3];
        };
  typedef struct mise_manche s_mise_manche;
```

```
struct info_camel{
   int hauteur;
   int case_;
   int couleur;
};
 typedef struct info_camel s_info_camel;
```

```
struct pari_parti{
int* premier;//premier[0]=3>>> le joueur 1 a
int* ordre_premier;//ordre_premier[0]=6>>>le
int* dernier;
int* ordre_dernier;
};
typedef struct pari_parti s_pari_parti;
```

Variables

```
int nb_joueur;
int joueur venant de jouer=0;
```

Tableaux dynamiques

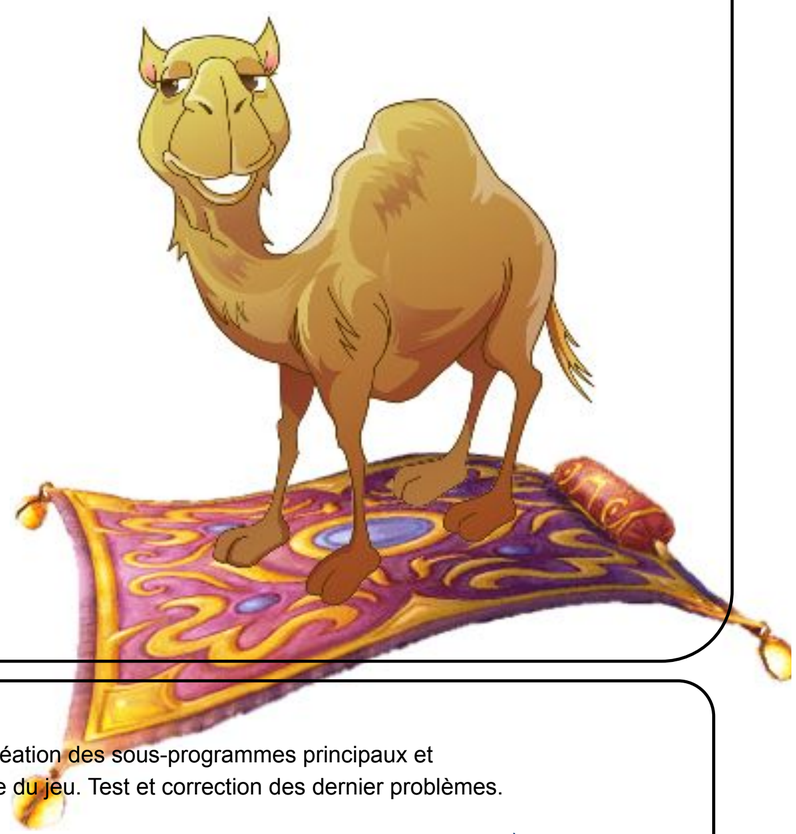
Une case par joueur
Case 0 -> joueur 1
Case 1-> joueur 2
etc…

```
int* argent;
int* desert_joueur;
int* carte_pyramide;
argent= (int*)malloc(sizeof(int)*nb_joueur);//dimension
desert_joueur= (int*)malloc(sizeof(int)*nb_joueur);
carte_pyramide= (int*)malloc(sizeof(int)*nb_joueur);
```

Tableaux

Chameau
rouge bleu blanc jaune vert

