Examen pratique d'août 2023

CAMPAGNE DE FINANCEMENT PARTICIPATIF

Le **financement participatif**, ou *crowdfunding*, est une méthode de collecte de fonds s'effectuant via une plateforme en ligne et permettant à toute personne de contribuer au financement d'un projet, imaginé par une personne ou une entreprise, afin de permettre sa réalisation.

Par exemple, supposons que vous ayez l'idée d'un concept original : rendre un avion en papier téléguidable grâce à un petit module amovible et une application pour smartphone (voir les figures 1 et 2).

Fig. 1 *– Un avion en papier téléguidé par smartphone.*

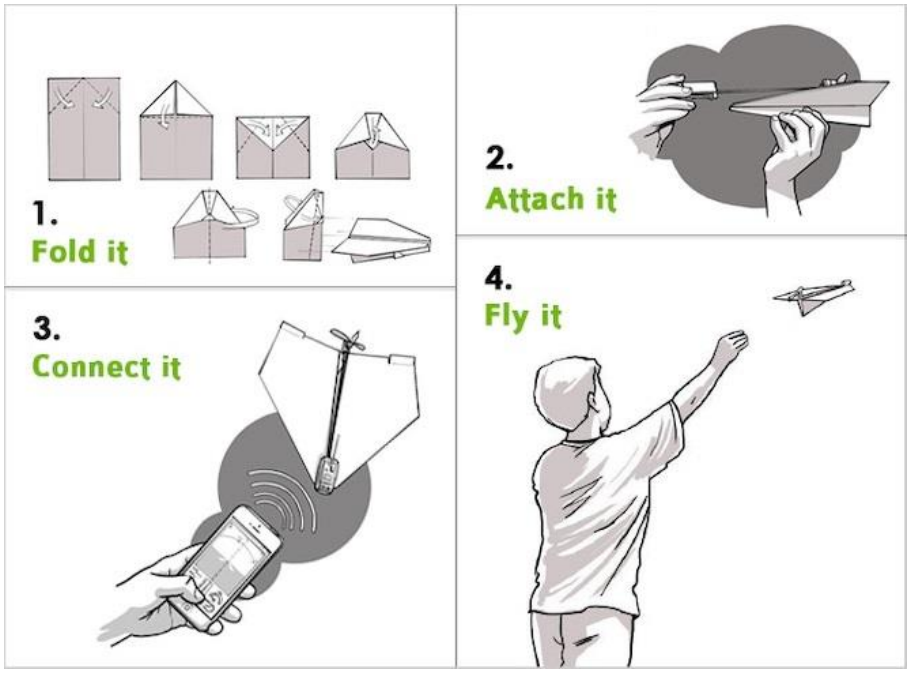


Fig. 2 *– Mode d'emploi.*

Vous estimez que vous avez besoin d'un budget d'au moins 50.000€ pour débuter la production de ce produit afin qu'il soit rentable. Pour éviter de vous endetter, vous décidez de lancer une campagne de financement participatif sur la plateforme *Kickstarter*.

Une fois la campagne lancée, vous souhaitez disposer d'un maximum de données sur le déroulement de la campagne afin de déterminer suffisamment tôt la viabilité de cette dernière.

DESCRIPTION DE L'APPLICATION

Au lancement de l'application, l'utilisateur doit spécifier deux données :

- la **durée en jours** de la campagne de financement (sur *Kickstarter*, la durée d'une campagne ne peut pas excéder 90 jours) ;
- le **seuil de financement** à atteindre pour que le projet soit réalisable.

La plateforme Kickstarter n'accepte pas les montants monétaires avec décimales. Les montants sont donc tous exprimés sous la forme de nombres entiers.

L'utilisateur doit ensuite pouvoir encoder les montants des contributions chronologiquement, jour par jour.

À tout moment, il doit pouvoir visualiser les informations suivantes : le montant total des contributions ; le pourcentage de complétion du financement ; le nombre total de contributions ; les données quotidiennes.

Lorsque le seuil de financement est atteint, un message doit être afficher pour en informer l'utilisateur.

