Intelligence Artificielle

PROJET - Rapport

1.12.2018

Membres :

Antoine BOUQUET
Alice CANTAGREL

Sommaire

I. Introduction

A. Objectif

L'objectif de ce projet est de créer "un programme informatique permettant de tester et d'évaluer de façon automatique les connaissances d'une personne dans le domaine de l'intelligence artificielle, sur la base des enseignements du module" IA de deuxième année à l'ENSC.

B. Cahier des charges

Pour réaliser ce projet, nous devions suivre les instructions suivantes :
→ Le programme doit être réalisé en C# et utiliser la technologie WinForms.
→ Le programme pose à l'utilisateur une liste de questions prédéfinies et stockées sous format xml.
→ Le programme doit pouvoir poser 20 questions de façon aléatoire
→ Le programme permet d'évaluer les réponses données par l'utilisateur, d'attribuer une note et d'afficher les corrections éventuelles.
→ Le programme doit pouvoir évaluer les connaissances de l'utilisateur sur la mise en oeuvre de l'algorithme de Dijkstra.

II. Conception et Réalisation

A. Choix de conception

1. Réflexion

Pour mener à bien ce projet, nous avons fait plusieurs choix de conceptions.

Tout d'abord, nous avons choisi de répartir le programme sur quatre formulaires différents : un premier formulaire de présentation qui explique le contexte du test, un deuxième formulaire sur lequel les questions et leurs réponses vont s'afficher au fur et à mesure du test, un troisième formulaire où l'utilisateur doit faire tourner l'algorithme Dijkstra à la main et remplir les noeuds de l'arbre et enfin un quatrième formulaire qui permet d'afficher la note obtenue ainsi qu'un récapitulatif des questions posées, leur réponse juste et la réponse donnée par l'utilisateur. Nous avons par conséquent choisi d'afficher la correction à la fin du test et non entre chaque question car la correction d'une question pourrait influencer la réponse de l'utilisateur pour une autre question.

Pour ce qui est du contenu du troisième formulaire, nous avons choisi de faire entrer à l'utilisateur la liste des ouverts et la liste des fermés étape après étape avec pour la notation de chaque noeud une séparation par seulement une virgule. Nous avons choisi de faire commencer l'utilisateur avec le noeud initial déjà présent dans l'ensemble des fermés. Lorsque celui-ci remplit les deux listes, il doit cliquer sur le bouton 'Suivant' pour passer à l'itération suivante. Un compteur d'itérations a alors été rajouté pour montrer à l'utilisateur que son clic a bien été pris en compte. Une fois que l'utilisateur a atteint le nombre maximal d'itérations possibles pour trouver le chemin le plus court entre deux points, le bouton 'Suivant' se désactive, indiquant à l'utilisateur de passer au remplissage de l'arbre. A ce moment là, la partie du remplissage de l'arbre devient ciblable. L'affichage de l'arbre se fait quant à lui à l'aide d'un treeview qui apparaît vide à l'utilisateur, celui-ci doit alors indiquer le numéro du noeud qu'il souhaite renseigner et cliquer dans le treeview sur le noeud qu'il souhaite modifier et enfin valider son action. Lorsqu'il a fini de remplir l'arbre, il peut cliquer sur le bouton 'Résultat' qui lui indique alors s'il a juste aux deux points précédents ou auquel/auxquels il a eu faux le cas échéant. Ne pas remplir une itération ou l'arbre est considéré comme une réponse fausse.

Concernant les questions à poser à l'utilisateur, nous avons décidé de ne poser que des questions type QCM car ce type de questions nous semblait le plus approprié à la technologie WinForms. Pour chaque question, nous avons décidé de proposer quatre réponses à l'utilisateur dont une et une seule réponse est correcte. De ce fait, chaque question admet une réponse juste. Une réponse juste rapporte un point à l'utilisateur et cela pour chaque question. L'exercice sur Dijkstra est quant à lui noté sur 3 points. L'utilisateur est noté sur 20 points au total : 17 questions lui sont donc posées puis en suivant un exercice sur Dijkstra.

2. Choix techniques

Le projet a été réparti en deux parties : la première partie concerne la passation d'un questionnaire type QCM pour évaluer les connaissances générales de l'opérateur en IA tandis que la deuxième partie se consacre à l'évaluation de la connaissance de la mise en oeuvre de l'algorithme de Dijkstra.

Pour les choix techniques de la **partie 1** de notre projet, nous avons choisi de structurer notre programme comme suit :

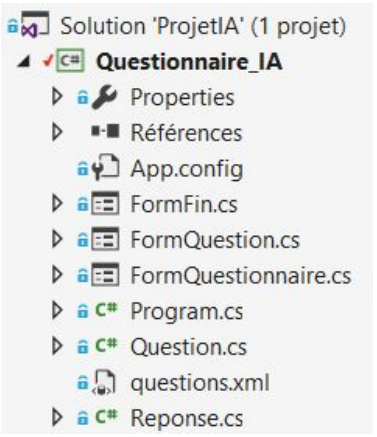

image 1 : Structure de la partie 1 du programme

Nous retrouvons les trois formulaires évoqués plus haut. Comme indiqué sur l'image 1, nous avons choisi de créer deux classes : une classe Question et une classe Reponse ce qui nous permet ensuite de pouvoir créer des listes de ces éléments.

