BUT INFORMATIQUE - PARIS DESCARTES

Comparaison d’approches algorithmiques

Projet réalisé par Yohann PECH et Mehdi Bouziane du groupe 106

2022-2023

Table des matières

Présentation du projet :

Cet SAE d'initiation au développement correspond à notre deuxième travail en groupe en rapport à la programmation, mais notre premier en C++. Ce projet était d'une certaine complexité au niveau de l'algorithmie mais était aussi l'occasion de corriger les erreurs du projet précédent en nous initiant à la compilation séparée avec plusieurs fonctions sur plusieurs fichiers.

L'objectif du projet était de réaliser un jeu qui permettrait l'affrontement entre humain/humain, humain/robot ou encore robot/robot.

Le but du jeu était de former un mot du dictionnaire de la langue française en annonçant chacun son tour une lettre. Au fur et à mesure, les joueurs forment un mot, et si un joueur a un doute, il peut demander au joueur précèdent le mot auquel il pensait en saisissant le caractère '?'.

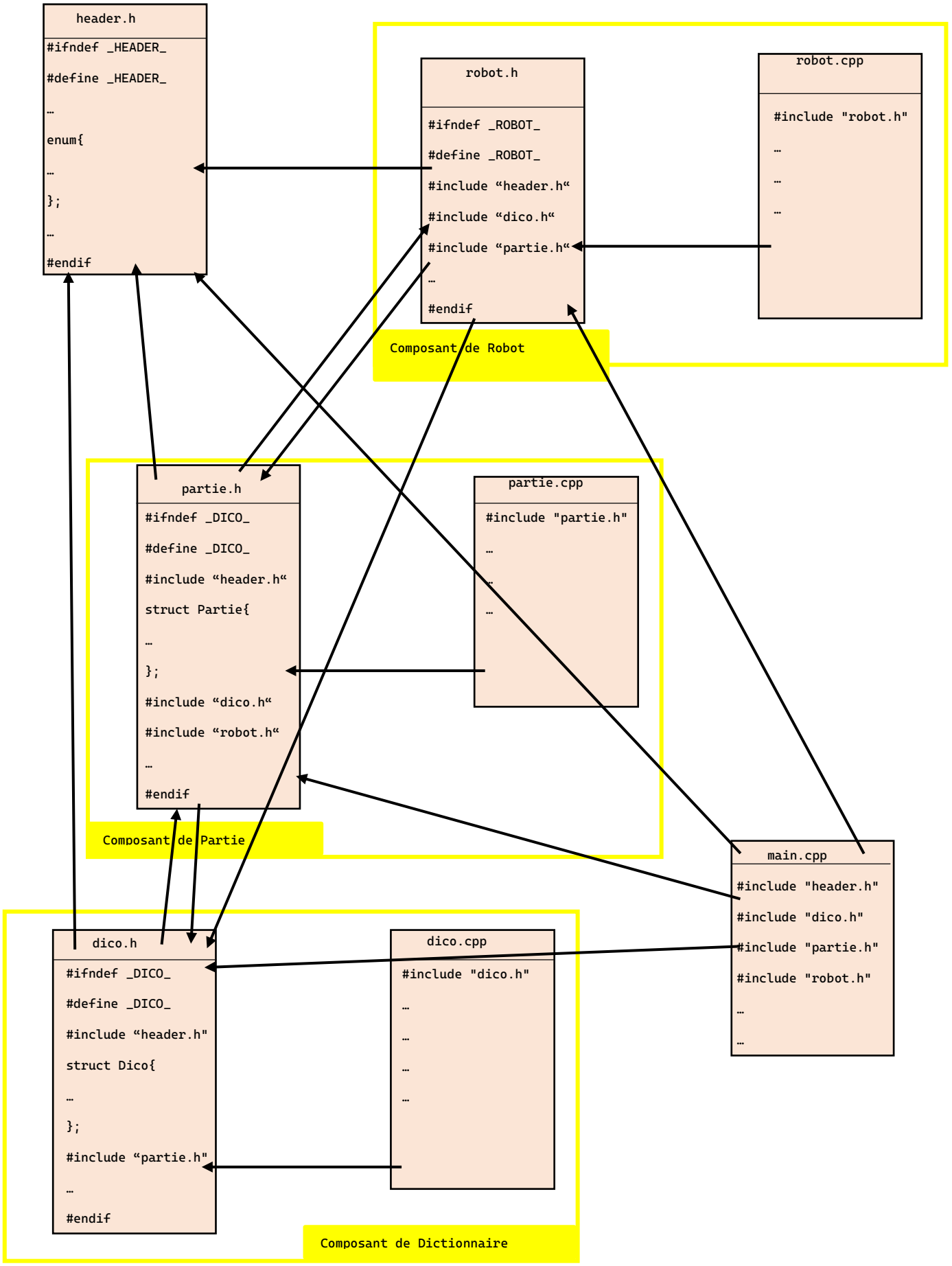
Le jeu se termine lorsqu'un seul joueur obtient quatre quarts de singe et dans ce cas-là, il perd. Un quart de singe peut s'obtenir :

