BIRET-TOSCANO Esteban
FALETTI Léo
4a

SAE 01 : Implémentation d'un besoin client

Problématique : *Au département informatique, les enseignant.e.s souhaitent disposer d'une application qui tire au sort la prochaine personne interrogée, en respectant certaines contraintes.*

Le client (ici les enseignant.e.s) désire avoir la possibilité de lancer l'application via un terminal (>java.menu). Il souhaite également que l'application propose un menu qui permet le tirage au sort ou l'arrêt, ré affiché après chaque tirage ou action. L'application devra donc tirer au sort et afficher la prochaine personne interrogée, en respectant certaines contraintes que nous allons définir ci dessous.

Contraintes :

- un même étudiant ne pourra pas être affiché deux fois de suite.

- les étudiants avec le moins de passage auront une priorité absolu pour être désigné (si il y a 5 étudiants dans la promo, et que le numéro 2, 3 et 5 ont été tirés au sort, ils ne pourront plus être désigné tant que les étudiants numéro 1 et 4 ne seront pas passés).

- une fois que tous les étudiants de la promo ont été désignés, on propose à l'utilisateur si il désire recommencer un tour de la promo, ou si au contraire il souhaite s'arrêter là.
Naturellement, le dernier étudiant tiré au sort du tour de promo précédent ne pourra pas être le premier du nouveau tour de promo.

Démarche pour le menu :

Nous avons tout d'abord modifié le menu, en proposant plusieurs options :

- Si l'utilisateur saisit le chiffre 2, la liste des étudiants s'affiche, avec leur numéro respectif.

- Si l'utilisateur saisit le chiffre 1, un étudiant est tiré au sort et affiché.

- Si l'utilisateur saisit le chiffre 0, le programme s'arrête.

- Si l'utilisateur saisit un chiffre ou nombre différent de ces derniers, alors le programme lui affichera un message d'erreur, l'invitant à re-saisir un chiffre. Cette étape se répétera jusqu'à ce que l'utilisateur saisisse un chiffre correct (0, 1 ou 2).

Les options :

Au lancement du programme, si l'utilisateur saisit le nombre 2, la liste des étudiants (et leur numéro dans la promo) sera affichée, suivie du menu.
L'utilisateur peut donc utiliser cette option une infinité de fois.

Au lancement du programme, si l'utilisateur saisit le nombre 0, alors il se fermera instantanément.

Au lancement du programme, si l'utilisateur saisit le nombre 1, un étudiant tiré au hasard dans la promo sera affiché, puis un message demandant si l'utilisateur souhaite continuer sera affiché. Si il désire arrêter de tirer au sort des étudiants, il saisira le chiffre 0, qui l'emmènera donc dans le menu, avec la possibilité de re choisir son option en saisissant un chiffre (nous pouvons avec ce programme faire des allers-retours vers le menu et les options une infinité de fois).
Si il désire continuer de tirer au sort et d'afficher un étudiant, il devra donc saisir le chiffre 1, qui affichera à l'écran le prochain étudiant désigné.

A chaque fois qu'un étudiant est tiré au sort, le programme demandera à l'utilisateur si il souhaite continuer ou s'arrêter.

Une fois que tous les étudiants de la promo sont passés, le programme avertira l'utilisateur, en lui demandant si il souhaite continuer ou s'arrêter là.

L'utilisateur peut quitter à tout moment, même si tous les étudiants de la promo ne sont pas passés. De plus, le programme nous permet de tirer au sort un étudiant à l'infini, tant que l'utilisateur ne signale pas qu'il désire s'arrêter.

Démarche pour le tirage aléatoire d'un étudiant :

Pour effectuer ce tirage en respectant les différentes contraintes citées précédemment, nous avons crées différents sous programmes, appelés par la suite dans le ***main.***

Le premier sous programme nommé ***texteMenu*** permet d'afficher le menu avec ses différentes options.

Au lancement du programme, la nombre d'étudiants de la promo est affiché à l'aide du sous programme ***ListeEtudiants.nbEtudiant***, qui prend donc en paramètre le fichier .csv dans le dossier du programme, où se trouve un tableau recensant tout les étudiants de la promo.