Concernant la **partie 2** de notre projet, nous avons intégré à notre solution le formulaire FormDijkstra ainsi que les classes SearchTree, GenericNode et sa classe fille Node2 qui nous permettent de faire les traitement nécessaires dans le réseau de noeuds. Il y a aussi un fichier texte "graphe1.txt" avec le schéma d'un graphe. Ce graphe est celui proposé lors du déroulement de l'algorithme Dijkstra. Il peut être modifié directement à partir du fichier texte permettant de mettre le graphe souhaité dans l'application.

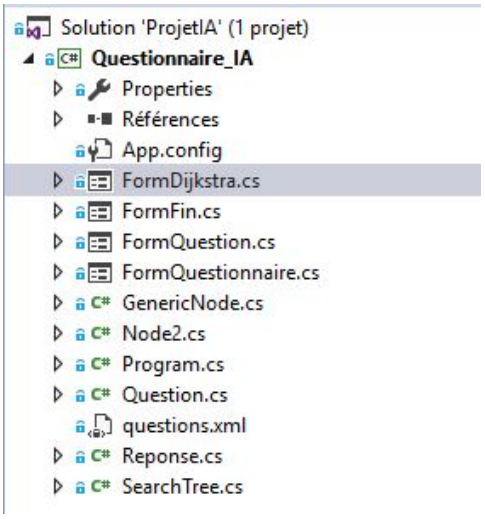

image 2 : Structure du programme complet

B. Réalisation

Voici quelques captures d'écran montrant les différentes interfaces de notre projet :

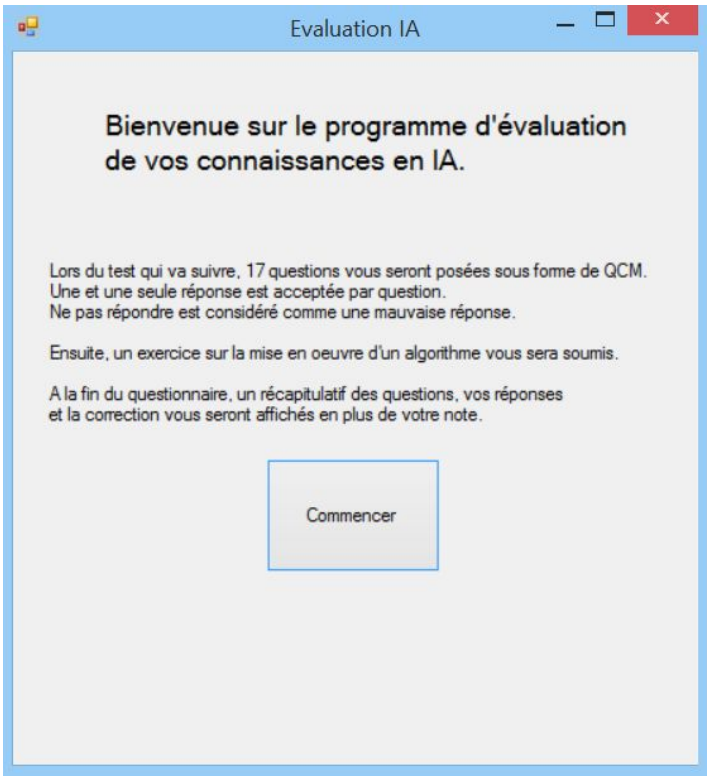

image 3 : Formulaire de lancement

Tout d'abord, un écran permet de présenter le contexte du test et de lancer la phase de questionnaire après un clic sur le bouton "Commencer".

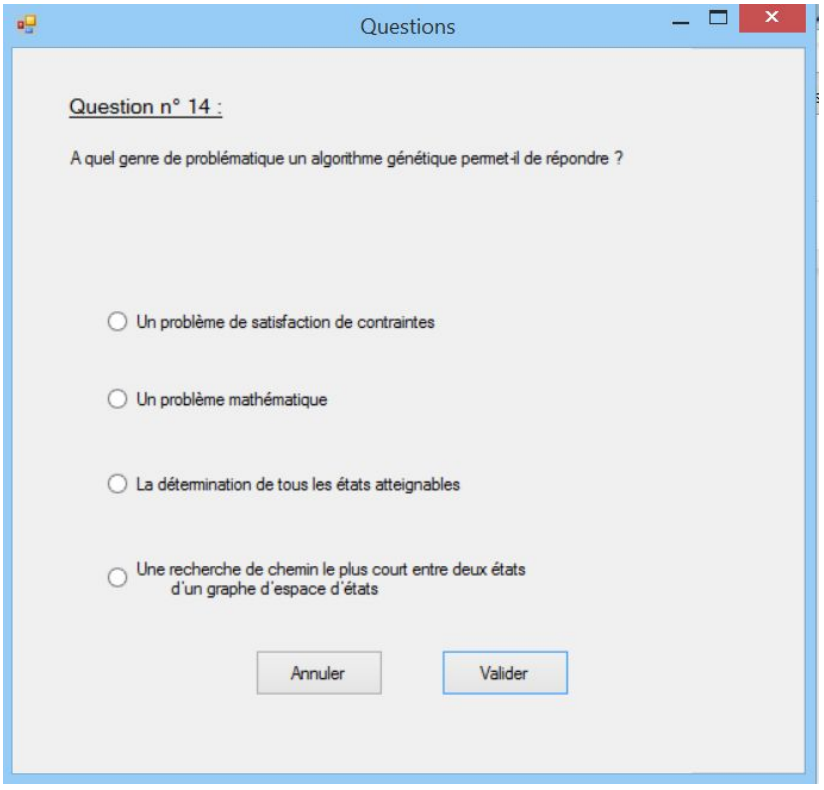

image 4 : Exemple de question

L'image 4 présente un exemple de question posée. En haut à gauche, l'utilisateur peut voir à quelle question il en est. En dessous, l'intitulé de la question lui est posé. Ensuite les quatres réponses lui sont proposées. Un bouton 'Annuler' permet d'effacer le choix fait tandis que le bouton 'Valider' permet d'enregistrer la réponse donnée et de passer à la question suivante. Aucun retour en arrière n'est possible.