- En donnant la dernière lettre de n'importe quel mot existant dans le dictionnaire fourni. (Par exemple, le mot formé dans la partie est 'ABR', et je pense au mot 'ABRICOT'. Alors je mets un 'I' mais la conséquence est que je prends un quart de singe, car j'ai formé le mot 'ABRI', qui est un mot existant dans le dictionnaire.
- En abandonnant la manche par la saisie du caractère ' !'
- En saisissant le caractère ' ?' et dans ce cas-là, le joueur demande au joueur précédemment le mot auquel il pensait, et il y a trois possibilités :
    - S'il renseigne un mot qui ne commence pas par les mêmes lettres que le mot formé dans la partie, alors le joueur précèdent obtient un quart de singe
    - Si le mot saisi existe dans le dictionnaire, alors le joueur qui a saisi ' ?' obtient un quart de singe
    - Si le mot saisi n'existe pas dans le dictionnaire, alors le joueur précèdent obtient un quart de singe

Il y avait aussi des consignes à respecter, notamment au niveau de l'affichage du jeu, qui est très précis. En effet, le jeu doit être construit de manière à être le plus compréhensible possible. L'affichage se déroule de la manière suivante :

L'utilisateur doit saisir un nombre de joueurs qu'il précise avec la nature 'H' pour *humain* et 'R' pour *robot*. En fonction des joueurs saisi, le programme va les identifier : leur numéro suivi de leur nature. Par exemple si l'utilisateur a saisi 'HRH', alors la partie sera composée de 3 joueurs, le premier est un humain et sera nommé 1H, le deuxième est un robot nommé 2R, et la troisième 3H.

Graphe de dépendance des fichiers sources



Code source des tests unitaires :

Pour vérifier et tester notre programme, nous avons créé un code de test unitaire pour :

1) Le dictionnaire (dico.cpp et dico.h – voir fichier test_unitaire_dico.cpp dans le répertoire tests_unitaires) :

```
int main() {

  //Création du dictionnaire
  Dico dico;
  remplirDico(dico, "test_Dico.txt");

  //Vérification du remplissage du dictionnaire
  // (qui contient des sauts de ligne, tabulation, ou espaces entre les mots
  // et des mots en minuscules et majuscules)
  assert(dico.tailleDico == 5);
  assert(strcmp(dico.mots[0], "AUREVOIR") == 0);
  assert(strcmp(dico.mots[1], "BONJOUR") == 0);
  assert(strcmp(dico.mots[2], "HELLO") == 0);
  assert(strcmp(dico.mots[3], "WORD") == 0);
  assert(strcmp(dico.mots[4], "ZOO") == 0);
  char mot[MAX_CHAR] = "Bonjour";
  unsigned int valeur = trouverMotDansDico(dico, mot);
  assert(valeur == 1);

  desallouerDico(dico);
  assert(dico.mots == NULL);
}
```

