Yann.Secq@univ-lille.fr
Fabien.Delecroix@univ-lille.fr

Préparation au TD n°2

Objectifs : Consolider les notions de **types**, **conversions**, **opérations** et **appel de fonction**. Initier aux **conditions** et **alternatives simples**.

1 Retour sur les appels de fonction

Exercice 10 du TD n°1.

Exercice 1 : Appels de fonction

En TP, vous avez utilisé des fonctions fournies dans `program.jar` qui permettent de réaliser diverses choses dans un programme en `iJava`. Ici, pour chaque opération souhaitée, indiquez la fonction pour la réaliser et le code correspondant.

Rappel : Les fonctions vues jusqu'ici se nomment : `charAt`, `length`, `print`, `println`, `random`, `readChar`, `readDouble`, `readInt`, `substring`.

	Ce qu'on souhaite faire	**Nom de la fonction utile**	**Code correspondant**
a	afficher l'entier 42		
b	déclarer une chaîne de caractères `s` et l'initialiser par la sous-chaine des 5 premiers caractères de `"abcdefgh"`		
c	récupérer la taille de la chaîne de caractères `s` et la stocker dans une variable `taille` à déclarer		
d	Déclarer d'abord une variable `n` et stockez-y un entier saisi par l'utilisateur. Puis triplez la valeur contenue par `n`		
e	Déclarer d'abord une variable `d` et stockez-y un double saisi par l'utilisateur. Puis divisez par quatre la valeur contenue par `d`		

2 Conditions et booléens

2.1 Alg(Un)o

Alg(Un)o est un jeu de cartes dans lequel les participant·e·s cherchent à se débarrasser de toutes leurs cartes afin de gagner la partie. Pour cela chaque participant·e dispose de cartes d'un jeu classique et de cartes *conditions* (petites cartes vertes) et *alternative* (grandes cartes oranges) leur permettant d'exprimer des situations sur un ensemble de cartes et des actions liées (piocher ou se débarrasser d'une carte).

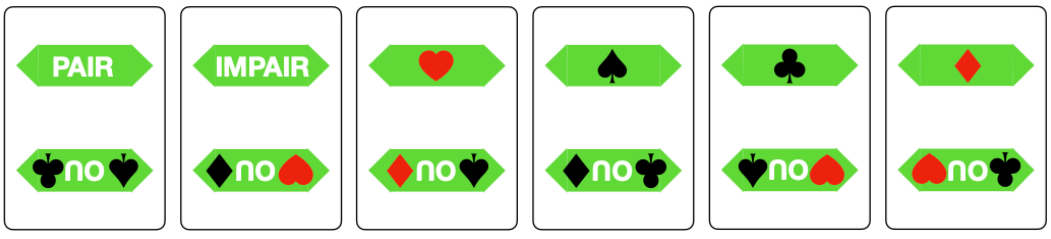

FIGURE 1 – Cartes conditions du jeu Alg(un)o

En jouant, les participant·e·s évaluent des conditions par rapport à des situations et déduisent les actions à réaliser en fonction de la valeur de la condition. Bien que ludique cette activité entraîne des mécanismes de raisonnements importants en algorithmique et programmation. Après présentation et distribution du jeu par votre enseignant·e, débutez une partie et jouez-y en groupe de 4 ou 5 pour quelques tours de jeu.