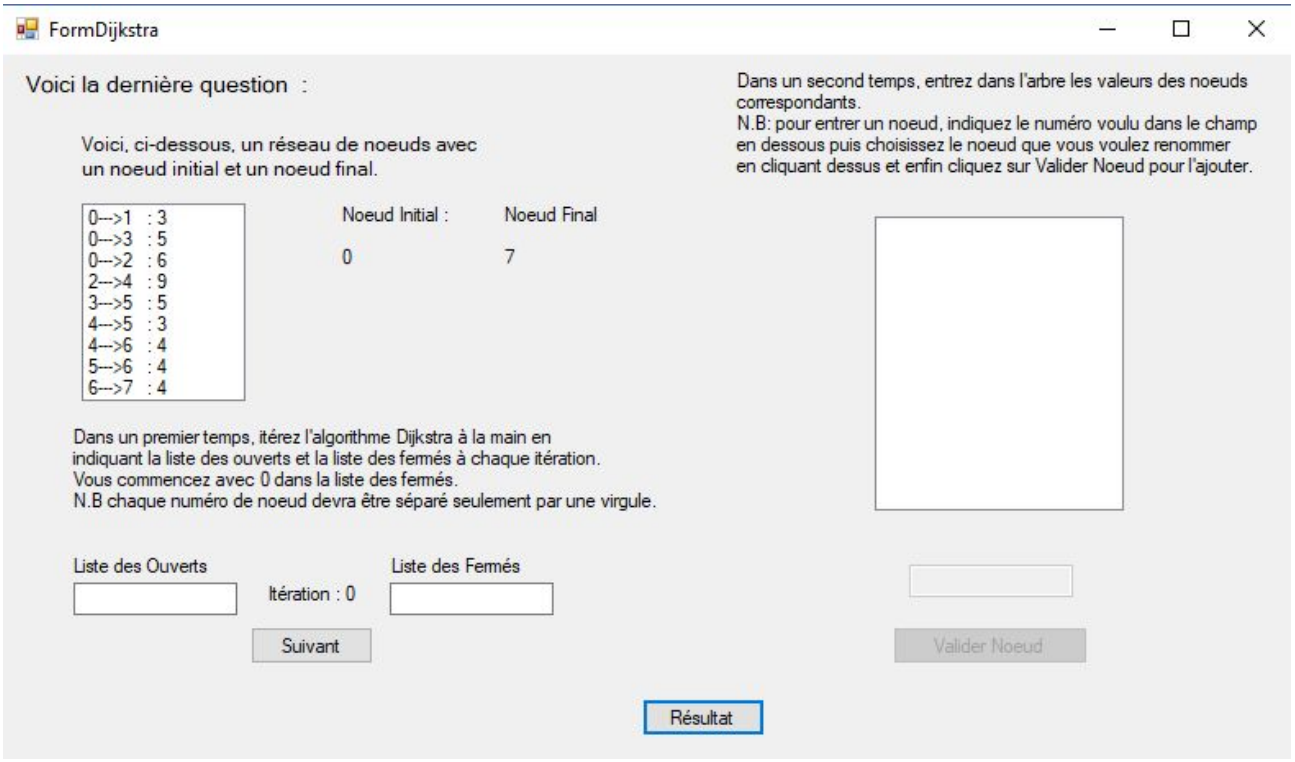

image 5 : Formulaire Dijkstra avant l'entrée des ouverts et des fermés

L'exercice sur Dijkstra se pose comme montré dans l'image 5. La box à gauche permet de présenter les valeurs des distances entre les différents noeuds. Ces valeurs sont initialisées à partir des données stockées dans le fichier "graphe1.txt" sous la forme suivante :

```
nombre de noeuds : 8
arc1 : 0 1 3
arc2 : 0 3 5
arc3 : 0 2 6
arc4 : 2 4 9
arc5 : 3 5 5
arc6 : 4 5 3
arc7 : 4 6 4
arc8 : 5 6 4
arc9 : 6 7 4
```

image 6 : Exemple d'arbre proposé pour l'exercice de Dijkstra

Chaque arc représente une relation entre deux noeuds de l'arbre : les deux premières valeurs indiquées sont les deux noeuds en relation et la troisième représente la distance entre ces deux noeuds.

Deux textBox permettent de rentrer les valeurs de l'ensemble des ouverts et l'ensemble des fermés à chaque itération. Le bouton 'Suivant' permet d'itérer l'algorithme de Dijkstra qui vérifie les données rentrées par l'utilisateur. La partie de droite devient modifiable une fois que les deux ensembles ont été remplis. Une textBox permet de rentrer les valeurs à écrire dans l'arbre tandis que le bouton 'Valider Noeud' permet l'écriture dans l'endroit sélectionné de l'arbre.