```
int hauteur_camel[5]={0,0,0,0,0};
int case_camel[5]={0,0,0,0,0};
int dede[5]={1,1,1,1,1};
```



Planning: Réflexion global sur feuille, création des variables et répartition des sous-programmes:
Paul -> Affichage, tirage des dés, cartes desert, fin de manche
Esteban -> Paris partie et manches, tour du joueur, début/fin partie

20 MARS

Codage des sous-programmes séparément mais beaucoup de communication pour faciliter la compréhension. Mise en commun des sous-programmes affichage et tour du joueur pour commencer la structure final du jeu.

31 AVRIL

Mise en commun, création des sous-programmes principaux et création de la structure du jeu. Test et correction des dernier problèmes.

12 MAI

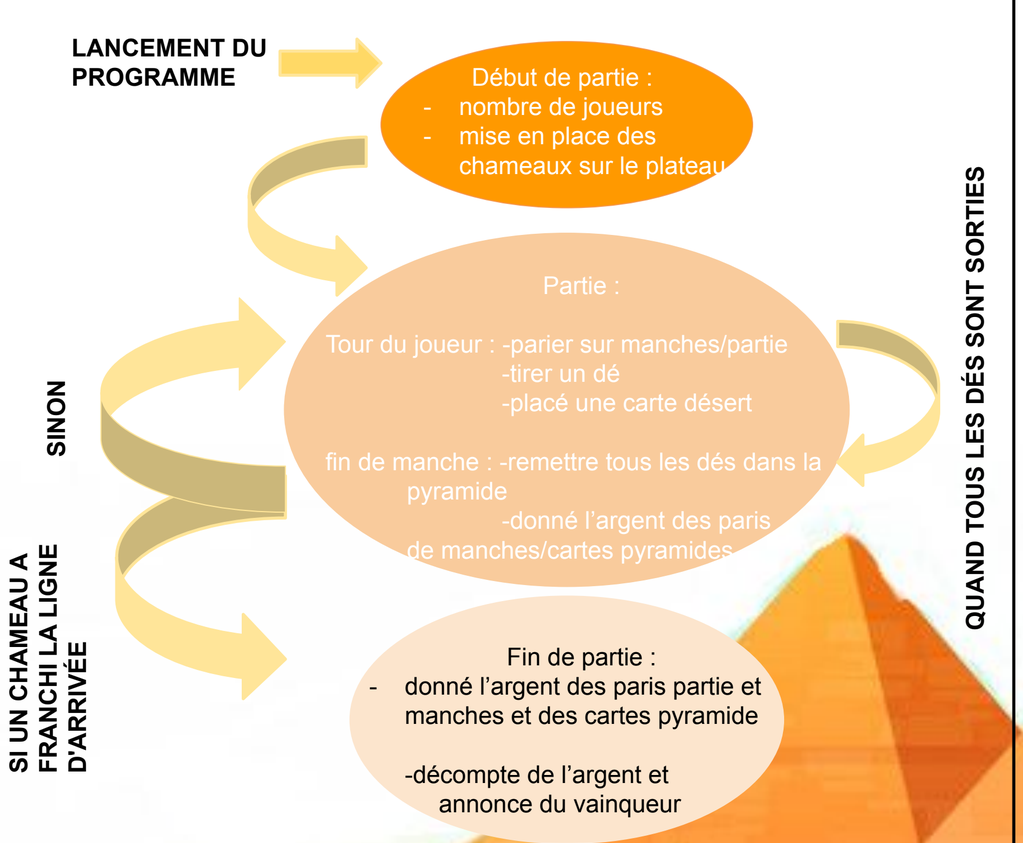
Problèmes rencontré:

Désaccord sur le stockage des information (plusieurs tableaux de dimension le nombre de joueur ou type structuré de plusieurs tableaux. Mauvaise compréhension de certain aspect du jeu (carte desert).

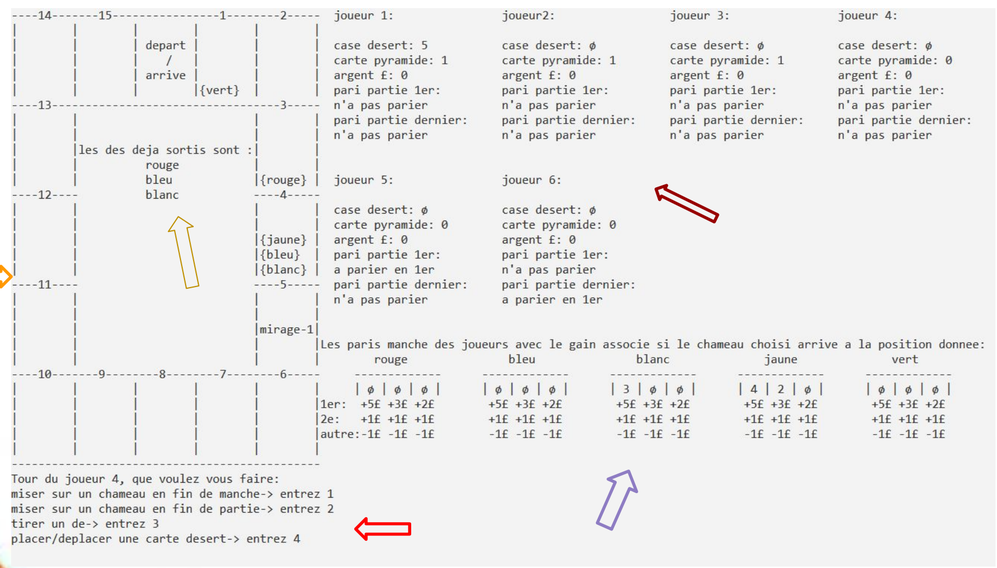
Esteban-> problèmes lors du codage des paris manches, code très long et difficulté à l'optimiser.
Paul->

Confusion dans le nom des variables qui nous a fait perdre beaucoup de temps. Probleme dans la distribution de l'argent difficile à comprendre dû à la taille du programme. Confusion dans le numéro des joueurs.

Eléments de conception



Capture d'écran



Conclusion

Ce projet nous a permis de pratiquer bien plus que nous avions pu le faire en cours. Au fur et à mesure de l'avancé, coder des programmes est devenu bien plus rapide et automatique. Nous avons aussi compris l'importance de la conception, la discussion et la compréhension du sujet. Nous avons pris l'habitude de faire des appel régulier pour communiquer sur notre avancé. Grace a ces mesures, la mise en commun, même si elle nous a pris un certain temps, n'a pas été très compliqué.

Même si dans l'ensemble, le projet c'est bien dérouler, certain point aurait pu être optimisé:

- Il aurait surement été plus simple de créé plus de type structuré, pour limiter le nombre de variable appelé dans chaque sous programmes.
- Nous aurions aussi pu améliorer les boucles, pour eviter de parcourir tout nos tableaux et seulement parcourir les cases necesaires
- Nous aurions sûrement pu optimisé le sous-programmes récompense_pari_manche, qui est très répétitif et prend énormément de lignes.

ESTACA
CREATEUR DE NOUVELLES MOBILITES
GROUPE ISAE