Ensuite, le sous programme ***testMenu***, appelé également dans le ***main***, permet au lancement du programme de demander à l'utilisateur de choisir une option en saisissant un chiffre, à l'aide du sous programme ***saisieIntC***. Tant que l'utilisateur ne saisit pas le chiffre 0, 1 ou 2, un message d'erreur s'affichera, invitant l'utilisateur à re-saisir un chiffre.
Toujours dans ce sous programme se trouve un try – switch, qui permet de réaliser différentes actions en fonction du nombre saisi par l'utilisateur (choixUtilisateur).
Si l'utilisateur saisit le chiffre 2 (dans le menu), le sous programme ***testMenu*** va faire appel au sous programme ***afficherPromo***, qui permet donc d'afficher tous les étudiants de la promo avec leur numéro respectif.
Si l'utilisateur saisit le chiffre 0 (dans le menu), le programme va s'arrêter et afficher à l'utilisateur «Au revoir… ...».
Si l'utilisateur saisit le chiffre 0 autre part que dans le menu, il va être redirigé vers ce dernier.
Enfin, si l'utilisateur saisit le chiffre 1 (dans le menu), le sous programme ***testMenu*** va faire appel au sous programme ***traiterChoix1***, qui permet de tirer au sort et d'afficher un étudiant, en respectant évidemment les contraintes citées précédemment.
Tant que l'utilisateur saisit le chiffre 1 (hors du menu), il continuera d'afficher un étudiant tiré au sort.

Fonctionnement du sous programme traiterChoix1 :

Pour faire fonctionner ce sous programme, nous avons d'abord initialisé plusieurs variables et un tableau.

- la variable e, qui permet de stocker le numéro de l'étudiant qui vient de passer, qui change donc de valeur à chaque fois qu'un étudiant est tiré au sort.
- la variable nbPassage, qui compte le nombre d'étudiants passés (réinitialisée si tous les étudiants sont passés et que l'utilisateur souhaite continuer à tirer au sort des étudiants).
- la variable randomNumetu, qui permet de tirer un nombre aléatoire entre 0 et la taille de la promo (promo.length).
- le tableau d'entier etu[] de taille [promo.length], permettant de voir si un étudiant est passé ou non.
- le booléen choix, initialisé à true, qui permet de continuer l'exécution du code si il est vrai.

Au début du programme, l'ordinateur tire un nombre aléatoire entre 0 et la taille de la promo, et affiche l'étudiant associé a ce numéro. Les différentes variable et le tableau sont ensuite mis à jour (e, etu[], et nbPassage). La variable e contient désormais le chiffre associé a l'étudiant tiré précédemment, la variable nbPassage a été augmenté de 1, et le tableau etu[] a été modifié. En effet, ce tableau contient au début uniquement des cases vides, qui sont ensuite remplacées par des 1, au fur et à mesure du tirage d'étudiants.
Ensuite, le programme demande à l'utilisateur si il souhaite continuer ou non le tirage, en l'invitant à saisir le chiffre 1 ou 0. Tant que le chiffre saisi n'est pas un de ces deux, une boucle while va demander à l'utilisateur de re saisir un chiffre. Si l'utilisateur saisit 1, le booléen choix reste à true, et le programme peut donc se poursuivre. Dans le cas échéant, le booléen se met à false, et l'utilisateur est redirigé vers le menu. S'en suit une multitude de boucle for et while ainsi que des if et else, permettant de vérifier toutes les contraintes citées précédemment.

En effet, l'ordinateur va tirer un nombre aléatoire tant que :

- ce dernier est égal à e

ou

- que la case du tableau etu[] d'indice randomNumetu comporte un 1 (permet donc d'éviter qu'un étudiant soit tiré au sort alors que tous les étudiants de la promo ne sont pas encore tous passés).