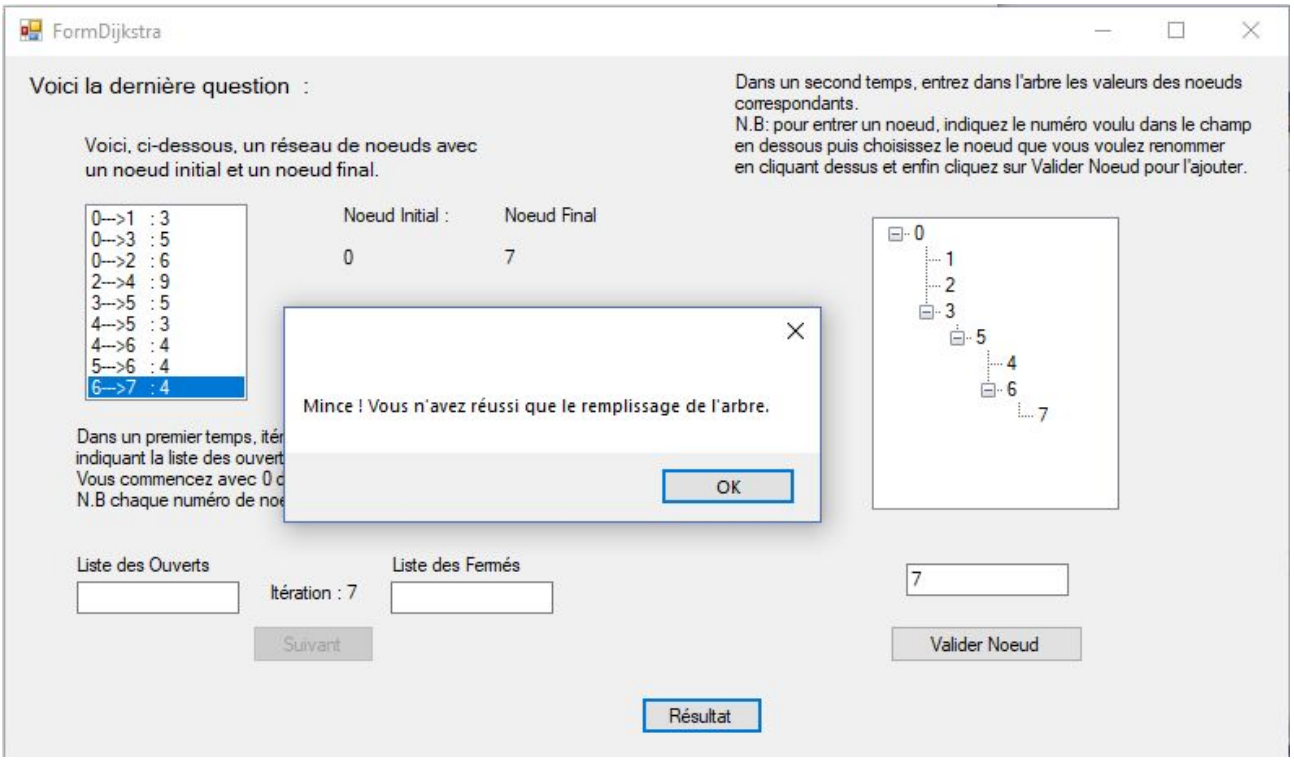

image 7 : Message de retour sur l'algorithme de Dijkstra

Ici, l'arbre a été complété correctement mais pas l'ensemble des listes : l'opérateur n'aura qu'un point sur les trois de l'exercice.

