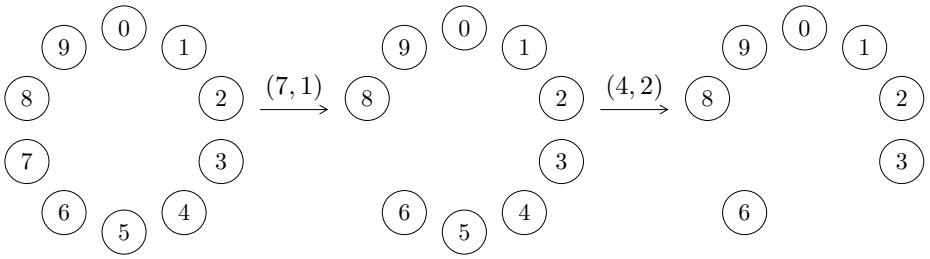
Tâche 1.1 – Pièces (coins) – (100 pts)

Plaçons n pièces de monnaie en cercle sur la table et numérotons les de 0 à $n-1$ dans le sens horaire. On joue ensuite un jeu à deux joueurs, jouant chacun à leur tour (le joueur 1 commence). À chaque tour, le joueur dont c'est le tour doit prendre une ou deux pièces de monnaie sur la table. Si le joueur choisit de prendre deux pièces, elles doivent être l'une juste à côté de l'autre. Le gagnant du jeu est celui qui prend la dernière pièce de monnaie.

Voici une exemple de début de partie. Le premier nombre correspond à l'indice de la première prise (dans l'ordre horloger). Le second nombre est le nombre de pièces prises.



Tâche

Dans cette tâche, on vous demande d'implémenter la stratégie du joueur 2 dans ce jeu afin qu'il puisse gagner dans tous les cas.

Limites et contraintes

La taille n est comprise entre 10 et 100 (inclus), par ailleurs :

Sous-tâche A (25pts)	Le nombre de pièces est pair, le joueur 1 joue de façon aléatoire
Sous-tâche B (25pts)	Le nombre de pièces est pair, le joueur 1 joue avec un meilleur niveau
Sous-tâche C (25pts)	Le nombre de pièces est pair, le joueur 1 joue de façon optimale
Sous-tâche D (25pts)	Le nombre de pièces est impair, le joueur 1 joue de façon optimale

Le score est calculé comme suit : il y a 25 parties différentes dans chacune des sous-tâches. Vous marquez 1 point pour chaque partie gagnée par le joueur 2.

Fonction à implémenter

`int play(int i, int last_coin, int last_nb, int size)` : Cette fonction prend en arguments :

i Le numéro de tour de jeu, vous recevrez donc 2, 4, 6, ...

last_coin L'indice des pièces qui viennent d'être choisie par le joueur 1. Si il a choisi deux pièces, c'est l'indice de la première pièce dans le sens horloger.

last_nb Le nombre de pièces qui viennent d'être choisies par le joueur 1 : 1 ou 2.

size Le nombre de pièces au départ du jeu.

Cette fonction doit retourner le jeu que le joueur 2 va effectuer sous la forme d'un entier, utilisez la fonction `int playValue(int coins_index, int coins_nb)` pour le générer. Celle-ci prend comme arguments :

coins_index L'indice des pièces que le joueur 2 choisit. Si il a choisi deux pièces, c'est l'indice de la première pièce dans le sens horloger.

coins_nb Le nombre de pièces choisies par le joueur 2 : 1 ou 2.

Exemple

Considérons l'exemple ci-dessus.

Voici une partie possible :

Vous recevez	Vous retournez	Pièce(s) joueur 1	Pièce(s) joueur 2
$play(2, 7, 1, 10)$	$playValue(4, 2)$	7	4, 5
$play(4, 2, 2, 10)$	$playValue(8, 2)$	2, 3	8, 9
$play(6, 0, 2, 10)$	$playValue(6, 1)$	0, 1	6

Dans ce cas, le joueur 2 a gagné car il a pris la dernière pièce.

Notes pratiques

Cette tâche est interactive. Vous ne pourrez pas utiliser la compilation et le test automatique de gedit pour cette tâche. Suivez les étapes suivantes pour compiler et tester manuellement votre programme. Demandez de l'aide à un surveillant si nécessaire.

- Ouvrez un terminal (l'application LXTerminal)
- Pour afficher la liste des fichiers dans un répertoire, utilisez la commande `ls`
- Pour changer de répertoire vers le squelette de cette tâche, utilisez la `cd nom_du_répertoire`

Pour ceux qui utilisent C++, les commandes sont les suivantes :

- Compilation : `g++ -std=c++11 -Wall grader.cpp coins.cpp`
- Exécution : `./a.out < input.txt`

Pour ceux qui utilisent Java, les commandes sont les suivantes :

- Compilation : `javac grader.java`
- Exécution : `java grader < input.txt`

Le résultat de votre programme sera ainsi affiché à la console : "Player lost" si vous (joueur 2) avez perdu la partie et "Player won" si vous avez gagné la partie. Si vous voulez modifier le fichier `input.txt`, sachez qu'il contient le nombre de pièces, une graine pour le jeu aléatoire du joueur 1 (changer le pour avoir un jeu différent de la part du joueur), et le niveau du joueur (mais uniquement le jeu aléatoire est donné dans l'évaluateur que vous avez afin de ne pas révéler la stratégie optimale).

Remarques

— Il est interdit d'appeler une fonction de génération de nombre aléatoire pour résoudre ce problème.
— La solution est relativement courte. Si vous êtes en cours d'écriture d'une longue stratégie de jeu, c'est probablement que vous êtes sur une fausse piste.
— Vous ne devez soumettre que le fichier `coins.cpp` ou `coins.java`. Il doit implémenter la fonction, `play`, telle que décrite ci-dessus.
— Si lors de votre soumission sur le serveur, vous obtenez "Résultat incorrect", cela signifie typiquement que vous avez perdu.
— Si lors de votre soumission sur le serveur, vous obtenez que le code de retour est différent de zéro, cela signifie typiquement que vous avez fait une action interdite dans les règles du jeu.
— Vous pouvez bien entendu considérer que le joueur 1 va toujours respecter les règles du jeu.
— N'imprimez rien à la console (`print`) dans le programme que vous soumettez.