Référez-vous aux exemples d'exécutions donnés en fin d'énoncé afin de mieux comprendre le résultat attendu.

AVANT-PROPOS

À deux exceptions près, les types **String** et **PrintStream**, vous ne pouvez pas utiliser le paradigme orienté objet pour coder cette application. N'oubliez pas de qualifier toutes vos fonctions de `static` !

PRÉPARATION

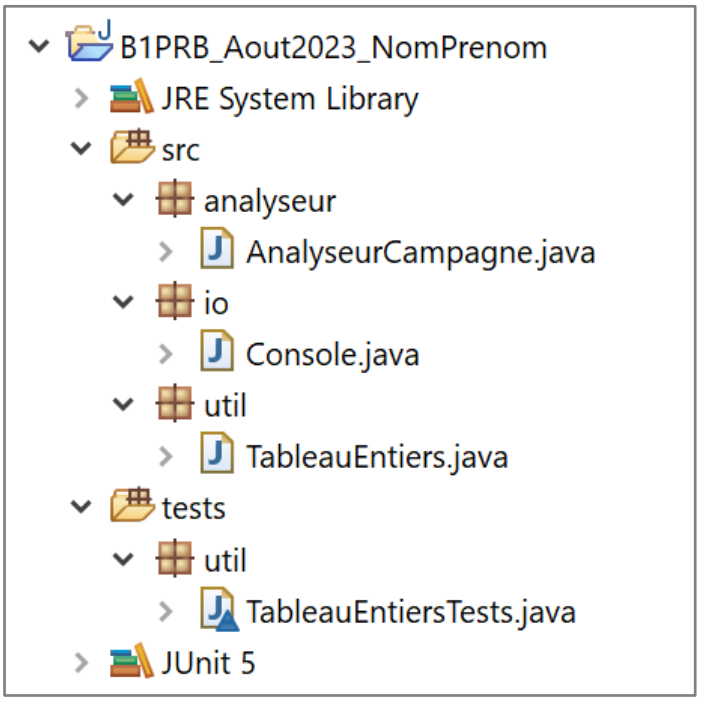
1. Dans Eclipse, créez un projet Java nommé **B1PRB_Aout2023_NomPrenom**, où **NomPrenom** doit être remplacé par vos nom et prénom.

   N'oubliez pas de vérifier que l'encodage utilisé pour les fichiers est l'**UTF-8**. Pour ce faire, il faut suivre le chemin : *Window* > *Preferences* > *General* > *Workspace* > *Text file encoding*.

   Pour résoudre le **bug d'affichage en console** des caractères accentués, suivez la procédure suivante :

   - se rendre dans le panneau de configuration du JRE utilisé par Eclipse (*Window* > *Preferences* > *Java* > *Installed JREs*) ;
   - sélectionner le JRE, puis cliquer sur le bouton ***Edit*** ;
   - saisir « -Dsun.stdout.encoding=UTF-8 » dans le champ ***Default VM arguments***.

2. Reproduisez exactement l'organisation suivante :



   La classe **Console** est à télécharger au début de la page HELMo Learn du cours. Les autres classes doivent être créées par vos soins. La fonction *main* doit être déclarée dans la classe **AnalyseurCampagne**.

   [IMPORTANT] Le projet déposé sur l'espace de cours doit être complet et exécutable !

FONCTIONNALITÉS INDISPENSABLES

Les fonctionnalités suivantes doivent être opérationnelles, tout en veillant à respecter les consignes données dans leur description :

- Les trois fonctions *ajouterElement*, *sommer(int[ ] t)* et *sommer(int[ ][ ] t)* de la classe **TableauEntiers** doivent être correctement réalisées, testées et utilisées.
- Dans la fonction *main* de la classe **AnalyseurCampagne**, un tableau d'entiers à deux dimensions doit être correctement créé et initialisé afin de répertorier les **contributions quotidiennes** (voir la section *« Les structures de données »*).
- La fonction *main* doit effectuer l'acquisition de la durée de la campagne et du seuil de financement du projet. De plus, l'utilisateur doit pouvoir encoder autant de contributions que souhaité pour la première journée de la campagne (voir l'exemple d'exécution *« MINIMUM REQUIS »* en fin d'énoncé). Les montants de ces contributions doivent être effectivement ajoutées à la première ligne du tableau à deux dimensions (cité au point précédent) à l'aide de la fonction **TableauEntiers**.*ajouterElement*.

