R1.01 – TP2(A)
Booléens et Structures de contrôle

L'objectif de ce TP est d'étudier :
- *les expressions booléennes*
- *les structures de contrôle (if et loop)*

1. Avant de commencer...

✓ Lisez entièrement ce sujet

✓ Ouvrez un terminal et placez-vous dans votre répertoire `R1.01` et lancez `IJ` en tapant la commande `idea`

✓ Sous `IJ` : créez un nouveau projet TP2A dans le répertoire `R1.01`

Rappel de la procédure à suivre dans le fichier UTILISATION_IJ accessible à partir de ce parcours

NOTE : les parties sont indépendantes les unes des autres, commencez par celle qui vous inspire le plus…

2. Conditions météorologiques : classe Meteo

Dans cette partie, vous jonglerez entre booléens et instructions conditionnelles pour déterminer quel est l'équipement optimal selon les conditions météorologiques du jour…

RÈGLE « ÉQUIPEMENT OPTIMAL » :
- Nous considérons qu'on est toujours bien équipé s'il ne pleut pas
- S'il pleut, pour être bien équipé il faut :
  - *Avoir un parapluie ou un chapeau*
  - *Ne pas être chaussé de tongs*

2.1. Créez une nouvelle classe java `meteo` (commande New Java Class du menu File)

2.2. Dans la classe `meteo` :

- Créez une procédure `main`

cf. raccourci rappelé dans le fichier UTILISATION_IJ accessible à partir de ce parcours

- Dans l'enveloppe de la classe `main` :
  - ✓ Déclarez 5 variables se type `boolean` : `pluie`, `parapluie`, `tongs`, `chapeau`, `bonEquipement`
  - ✓ Déclarez une variable `lecteur` de type `Scanner` et importez la classe `java.util.Scanner`

cf. le raccourci rappelé dans le fichier UTILISATION_IJ accessible à partir de ce parcours

  - ✓ Écrivez les instructions permettant :
    - l'initialisation de `pluie`, `parapluie`, `tongs` et `chapeau` par saisie :
      l'utilisateur doit être invité à répondre `true` ou `false` aux questions relatives à la météo ou à son équipement
    - l'initialisation de la variable `bonEquipement` à partir des valeurs des booléens `pluie`, `parapluie`, `tongs` et `chapeau` en respect de la règle « ÉQUIPEMENT OPTIMAL »
- Terminez le programme par un message indiquant à l'utilisateur s'il est bien équipé
- Compilez et exécutez le programme plusieurs fois pour le tester

3. Conditions calendaires : classe Calendrier

*Dans cette partie, vous utiliserez des **structures conditionnelles** pour afficher le nombre de jours que comporte une année saisie par l'utilisateur.*

3.1. Créez une nouvelle classe `Calendrier`

3.2. Dans la classe `Calendrier` :

- Créez une procédure `main` où vous déclarerez :
  - ✓ une variable uneAnnee et une variable `nbJours` de type entier
  - ✓ ainsi qu'une variable `lecteur` de type `Scanner` et importez la classe `java.util.Scanner`

3.3. Années qui terminent un siècle

INDICATIONS :
Une année multiple de 100 termine un siècle (ex : l'an 200 a terminé le 2ème siècle, l'an 1500 a terminé le 15ème siècle, l'an 2000 a terminé le 20ème siècle, etc.)

- Dans la procédure `main` :
  - ✓ Écrivez les instructions d'initialisation par saisie de la variable uneAnnee
  - ✓ Écrivez les instructions permettant à l'utilisateur de savoir si uneAnnee termine ou non un siècle
- Compilez puis exécutez plusieurs fois ce programme en testant au fil des exécutions les valeurs : 100, 1810, 2000, 2013 (pour la variable uneAnnee)

3.4. Nombre de jours d'une année donnée

INDICATIONS :
- ✓ Une année bissextile comporte 366 jours (29 en février), une année "normale" comporte 365 jours
- ✓ Pour être bissextile, une année est soit :
  - * une année qui ne termine pas un siècle mais qui est multiple de 4
  - * une année qui termine un siècle et dans ce cas, son quotient dans la division par 100 doit être multiple de 4

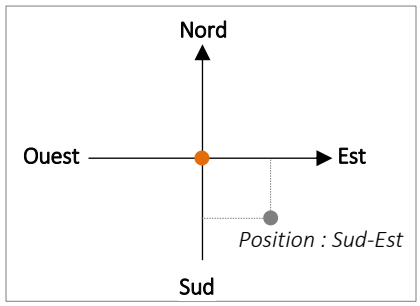
Exemples :
- *2012 était bissextile (ne termine pas un siècle et est multiple de 4)*
- *1900 n'était pas bissextile (termine un siècle et son quotient dans la division par 100 n'est pas multiple de 4)*
- *2000 était bissextile (termine un siècle et son quotient dans la division par 100 est multiple de 4)*
- *2021 ne termine pas un siècle (pas multiple de 100) et n'est pas bissextile (un nombre impair n'est pas multiple de 4)*