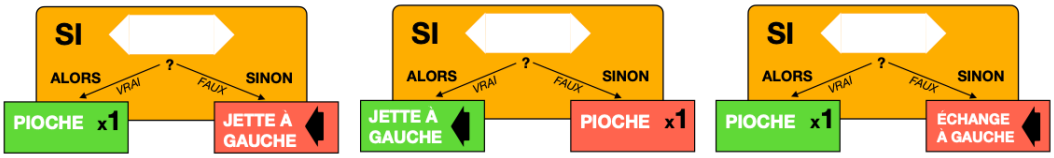

FIGURE 2 – Cartes alternatives du jeu Alg(un)o

Les règles sont simples, chacun·e dispose de 5 cartes normales qu'il ou elle dispose faces visibles sur la table, puis chacun à son tour, on tire une carte *alternative* et deux cartes *condition*. Le but est de choisir la condition qui est la plus favorable pour soi même et la plus défavorable pour les autres participant·e·s, sachant que l'algorithme défini s'applique d'abord au joueur courant puis successivement aux autres joueuses en tournant dans le sens horaire. A noter : lorsque vous avez une action pour vous débarasser d'une ou plusieurs cartes, vous pouvez les choisir librement, ce n'est pas du tout lié à la condition.

Exercice 2 : La bonne condition

Dans cet exercice, on se trouve dans une phase de jeu où trois joueuses et joueurs disposent des mains illustrées ci-dessous. Par exemple, la joueuse 1 a 3 cartes : le 4 de trèfle, le 2 de trèfle et le 3 de trèfle.

Main J1	Main J2	Main J3
$4\clubsuit, 2\clubsuit, 3\clubsuit$	$5\clubsuit, V\heartsuit, A\clubsuit$	$10\diamondsuit, 9\diamondsuit, R\diamondsuit$

Étant donnée la configuration ci-dessus de jeu à Alg(Un)o, complétez le tableau qui suit :
— pour les 3 premières lignes, en indiquant pour chaque joueur s'il remplit ou non la condition évaluée.
— pour les 2 dernières lignes, en indiquant une condition qui soit remplie par les joueurs et joueuses dont la condition est à `true`, mais pas ceux à `false`.

Condition évaluée	J1	J2	J3
♣ ET ♡			
♣ OU ♢			
♠ OU ♣			
	`true`	`true`	`false`
	`false`	`true`	`false`

Exercice 3 : La bonne alternative

On s'intéresse dans cet exercice à une alternative complète, c'est-à-dire constitutée de sa condition et des deux actions à déclencher en fonction de la valeur de la condition.

Main J1	Main J2	Main J3	Pioche
$4\clubsuit, 2\clubsuit, 3\clubsuit$	$5\clubsuit, V\heartsuit, A\clubsuit$	$10\diamondsuit, 9\diamondsuit, R\diamondsuit$	$A\diamondsuit, D\heartsuit$

Étant donnée la configuration ci-dessus de jeu à Alg(Un)o, que va-t-il se passer sachant que l'alternative est la suivante : **`SI (♡ OU ♢) ALORS PASSE À GAUCHE SINON PIOCHEx1`**

On supposera ici que lorsqu'un joueur doit passer une carte, il choisit celle située le plus à droite de sa main. Remplissez le tableau suivant en étant attentif à ce que la dernière ligne corresponde à l'état des différentes mains une fois que tous les joueurs ont joué, en appliquant le coup du premier joueur au dernier.

	J1	J2	J3
Condition			
Action			
Main			

On suppose maintenant que l'on dispose des actions `PASSE À GAUCHE`, `PASSE À DROITE`, `PIOCHEx1` et que l'on peut construire les conditions que l'on souhaite. Étant donné la configuration des mains indiquée ci-dessous :

Main J1	Main J2	Main J3	Pioche
$2\clubsuit, 6\heartsuit, 3\clubsuit$	$V\spadesuit, D\heartsuit, A\clubsuit$	$7\spadesuit, 9\diamondsuit, R\diamondsuit$	$D\heartsuit, A\diamondsuit$

Quelle est la meilleure alternative que peut construire les joueurs et joueuses J1 ? le J2 ? lJ3 ?