  [REMARQUE] Dans cette version minimale, il n'est pas nécessaire de vérifier la validité des données saisies par l'utilisateur.

[IMPORTANT] Si l'une de ces fonctionnalités n'est pas observée, cela entraîne automatiquement l'échec pour l'examen. Attention, la réussite de ces fonctionnalités seules ne garantit pas l'obtention d'une note supérieure ou égale à 10 / 20 ! Ne vous contentez pas de ces fonctionnalités et veillez à soigner les tests unitaires et la javadoc des fonctions.

APPROCHE RECOMMANDÉE

Dans un premier temps, **ne réalisez que les fonctions indispensables**. Ces dernières sont indiquées dans la section *« Fonctionnalités indispensables »* et sont marquées d'un point d'exclamation ⓘ au niveau de leurs descriptions.

Réalisez ensuite la fonction *main* sur base des directives données dans la section *« Fonctionnalités indispensables »* et de l'exemple d'exécution MINIMUM REQUIS.

Il est recommandé de ne réaliser la suite du programme que lorsque ces deux premières étapes sont terminées.

LES STRUCTURES DE DONNÉES

Les montants des contributions quotidiennes

Utilisez un **tableau d'entiers à deux dimensions** pour enregistrer les montants des contributions par jour de campagne. Chaque ligne du tableau répertorie les montants obtenus lors d'une même journée.

Par exemple, dans la figure 3 ci-dessous, la 1ère ligne correspond aux contributions de la 1ère journée, la 2e aux contributions de la 2e journée et la 3e aux contributions de la 3e journée.

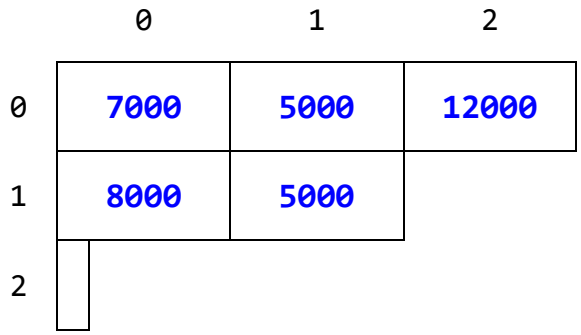


Fig. 3 – Un tableau répertoriant les contributions quotidiennes pour une campagne de 3 jours.

La 1ère ligne est complète et comporte 3 contributions d'un total de 24.000€.

La 2e ligne comporte 2 contributions d'un total de 13.000€. Sur base du tableau seul, il n'est pas possible de déterminer si la journée correspondante est terminée. A cette fin, une autre variable est nécessaire.

La 3e ligne du tableau est vide car aucune contribution n'a encore été encodée pour la journée correspondante.

Au total, il y a actuellement 5 contributions pour un montant total de 37.000€.

Lorsque l'utilisateur a spécifié la durée de la campagne, créez ce tableau avec le nombre de lignes correspondant. Chaque ligne doit être un tableau vide (voir la figure 4).

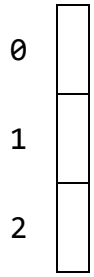


Fig. 4 – Le tableau au lancement d'une campagne de 3 jours (aucune contribution encodée).

LES CLASSES TABLEAUENTIERS ET TABLEAUENTIERSTESTS

Toutes les fonctions de la classe **TableauEntiers** doivent être documentées avec des commentaires javadoc et testées à l'aide de JUnit 5.

Dans la classe **TableauEntiers**, déclarez :

→ Une fonction nommée *ajouterElement* qui retourne une copie des éléments d'un tableau d'entiers à une dimension augmentée d'un nouvel élément :

```
public static int[] ajouterElement(int[] t, int element)
```

Le paramètre `t` est le tableau dont les éléments doivent être copiés.

Le paramètre `element` est l'élément à ajouter aux éléments du tableau `t`.