- Complétez la procédure `main` avec les instructions permettant à l'utilisateur de :
  - ✓ savoir si uneAnnee est une année bissextile ou non
  - ✓ connaître le nombre de jours de l'année uneAnnee
- Compilez, puis exécutez plusieurs fois la classe Calendrier :
  Au fil des exécutions vous donnerez à la variable uneAnnee, *les valeurs* 1900, 1915, 1916, 2000, 2014, 2020

	1900	1915	1916	2000	2014	2020
termine un siècle	oui	non	non	oui	non	non
bissextile	non	non	oui	oui	non	oui
nombre de jours	365	365	366	366	365	366

4. Conditions géographiques : classe Localisation

Dans cette partie vous utiliserez à nouveau les structures conditionnelles pour déterminer la position géographique d'un utilisateur relativement à un point de référence de coordonnées (0,0).



4.1. Créez une nouvelle classe `Localisation`

4.2. Dans la classe `Localisation`, créez une procédure `main`
où vous déclarerez :

- ✓ deux entiers `x` et `y`
- ✓ une variable `lecteur` de type `Scanner` *(pensez à importer la classe `java.util.Scanner`)*

4.3. **Version sans boucle**

- Dans la procédure `main` ajoutez les instructions permettant :
  - ✓ la saisie par l'utilisateur de ses coordonnées (`x` et `y`)
  - ✓ l'affichage de sa position (centre / nord / sud / nord-est / sud-est / nord-ouest / sud-ouest)

  L'algorithme utilisé devra optimiser les tests effectués…
- Compilez et exécutez ce programme

4.4. **EXTENSION - Version avec boucle** :

Complétez le programme pour que l'utilisateur puisse "jouer " au moins une fois et « rejouer » tant qu'il le désire.

Pour cela :

- Déclarez un caractère `jeu` initialisé à **'o'**
- Reportez les instructions précédentes dans une boucle `do {...} while (...);` contrôlée par la valeur de la variable `jeu` dont la saisie sera demandée à l'utilisateur en fin du bloc d'instructions de la boucle
  Exemple de message de saisie : *"Tapez o pour rejouer, n pour arrêter"*

5. Compétition de parachutisme : classe Competition

*Dans cette partie, vous serez amenés à utiliser différentes **structures itératives.***

THÈME : COMPÉTITION DE PARACHUTISME

✓ Dans cette compétition, les concurrents doivent effectuer un enchaînement de figures en chute libre. Cet enchaînement est chronométré en secondes et centièmes de seconde, par un jury au sol.
À titre indicatif, un temps habituel se situe entre 6 et 16 secondes.

✓ Quand un concurrent arrive au sol :

* chaque membre du jury (juge) donne son estimation du temps qu'il a mis entre le début de sa première figure et la fin de sa dernière figure.

* la moyenne des temps estimés par les juges contribue à l'évaluation de sa performance

5.1. Créez une nouvelle classe `Competition` dans laquelle vous créerez une procédure `main`

5.2. **Saisie et affichage de l'estimation du temps d'un concurrent par chaque membre du jury**

- Déclarez une variable `temps` de type `float`, un entier `nbJuges` et une variable `lecteur` de type `Scanner`
- Écrivez les instructions permettant :
  - ✓ la saisie du nombre de juges (membres du jury)
  - ✓ dans une boucle contrôlée par le nombre de juges : la saisie du temps que chaque juge a estimé

  NOTES
  (1) réfléchissez au type de boucle le plus adapté
  (2) la saisie d'un temps doit être précédée du message "Temps estimé par le juge N°X ? "
  (où X est le n° d'un juge, ce numéro étant obtenu dynamiquement)
  (3) aucun contrôle de saisie n'est requis (vous ferez attention lord de l'exécution de saisir des valeurs strictement positives et compatibles avec le type `float`
- Compilez et exécutez

5.3. Saisie contrôlée et affichage de l'estimation du temps d'un concurrent par chaque membre du jury

- Déclarez une constante de type `float`, nommée `tempsMAX` et initialisée avec la valeur 20.0f
- Enrichissez la boucle de saisie des temps estimés par les juges, par une boucle de contrôle de saisie du temps : l'utilisateur doit y être invité à répéter la saisie jusqu'à ce qu'il ait entré une valeur comprise entre 0 et 20

  NOTE Réfléchissez au type de boucle le plus adapté et déclarez si besoin de nouvelles variables pour la gérer
- Compilez et exécutez

5.4. Calcul et affichage du temps minimum, du temps maximum et de la moyenne des temps estimés par les juges

- Déclarez des variables `min`, `max` et `moy` de type `float`
- Complétez la classe `Competition` avec les instructions permettant :
  - ✓ le calcul du *temps minimum* et du *temps maximum* parmi les temps estimés par les juges
  - ✓ le calcul de la *moyenne des temps estimés* par les juges
  - ✓ l'affichage du temps minimum, du temps maximum et de la moyenne des temps estimés (vous soignerez l'affichage de ces valeurs)
- Compilez et exécutez