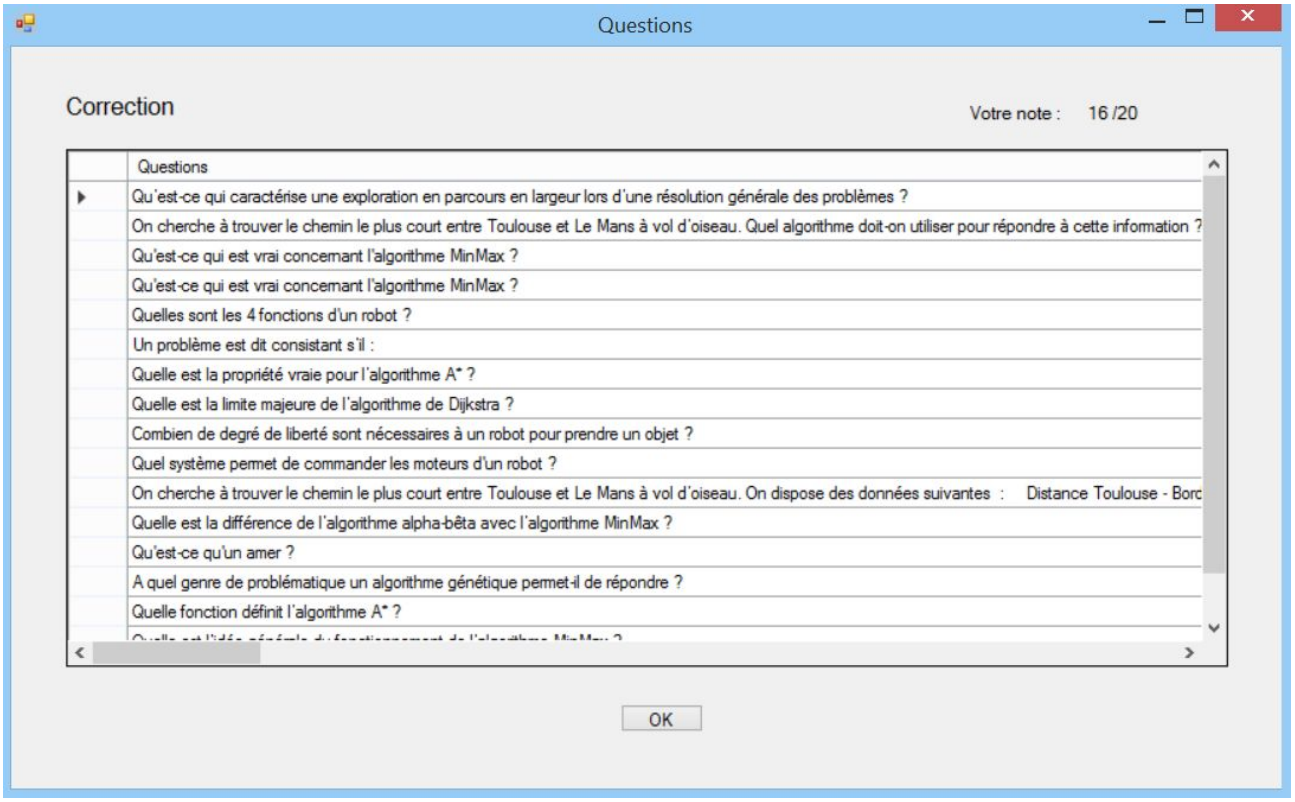

image 8 : Formulaire de correction

En haut à gauche, la note de l'opérateur est affichée. En dessous, l'ensemble des intitulés des questions posées est rappelé.

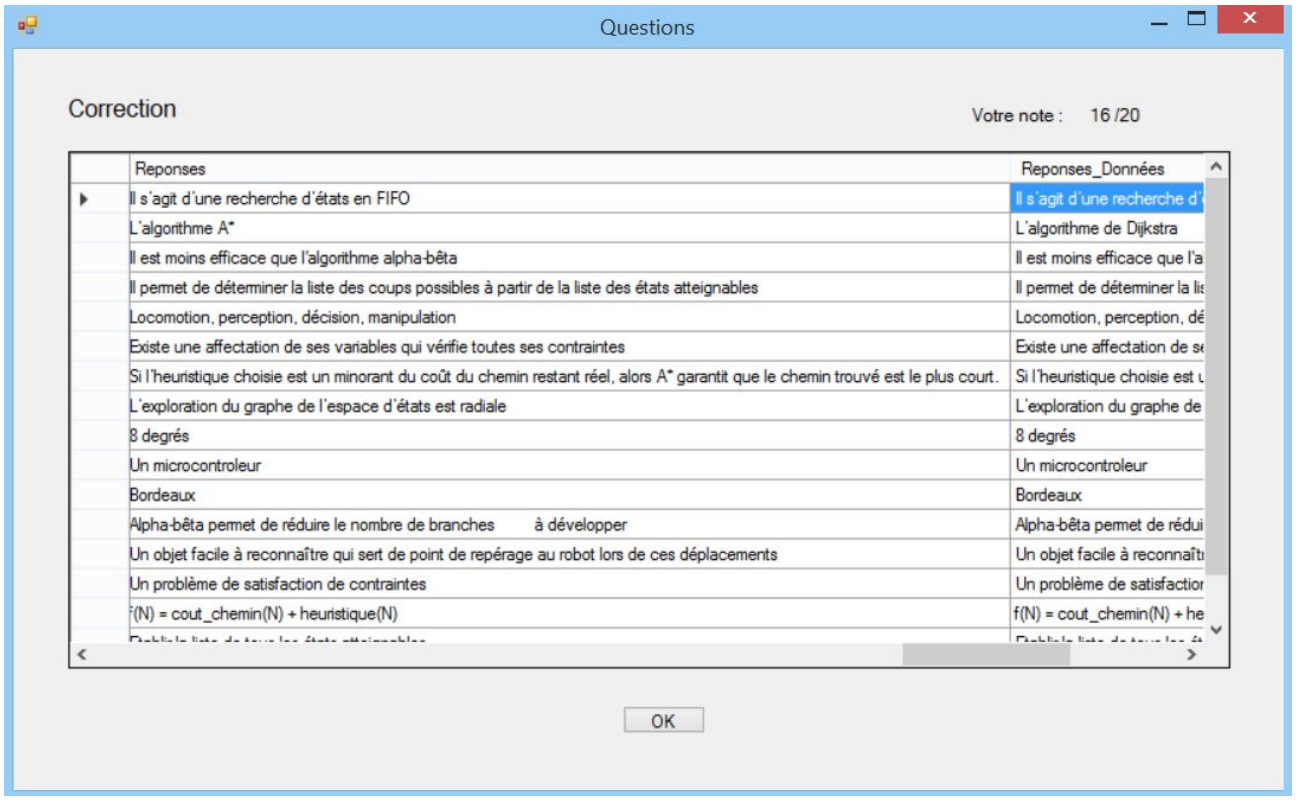

image 9 : Formulaire de correction

Comme on peut le voir sur l'image 9, le formulaire de correction affiche aussi les réponses justes (Reponses) et les réponses données par l'utilisateur (Reponses_Donnees). Si l'utilisateur n'a pas sélectionné de réponse avant de valider, le formulaire de correction lui indique qu'il n'a pas répondu et aucun point ne lui est attribué (image 10).