Exemple : si `t` est le tableau `[4, 5, 6]` et `element` l'entier 7, alors le tableau retourné est `[4, 5, 6, 7]`.

[SUGGESTION] Aidez-vous de la fonction *copyOf* de la classe **Arrays**.

→ Une fonction nommée *sommer* qui retourne la somme de toutes les valeurs d'un tableau d'entiers à une dimension :

```
public static int sommer(int[] t)
```

Exemple : si `t` est le tableau `[2, 8, 1]`, alors la fonction retourne l'entier 11.

[CAS PARTICULIER] Si `t` est un tableau vide, alors la fonction retourne 0.

[PRÉCONDITION] La référence réceptionnée par le paramètre `t` est supposée valide (ne vaut pas `null`).

→ Une fonction nommée *sommer* qui retourne la somme de toutes les valeurs d'un tableau d'entiers à deux dimensions :

```
public static int sommer(int[][] t)
```

Exemple : si `t` est le tableau `[[2, 8, 1], [], [4, -2]]`, alors la fonction retourne l'entier 13.

[CONTRAINTE] Pour effectuer son traitement, cette fonction doit utiliser la fonction *sommer(int[] t)* précédemment déclarée.

[CAS PARTICULIER] Si `t` est un tableau vide ou un tableau ne comportant que des lignes vides, alors la fonction retourne 0.

[PRÉCONDITIONS] Le tableau réceptionné par le paramètre `t` est supposé respecter les propriétés suivantes : sa référence est valide (ne vaut pas `null`) ; chaque référence de ligne est valide (ne vaut pas `null`).

→ Une fonction nommée *compterElements* qui détermine le nombre d'éléments contenus dans un tableau d'entiers à deux dimensions :

```
public static int compterElements(int[][] t)
```

Exemple : si `t` est le tableau `[[2, 8, 1], [], [4, -2]]`, alors la fonction retourne l'entier 5.

[CAS PARTICULIER] Si `t` est un tableau vide ou un tableau ne comportant que des lignes vides, alors la fonction retourne 0.

[PRÉCONDITIONS] Le tableau réceptionné par le paramètre `t` est supposé respecter les propriétés suivantes : sa référence est valide (ne vaut pas `null`) ; chaque référence de ligne est valide (ne vaut pas `null`).

LES CLASSES ANALYSEURCAMPAGNE ET ANALYSEURCAMPAGNETESTS

À l'exception de la fonction *main*, toutes les fonctions de la classe **AnalyseurCampagne** doivent être documentées avec des commentaires javadoc. Cependant, aucune des fonctions de cette classe ne doit être testée à l'aide de JUnit 5.

Dans la classe **AnalyseurCampagne**, déclarez :

→ Une fonction nommée *lireEntier* qui effectue l'acquisition sécurisée d'un nombre entier :

```
public static int lireEntier(String question)
```

Le paramètre `question` est la chaîne de caractères à afficher en guise de question pour spécifier la valeur que doit saisir l'utilisateur.

L'acquisition **doit être répétée** tant que la saisie ne correspond pas à un nombre entier.

Exemple : si `question` est la chaîne de caractères `"Votre âge ? "`, alors l'exécution de la fonction pourrait produire l'affichage suivant en console :

```
Votre âge ? dix-neuf
Votre âge ? 19
```

L'acquisition est ici répétée une fois car la première saisie ne correspond pas à une représentation valide d'un nombre entier.

[PRÉCONDITION] La chaîne `question` est supposée valide.

[CONTRAINTE] Pour déterminer la validité de la chaîne de caractères saisie, utiliser une expression régulière :

- L'entier saisi doit comporter au moins un chiffre et doit être compris dans l'intervalle]-2.000.000.000 ; 2.000.0000.000[.

- Il peut être préfixé d'un signe - ou +.
- Si des zéros non significatifs sont présents, ces derniers sont ignorés (par exemple, la saisie `"0000000050000"` est considérée comme valide).