Exercice 4 : Construire une alternative entièrement

Nous allons maintenant utiliser une version légèrement modifiée d'Alg(Un)o dans laquelle l'alternative est suivie d'une instruction. Cette dernière est exécutée systématiquement : que cela soit la branche de droite ou de gauche qui est sélectionnée (en fonction de la valeur de la condition), une fois l'alternative terminée, l'exécution séquentielle se poursuit avec les instructions se trouvant à la suite de l'alternative. Le schéma ci-dessous illustre cela avec les flèches situées après les actions dans les branches qui aboutissent toutes deux sur l'instruction suivant l'alternative.

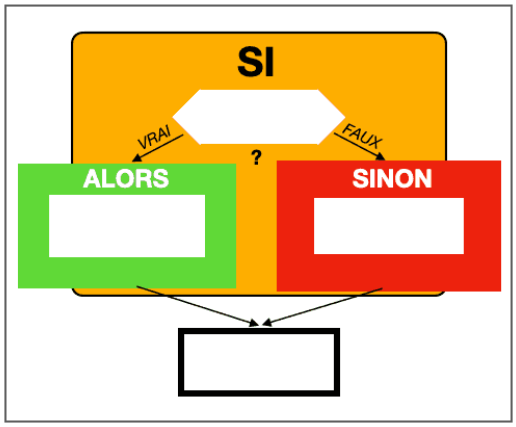

FIGURE 3 – Alternative suivie d'une instruction (exécutée quoiqu'il arrive)

Nous allons dans cette dernière partie construire l'ensemble d'une alternative en sélectionnant judicieusement à la fois la condition et les actions des deux branches, ainsi que l'instruction suivant l'alternative, qui sera donc systématiquement exécutée.

Voici les mains des différents participants :

Main J1	Main J2	Main J3	Pioche
$9\diamondsuit, 4\clubsuit, 3\heartsuit$	$5\clubsuit, V\heartsuit, A\clubsuit$	$10\diamondsuit, R\diamondsuit, 2\clubsuit,$	$A\diamondsuit, D\heartsuit$

Les actions disponibles sont : `PASSE À GAUCHE`, `PASSE À DROITE`, `PIOCHEx1` et `DÉFAUSSEx1`.

Sachant que vous pouvez construire les conditions que vous souhaitez, proposer pour chaque joueurs ou joueuses une alternative suivie d'une instruction qui lui sera la plus favorable.

2.2 Retour sur types et conversions

Rappel : L'opérateur permettant de forcer un changement de type, ou opérateur de *cast* en anglais, utilise une syntaxe singulière: un type entouré de parenthèses.

Exemple: `int note = (int) 15.6;`

Dans l'exemple ci-dessus, un nombre réel (15.6) est converti vers un entier (15) grace à l'opérateur de cast `(int)`. On remarque que l'opération effectuée n'est pas un arrondi à l'entier le plus proche mais bien une troncature, c'est-à-dire qu'on ne garde que la partie entière.

ATTENTION l'opérateur de forçage de type est plus prioritaire que les opérations arithmétiques, soyez donc attentifs au parenthésage de vos expressions !

Exercice 5 : De braves types

Dans cet exercice, il s'agit de remplir le tableau ci-dessous qui, pour chaque ligne, associe une expression de départ, une expression d'arrivée, leur type respectif et une opération de conversion qui permet de passer de l'une à l'autre. Si une conversion vous semble impossible, vous pouvez remplir les cases correspondantes par des croix.

	Expression `iJava` de départ	Type de départ	Opération de conversion (forçage de type)	Type d'arrivée
a	`'A'`		`(int) 'A'`	
b	`9.99`			`int`
c	`32`			`char`
f	`3.14 * 7`			`int`
i	`charAt("MOT",1)`			`int`
j	`'A' + random() * 10`			`char`
m	`n` (une variable définie plus tôt)	`int`		`char`

1. Il n'est pas possible, par une opération de conversion, de passer directement de la chaîne de caractères `"T"` au caractère `'T'`. Peut-on y parvenir autrement ?
2. Et inversement, comment peut-on passer du caractère `'T'` à la chaîne de caractères `"T"` ?