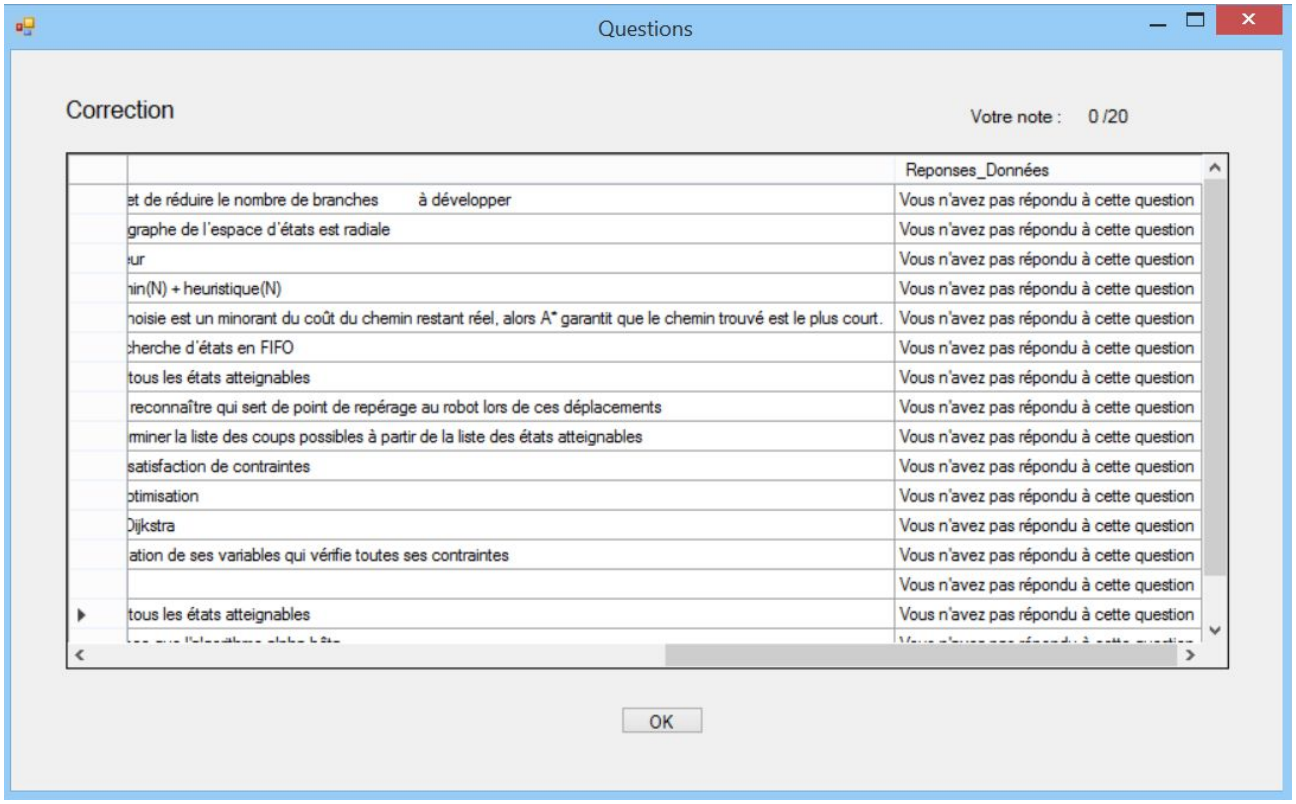

image 10 : Formulaire de correction quand l'utilisateur n'a pas répondu

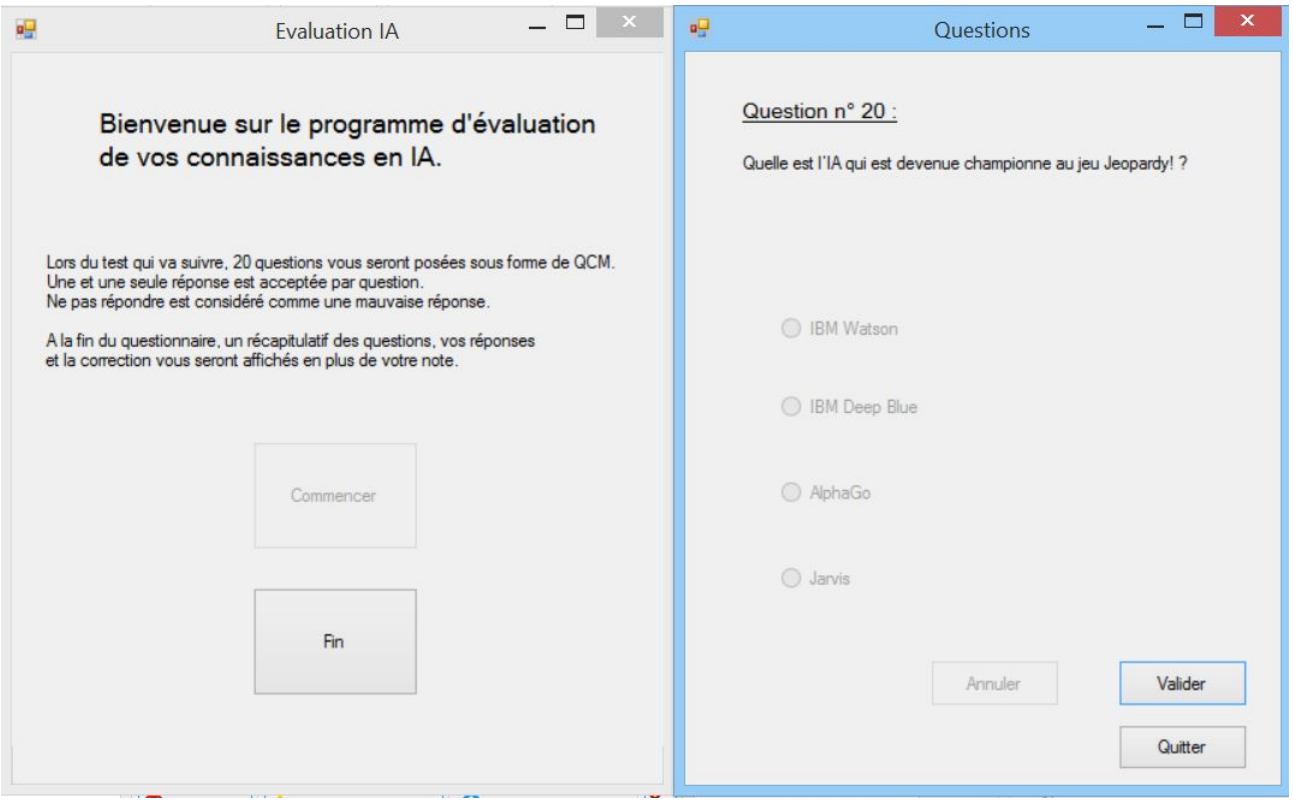

image 11 : Incitations à quitter

Lorsque les 17 questions sont posées, les interactions de l'utilisateur avec les interfaces sont restreintes et des boutons "Quitter" apparaissent pour indiquer à l'utilisateur qu'il a fini de réaliser les actions possibles.

III. Gestion de projet

A. Outils utilisés

Pour mener à bien ce projet, nous avons utilisé différents outils :

- Google Drive, pour le partage des documents relatifs au projet ainsi que nos idées;
- Microsoft Visual Studio, pour le développement du programme;
- GitHub et GitKraken, pour le partage et le versionning du code.

B. Répartition du travail

Pour travailler efficacement, nous nous sommes répartis le travail comme suit :

Partie 1	Antoine	Alice
Désérialisation	X	
Recherche des questions	X	X
Interfaces		X
Programme		X
Partie 2	**Antoine**	**Alice**
Interfaces	X	X
Programme	X	
Rapport	**Antoine**	**Alice**
Rédaction	X	X

De plus, chacun des membres du projet a suivi le déroulement de la partie sur laquelle il a moins travaillé. Une revue du code dans son intégralité a été effectuée ensemble.

IV. Bilan

A. Limites / Difficultés

La limite principale de notre programme est le remplissage du TreeView qui est pénible et non intuitif : il faut expliquer à l'utilisateur la démarche pour rentrer le nom d'un noeud dans l'arbre (écrire le nom du noeud dans la textbox, cliquer sur le noeud à modifier et appuyer sur valider pour effectivement écrire dans l'arbre).

Nous n'avons pas rencontrés de difficultés particulières au cours de ce projet.

B. Améliorations possibles

Comme amélioration de ce programme, plusieurs éléments pourraient être rajoutés. Tout d'abord, un nombre de questions plus élevé permettrait de varier plus les questions. Ensuite, des questions plus complexes pourraient être rajoutées : dans ce cas, leur notation serait alors plus élevée. On pourrait aussi envisager de tester la mise en oeuvre des différents algorithmes étudiés en cours d'IA par l'utilisateur. Enfin, comme dit précédemment, le remplissage du treeview pourrait être amélioré et rendu plus intuitif pour l'utilisateur.