2) Le déroulement de la partie (partie.cpp et partie.h – voir fichier test_unitaire_partie.cpp dans le répertoire tests_unitaires) :

```
int main() {

  Partie p;
  char joueurs[MAX_CHAR] = "hhhh";
  // on teste avec une partie de 4 humains
  init(p, joueurs);

//Vérification des paramètres de la partie et des fonctions utiles au bon déroulement
//de la partie

  assert(p.nbJoueurs == strlen(joueurs));
  assert(p.taillemot == 0);
  assert(p.nouveauTour == FAUX);
  assert(p.dernierJoueurPerdant == -1);

  for (unsigned int i = 0; i < p.nbJoueurs; i++) {
    assert(p.score[i] == 0.);
    assert(p.ordre[i] == toupper(joueurs[i]));
  }
  int valeur3 = ajoutLettre(p, '!');
  assert(valeur3 == 2);
  valeur3 = ajoutLettre(p, '?');
  assert(valeur3 == 1);

  for (int i = 0; i < MAX_LENMOT; i++) {
    ajoutLettre(p, 'A' + i);

  for (unsigned int i = 0; i < MAX_LENMOT; i++) {
    assert(p.mot[i] == 'A' + i);
  }
```

```
  unsigned int valeur1 = verifMot(p, "test");
  assert(valeur1 == 1);

  for (unsigned int i = 0; i < 4; i++) {
    p.score[1] += 0.25;
  }
  unsigned int valeur2 = finduJeu(p);
  assert(valeur2 = 1);

  strcpy(p.motJoueurDeviner, "test");
  resetMotJoueur(p);
  for (unsigned int i = 0; i < MAX_LENMOT; i++) {
    assert(p.motJoueurDeviner[i] == CHAINE_VIDE);
  }

  detruireJeu(p);
  assert(p.ordre == NULL);
  assert(p.score == NULL);
}
```

3) Les robots (robot.cpp et robot.h – voir fichier test_unitaire_robot.cpp dans le répertoire tests_unitaires) :

```
int main() {

  char joueurs[MAX_CHAR] = "rrr";
  //on teste la partie avec 3 robots

  //Initialisation d'un dictionnaire et d'une partie pour tester les fonction utiles
  //aux robots
  Partie p;
  Dico dico;
  remplirDico(dico, "test_Dico.txt");

  init(p, joueurs);

  ajoutLettreRobot(p, 'A');
  assert(p.taillemot == 1);
  assert(strcmp(p.mot, "A") == 0);
  assert(p.motRobot[0] == CHAINE_VIDE);
  int indexMot = 0;
  unsigned int valeur1 = ChercherMotDico_Robot(dico, p, indexMot);
  assert(valeur1 == 0);
  assert(indexMot == 0);

  char* listeLettre = new char[MAX_CHAR];
  resetListeLettre(listeLettre);

  for (unsigned int i = 0; i < MAX_CHAR; i++) {
    assert(listeLettre[i] == CHAINE_VIDE);
  }

  unsigned int valeur3 = VerifLettreUtil('A', listeLettre);
  assert(valeur3 == 0);

  unsigned int valeur2 = AjoutLettreUtil('A', listeLettre);
  valeur3 = VerifLettreUtil('A', listeLettre);
  assert(valeur2 == 0);
  assert(valeur3 == 1);

  valeur2 = AjoutLettreUtil('A', listeLettre);
  assert(valeur2 == 1);
  assert(valeur3 == 1);
}
```

Bilan du projet :

Pour un premier projet, nous sommes satisfaits car nous avons finalisé et accompli tous les objectifs demandés. Nous avons rencontré des difficultés au niveau de la compréhension du sujet, notamment au début avec les structures et également au niveau de l'affichage des moyennes.

Mais nous avons su nous adapter, cette adaptation provient des cours en amphithéâtre, des travaux pratiques et dirigés dans lesquels certains exercices, exemples ainsi que les conseils et l'aide des enseignants permettaient une meilleure compréhension des énoncés.

De ce fait, le sujet a été approprié et par conséquent a pu nous apporter énormément d'idées que nous avons par la suite intégré au code, par exemple la création d'une commande ''help'' qui indique la chaine de caractères à renseigner en cas d'erreur d'entrée.