→ Une fonction nommée *lireEntier* qui effectue l'acquisition sécurisée d'un nombre entier compris dans un intervalle donné :

```
public static int lireEntier(String question, int min, int max)
```

Le paramètre `question` est la chaîne de caractères à afficher en guise de question pour spécifier la valeur que doit saisir l'utilisateur.

Les paramètres `min` et `max` sont les bornes inclusives de l'intervalle.

L'acquisition **doit être répétée** tant que la saisie ne correspond pas à un nombre entier compris dans l'intervalle spécifié.

Exemple : si `question` est la chaîne caractères `"Votre âge ? "`, `min` l'entier 0 et `max` l'entier 120, alors l'exécution de la fonction pourrait produire l'affichage suivant en console :

```
Votre âge ? dix-neuf
Votre âge ? 190
Votre âge ? 19
```

L'acquisition est ici répétée à deux reprises car la première saisie ne correspond pas à une représentation valide d'un nombre entier et la seconde ne correspond pas à un entier compris entre 0 et 120.

[PRÉCONDITIONS] Les 3 paramètres sont supposés valides.

[CONTRAINTE] Pour effectuer son traitement, cette fonction doit utiliser la fonction *lireEntier(String question)* précédemment déclarée.

LES EXEMPLES D'EXÉCUTION

MINIMUM REQUIS

```
Durée de la campagne de financement (en jours) ? 3
Seuil de financement à atteindre ? 50000

Montant de la contribution ? 7000
(C)ontinuer, (q)uitter ? c

Montant de la contribution ? 5000
(C)ontinuer, (q)uitter ? c

Montant de la contribution ? 12000
(C)ontinuer, (q)uitter ? q
```

VERSION IDÉALE

Lorsqu'une contribution est encodée, pensez à vérifier si le seuil de financement a été atteint.

Les données doivent être obtenue à l'aide des fonctions sommer(int[ ] t), sommer(int[ ][ ] t) et compterElements(int[ ][ ] t) au lieu d'utiliser des variables supplémentaires dans votre fonction main.

```
Durée de la campagne de financement (en jours) ? 3
Seuil de financement à atteindre ? 50000

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 3

Montant récolté = 0 sur 50000 EUR (0%)
0 contribution(s)

Jour #1 : montant des contributions = 0 EUR ; 0 contribution(s)

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 1

Montant de la contribution ? 7000

MENU
```

```
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 1

Montant de la contribution ? 5000

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 1

Montant de la contribution ? 12000

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 3

Montant récolté = 24000 sur 50000 EUR (48%)
3 contribution(s)

Jour #1 : montant des contributions = 24000 EUR ; 3 contribution(s)

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 2

Jour de la campagne #2.

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 1

Montant de la contribution ? 8000

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
```

```
4. Quitter
Choix ? 1

Montant de la contribution ? 5000

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 3

Montant récolté = 37000 sur 50000 EUR (74%)
5 contribution(s)

Jour #1 : montant des contributions = 24000 EUR ; 3 contribution(s)
Jour #2 : montant des contributions = 13000 EUR ; 2 contribution(s)

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 2

Jour de la campagne #3.

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 1

Montant de la contribution ? 9000

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 1

Montant de la contribution ? 6000
Bravo, le projet est financé !

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
```

```
Choix ? 1

Montant de la contribution ? 7000

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 3

Montant récolté = 59000 sur 50000 EUR (118%)
8 contribution(s)

Jour #1 : montant des contributions = 24000 EUR ; 3 contribution(s)
Jour #2 : montant des contributions = 13000 EUR ; 2 contribution(s)
Jour #3 : montant des contributions = 22000 EUR ; 3 contribution(s)

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 2

Campagne terminée.

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 1

Le financement du projet est terminé.

MENU
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 3

Montant récolté = 59000 sur 50000 EUR (118%)
8 contribution(s)

Jour #1 : montant des contributions = 24000 EUR ; 3 contribution(s)
Jour #2 : montant des contributions = 13000 EUR ; 2 contribution(s)
Jour #3 : montant des contributions = 22000 EUR ; 3 contribution(s)

MENU
```

```
1. Encoder une contribution
2. Passer à la journée suivante
3. Consulter les données
4. Quitter
Choix ? 4

Fin du programme.
```