C. Apports

Ce projet nous a permis de développer nos compétences en terme de manipulation de la technologie WinForm. Nous avons aussi pu consolider nos apprentissages en IA.

V. Annexes

A. Questions posées et leurs réponses

1) Quelles sont les 4 fonctions d'un robot ?
   a) Locomotion, perception, décision, manipulation (true)
   b) Communication, action, réaction, décision
   c) Manipulation, localisation, réflexion, action
   d) Déplacement, réaction, localisation, perception

2) Quel système permet de commander les moteurs d'un robot ?
   a) Un microprocesseur
   b) Un servocontroleur
   c) Un microcontroleur (true)
   d) Un processeur

3) Qu'est-ce qu'un amer ?
   a) Une action qu'un robot ne peut pas effectuer dans un cas précis
   b) Une tâche pour laquelle un robot n'a pas été conçu
   c) Un objet facile à reconnaître qui sert de point de repérage au robot lors de ces déplacements (true)
   d) Un endroit inconnu pour un robot

4) Combien de degré de liberté sont nécessaires à un robot pour prendre un objet ?
   a) 3 degrés
   b) 5 degrés
   c) 8 degrés (true)
   d) 9 degrés

5) Qu'est-ce qui se rapporte au concept d'IA forte ?
   a) Une intelligence artificielle peut être autant consciente et intelligente que l'Homme (true)
   b) Une intelligence artificielle ne peut pas être consciente et intelligente comme l'Homme
   c) Une intelligence artificielle peut être plus intelligente que l'Homme
   d) Une intelligence artificielle peut en théorie être consciente et avoir une intelligence semblable à celle de l'Homme mais en pratique, cela n'est pas concevable

6) Qu'est-ce qui est vrai concernant l'algorithme MinMax ?
   a) Il est plus efficace que l'algorithme alpha-bêta
   b) Il permet de déterminer la liste des coups possibles à partir de la liste des états atteignables (true)
   c) Il permet de développer un nombre minimal de branches
   d) Il permet d'établir le nombre maximal d'états existants

7) Qu'est-ce qui est vrai concernant l'algorithme MinMax ?
   a) Il est plus efficace que l'algorithme alpha-bêta
   b) Il est plus efficace qu'un algorithme génétique
   c) Il est moins efficace qu'un algorithme génétique
   d) Il est moins efficace que l'algorithme alpha-bêta (true)

