Licence 3 d'Informatique
Programmation Avancée en C

Le Devin

Céline NOËL
Eric DIALLO

I- Présentation du projet

Ce projet réalisé au terme de notre cinquième semestre de Licence Informatique est un petit jeu style Akinator. Le devin (oui, il a une sale tête) doit tenter de deviner le personnage auquel l'utilisateur pense. Pour celà, il lui pose une série de vingt questions fermées (oui/non). L'utilisateur peut alors répondre de cinq manières différentes: “sûrement”, “probablement”, “peut-être”, “probablement pas” et “sûrement pas”. En utilisant les réponses fournies, le devin doit alors être en mesure de fournir une hypothèse quand au personnage choisi par l'utilisateur. S'il se trompe, il demande à l'utilisateur de rentrer son personnage, ainsi qu'une question qui aurait permis de l'identifier. On peut alors recommencer une nouvelle partie.

II- Mise en oeuvre

1°) *Gestion de la base de données*

Le programme a besoin de deux types de fichiers pour fonctionner: un fichier questions.txt, et un fichier population.txt, tous deux devant être placés dans le répertoire data/ du projet.

Le fichier questions.txt contient la liste des questions que le devin est susceptible de poser à l'utilisateur. Afin de respecter le format de lecture, ce fichier doit contenir une et une seule question par ligne.

Le fichier population.txt contient le nom des différents personnages connus par le programme, ainsi que la liste des réponses aux questions pour chacun. Le fichier est organisé de telle sorte que pour chaque personnage, une ligne soit occupée par son nom, et la suivante par la liste des réponses aux questions posées, terminée par -1. Les réponses sont des entiers compris entre 0 et 5 tels que 1 correspond à “sûrement”, …, 5 à “sûrement pas”, et 0 signifie que la réponse n'a pas encore été renseignée.

Le programme est capable d'étendre lui-même sa base de données. En effet, à la fin d'une partie, si le personnage n'a pas été trouvé, alors le programme va l'ajouter à la base, ainsi que la question qui aura été demandée à l'utilisateur (il est possible que des questions se retrouvent en double dans la base, mais celà ne modifie pas forcément l'efficacité du programme). Enfin, si au contraire, le personnage a été découvert et que certaines de ses réponses n'étaient pas renseignées, le programme pourra modifier leurs valeurs s'il a posé les questions correspondantes durant la partie (si à un moment, le devin est certain du personnage choisi, il posera de préférence les questions dont ses réponses n'ont pas été renseignées).

2°) *Choix des questions et détermination du personnage*

Une fois la base de données chargée par le programme, chaque personnage se voit attribuer une note (the higher, the better). Pour chaque question posée, si la réponse d'un personnage est éloignée de celle de l'utilisateur, sa note diminue, si au contraire, elle en est proche, le personnage gagne des points. Si la note d'un personnage descend trop bas, le personnage est éliminé. Si elle dépasse largement les notes attribuées aux autres, le programme considère qu'il a déterminé le bon personnage.

Afin de poser les bonnes questions et d'éliminer un maximum de possibilités, le programme attribue également des notes aux questions. Une question est bien notée si les réponses des personnages sont équiréparties (à peu près autant de réponses “sûrement”, que de réponses “probablement”, …, que de réponses “sûrement pas”). Celà permet en effet d'être sûr d'éliminer beaucoup de personnages.

3°) *Interface graphique*

Lorsqu'on lance le programme, l'interface graphique charge en mémoire les ressources nécessaires à l'affichage, puis affiche un menu. Ce menu propose à l'utilisateur de faire une partie, ou de quitter le programme.
Lorsqu'il décide de jouer, l'interface graphique affiche les images du devin, d'une bulle ou sont affichées les différentes questions, ainsi que des boutons de réponses possibles.
Lorsque le programme pense avoir trouvé le personnage auquel l'utilisateur pense, l'interface demande si le personnage est bien celui – ci et affiche 2 boutons : Oui et Non.
Si le programme s'est trompé, l'interface propose à l'utilisateur d'ajouter son personnage avec une question ainsi qu'une réponse correspondante.

Cette interface a été codée à l'aide de la librairie MLV. Les différentes images utilisées ont été dessinées à l'aide du logiciel GIMP. Les textes sont affichés avec la police d'écriture Journal.ttf .

III- Gestion du projet

1°) *Répartition des tâches*

Très globalement :
Céline => Population, Question, Character, Main
Eric => GUI, String, Stream, DBManager

Néanmoins, afin de tester le programme et de l'améliorer, nous avons tous les deux modifier des parties de chacun des fichiers. Par conséquent, chacun de nous a compris ce que l'autre a fait et est capable d'expliquer les bouts de code codés par le binôme.

2°) *Gestionnaire de versions*

Nous avons utilisé le server svn de l'université afin de garder le code à jour (et parce que c'était imposé, parce qu'autrement, git, c'est quand même plus pratique quand on code sans connexion).

3°) *Arborescence*

mydevin > data (images + population.txt + questions.txt)
doc (rapport + fichiers doxygen)
include (headers)
src (sources)
Makefile (makefile pour builder le projet)
svn.log (les logs des commits, générés avec `svn log > svn.log`)

Les répertoires bin/ et obj/ sont générés directement par le Makefile, merci donc de ne pas nous retirer de points si vous ne les voyiez pas ! :B