Si nbPassage est égal à la taille de la promo (ce qui veut dire que tous les étudiants sont passés), le programme va en informer l'utilisateur, en lui demandant si il souhaite poursuivre un nouveau tour de promo (sans obligation d'être fini), ou si il souhaite s'arrêter. Si l'utilisateur souhaite s'arrêter là, le programme va le rediriger vers le menu.
Si il souhaite au contraire poursuivre, la variable nbPassage va être remise à 0, et une boucle for parcourant toutes les cases du tableau etu[] va les remettre à la valeur 0.

Quelques exemples d'exécutions avec 5 étudiants :

```
Menu.main({ });
nombre de lignes : 5
nombre de lignes : 5
Il y a : 5 etudiants

/*********************************************/
                        Veuillez choisir une option :
                1 - Tirer au sort un étudiant aléatoirement
                2 - Afficher la promo
                0 - Quitter
/*********************************************/


Choisir une option2
Etudiant n°1: Afritt Barack
Etudiant n°2: Auboisdormant Abel
Etudiant n°3: Balmaske Alonzo
Etudiant n°4: Bu Amede
Etudiant n°5: Cape Andy

/*********************************************/
                        Veuillez choisir une option :
                1 - Tirer au sort un étudiant aléatoirement
                2 - Afficher la promo
                0 - Quitter
/*********************************************/


Choisir une option
```

Test de la fonction afficherPromo.

```
/*********************************************/
                        Veuillez choisir une option :
                1 - Tirer au sort un étudiant aléatoirement
                2 - Afficher la promo
                0 - Quitter
/*********************************************/


Choisir une option1
~ Cape Andy ~

Voulez-vous continuer ? (1 pour oui et 0 pour quitter)
```

Message demandant à l'utilisateur si il souhaite poursuivre ou non.

```
Choisir une option1
~ Cape Andy ~

Voulez-vous continuer ? (1 pour oui et 0 pour quitter)

1
~ Bu Amede ~

Voulez-vous continuer ? (1 pour oui et 0 pour quitter)

1
~ Balmaske Alonzo ~

Voulez-vous continuer ? (1 pour oui et 0 pour quitter)

1
~ Auboisdormant Abel ~

Voulez-vous continuer ? (1 pour oui et 0 pour quitter)

1
~ Afritt Barack ~

Voulez-vous continuer ? (1 pour oui et 0 pour quitter)

1
Tous les étudiants sont passés, voulez-vous continuer ? (1 pour oui et 0 pour quitter)
```

Les contraintes sont bien respectées, aucun étudiant n'est tiré au sort deux fois de suite et ils ont tous été tiré au sort une fois. Le programme informe l'utilisateur que tous les étudiants sont passés, et lui demande si il souhaite poursuivre sur un autre tour de promo ou bien s'arrêter là.

```
Erreur ! Veuillez saisir 1 ou 0.6
Erreur ! Veuillez saisir 1 ou 0.4
Erreur ! Veuillez saisir 1 ou 0.8
Erreur ! Veuillez saisir 1 ou 0.9
Erreur ! Veuillez saisir 1 ou 0.
```

Message d'erreur invitant l'utilisateur à saisir un chiffre tant qu'il n'est pas égal à 0 ou 1.

```
Choisir une option
Erreur ! Donnez une valeur comprise 0 et 2.6
Choisir une option
Erreur ! Donnez une valeur comprise 0 et 2.4
Choisir une option
Erreur ! Donnez une valeur comprise 0 et 2.8
Choisir une option
Erreur ! Donnez une valeur comprise 0 et 2.9
Choisir une option
Erreur ! Donnez une valeur comprise 0 et 2.7
Choisir une option
Erreur ! Donnez une valeur comprise 0 et 2.
```

Message d'erreur dans le menu invitant l'utilisateur à saisir un chiffre entier tant qu'il n'est pas compris entre 0 et 2 inclus.

```
Menu.main({ });
nombre de lignes : 5
nombre de lignes : 5
Il y a : 5 etudiants

/*********************************************/
                        Veuillez choisir une option :
                1 - Tirer au sort un étudiant aléatoirement
                2 - Afficher la promo
                0 - Quitter
/*********************************************/


Choisir une option0
AU REVOIR ...   ...
```

Option quitter dans le menu.