8) Pour créer une IA capable de jouer aux échecs, je dois utiliser :
   a) l'algorithme MinMax (true)
   b) L'algorithme génétique
   c) L'algorithme A*
   d) L'algorithme de Dijkstra

9) Quelle est la différence de l'algorithme alpha-bêta avec l'algorithme MinMax ?
   a) Il n'y a pas de différence
   b) Alpha-bêta permet d'atteindre plus d'états que MinMax
   c) Alpha-bêta permet de développer plus de branches que MinMax

d) Alpha-bêta permet de réduire le nombre de branches à développer (true)

10) Quelle est l'idée générale du fonctionnement de l'algorithme MinMax ?
    a) Etablir la liste des états atteignables à l'action suivante
    b) Etablir la liste de tous les états atteignables (true)
    c) Etablir le nombre maximal d'états existants
    d) Etablir le nombre minimal d'états existants

11) Quelle est l'idée générale du fonctionnement de l'algorithme alpha-bêta ?
    a) Etablir le nombre maximal d'états existants
    b) Etablir la liste des états atteignables à l'action suivante
    c) Etablir la liste de tous les états atteignables (true)
    d) Etablir le nombre minimal d'états existants

12) Qu'est-ce qui caractérise une exploration en parcours en largeur lors d'une résolution générale des problèmes ?
    a) Il s'agit d'une recherche d'états en FIFO (true)
    b) Il s'agit d'une recherche d'états en FIFI
    c) Il s'agit d'une recherche d'états en LIFO
    d) Il s'agit d'une recherche d'états en LIFI

13) Quel algorithme permet de trouver le plus court chemin entre un état initial et un état final ?
    a) L'algorithme MinMax
    b) L'algorithme de Dijkstra (true)
    c) L'algorithme alpha-bêta
    d) L'algorithme génétique

14) Quelle est la limite majeure de l'algorithme de Dijkstra ?
    a) Il n'y a pas de limite
    b) Le graphe de l'espace d'états est infini
    c) Atteindre chaque état de l'espace d'états demande un temps conséquent
    d) L'exploration du graphe de l'espace d'états est radiale (true)

15) Quelle fonction définit l'algorithme A* ?
    a) f(N) = cout_chemin(N) - heuristique(N)
    b) f(N) = cout_chemin(N) * heuristique(N)
    c) f(N) = cout_chemin(N) + heuristique(N) (true)
    d) f(N) = cout_Chemin(N) + heuristique(N+1)

16) On cherche à trouver le chemin le plus court entre Toulouse et Le Mans à vol d'oiseau. Quel algorithme doit-on utiliser pour répondre à cette information ?
    a) L'algorithme MinMax
    b) L'algorithme alpha-bêta
    c) L'algorithme de Dijkstra
    d) L'algorithme A* (true)

17) Quelle est la propriété vraie pour l'algorithme A* ?
    a) Si l'heuristique choisie est un majorant du coût du chemin restant réel, alors A* garantit que le chemin trouvé est le plus court.
    b) Si l'heuristique choisie est un minorant du coût du chemin restant réel, alors A* garantit que le chemin trouvé est le plus court. (true)
    c) Si l'heuristique choisie est un majorant du coût du chemin parcouru, alors A* garantit que le chemin trouvé est le plus court.
    d) Si l'heuristique choisie est un minorant du coût du chemin parcouru, alors A* garantit que le chemin trouvé est le plus court.

18) A quel genre de problématique un algorithme génétique permet-il de répondre ?
    a) Une recherche de chemin le plus court entre deux états d'un graphe d'espace d'états
    b) La détermination de l'ensemble des états atteignables à partir d'un moment donné
    c) Un problème de logique
    d) Un problème d'optimisation (true)

19) A quel genre de problématique un algorithme génétique permet-il de répondre ?
    a) Un problème de satisfaction de contraintes (true)

b) Un problème mathématique
c) La détermination de tous les états atteignables
d) Une recherche de chemin le plus court entre deux états d'un graphe d'espace d'états

20) On cherche à trouver le chemin le plus court entre Toulouse et Le Mans à vol d'oiseau. On dispose des données suivantes :
Distance Toulouse - Bordeaux par la route = 245 km | Distance Toulouse - Clermont-Ferrand par la route = 376 km
Distance Toulouse - Paris par la route = 681 km | Distance Toulouse - Marseille par la route = 404 km
Distance Toulouse - Le Mans par la route = 686 km | Distance Bordeaux - Le Mans à vol d'oiseau = 357 km
Distance Clermont-Ferrand - Le Mans à vol d'oiseau = 330 km | Distance Paris - Le Mans à vol d'oiseau = 185 km
Distance Marseille - Le Mans à vol d'oiseau = 660 km | Distance Toulouse - Le Mans à vol d'oiseau = 499 km
Par quelle ville faut-il passer ?
a) Bordeaux (true)
b) Clermont-Ferrand
c) Paris
d) Marseille

21) Un problème est dit consistant s'il :
a) N'existe pas de solution(s) à ce problème
b) Est possible d'affecter toutes les variables du problème
c) Existe une affectation de ses variables qui vérifie toutes ses contraintes (true)
d) Existe une variable qui vérifie toutes ses contraintes

22) Quelle est l'IA qui est devenue championne au jeu Jeopardy! ?
a) IBM Watson (true)
b) IBM Deep Blue
c) AlphaGo
d) Jarvis

23) Parmi ces différentes réponses, laquelle ne représente pas une difficulté pour le traitement automatique du langage naturel ?
a) La sémantique
b) La grammaire et la conjugaison
c) Le pluriel de mots (true)
d) Les expressions idiomatiques