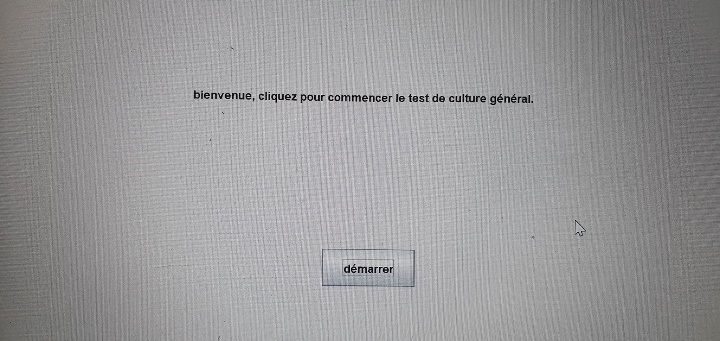
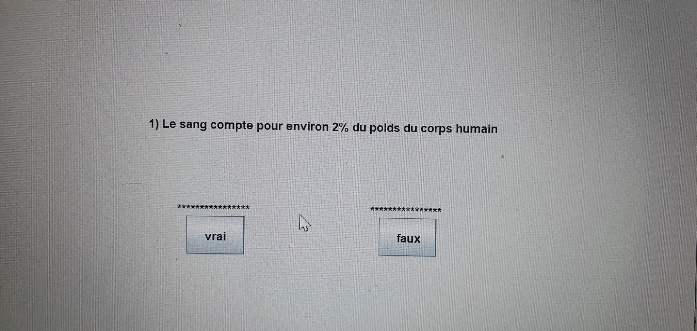
1. **Analyse et conception**

A.1 Description du sujet

Mon projet est un petit jeu ayant pour but d’aider une personne âgée atteinte de la maladie d’alzheimer à entrainer sa mémoire à court terme. Pour cela j’ai voulu créer un petit quiz de 10 questions. Le but de ce jeu est de répondre au question par la bonne réponse et ce plusieurs fois jusqu’à ce qu’il est 100% de bonnes réponses.

A.2 Cahier des charges

Le but est de faire l’application la plus simple visuellement parlant. Etant destinée à des personnes âgées ce petit jeu doit être le plus intuitif possible sans question possible sur la manière dont l’utilisateur doit faire. Donc nous aurons 1 à 2 boutons avec une question.  

A.3 Diagramme de classe UML

N’ayant eu besoin que d’une seule classe je n’ai pas su faire de diagramme étant donné que le but de ce diagramme est de montrer les interactions entre les différentes classes du programme.

A.4

Je n’ai pas trouvé d’autre moyen que de faire une seule classe afin de réaliser un projet qui correspondait au résultat final escompté.

A.5 Architecture MVC

Le pattern MVC a été implémenté de façon très basique car presque tous se passe au visuel et presque rien en arrière plan. Les bouton vrai ou faux font aller d’un endroit à un autre mais qui garde la même fonction que l’on clique sur l’un ou sur l’autre.

A.6 Collection

Je n’ai pas recueillis directement de données, cependant j’utilise ici un tableau afin d’avoir d’y stocker et récupérer mes questions et les valeurs attendues.

A.7 Interface graphique

L’écran devait être le plus simple possible j’ai donc au maximum 3 éléments en même temps sur l’écran. 2 JButton et 1 JLabel les JButtons étant les entrées vrai ou fausse, et le JLabel étant la question.

1. **Implémentation et robustesse**

B.1 Gestions des erreurs

La seule erreur que je pense possible est si l’ont entre en console des valeurs non acquises. Mais alors la mon programmes verra que les valeurs rentrées ne sont pas bonnes et demandera juste à l’utilisateur en console de retaper une des valeurs proposées.

B.2 Tests unitaires

Pour les test unitaires les seuls qui j’ai effectués sont des test ou j’isolait tout le reste de la partie cible souvent en plaçant les parties non visées en commentaire. Ce qui m’a permis de tester chaque fenêtre et chaque fonction de mon code indépendamment du reste.

B.3 Git

B.4 Propreté du code

Pour la structure du code j’ai utilisé un site : (<http://prettyprinter.de/index.php>). Pour le choix des variables j’ai essayer de faire au plus logique et de les appelées soit par leurs but soit par ce qu’elles allaient contenir.

1. **Documentation**

C.1 Javadoc

C.2 Manuel d’installation

Il faut que je télécharge mon document sur son ordinateur, que je me positionne dans le bon répertoire que j’effectue la commande :

Javac projet.java

1. **Bilan**

D.1 Conclusion sur le résultat obtenu

Dans l’ensemble général je pense que le code est convenable même si il y a autant de façon de programmer que de programmeurs… Pour les objectifs je suis mitigés . En effet certains des objectifs sont atteints mais pas tous, j’en était conscient tous le long de ce projet et j’ai chercher des alternatives mais rien. Page blanche.

D.2 Conclusion sur mes apprentissages

Ce projet était un premier pas dans le monde du langage Java, qui pour être honnête n’est pas mon préféré au vu de ca longueur/ complexité de programmation pour réaliser des tâches pourtant basiques. Mais j’en ressort avec pleins d’acquis sur ce langage et aussi sur l’univers des programmes/applications.

D.3 Points fort et pistes d’amélioration

Je pense que le plus gros points fort de ce projet est sa facilité d’exécution et d’utilisation au vue du public visé.

D.4 Bonus

Le seul bonus possible pour moi serait que le « client » est une personne âgée qui l’utilise régulièrement afin d’essayer de ralentir la progression de sa maladie.