

# Jeu du Plus-Moins

Le jeu du plus-moins consiste à deviner un nombre choisi aléatoirement.

Le joueur commence par choisir une valeur maximale. Le programme tire aléatoirement un nombre à découvrir compris entre 0 et cette valeur maximale.

Le joueur doit trouver le nombre à découvrir en le moins d'essais possible. A chaque essai :

- Le joueur saisie une valeur.
- Cette valeur est comparée avec le nombre à découvrir :
  - Si la valeur est inférieure au nombre à découvrir, le programme affiche « Plus »,
  - Si la valeur est supérieure au nombre à découvrir, le programme affiche « Minus »,
  - Si la valeur est égale au nombre à découvrir, le joueur a gagné.

Lorsque le joueur gagne, le programme affiche le nombre de coups joués pour trouver le nombre à .

## I – Implémentation simple :

Implémenter les règles du jeu directement dans la fonction `main`.

Pour le tirage aléatoire, inclure `<cstdlib>` et `<ctime>` au fichier source. On utilisera alors les fonctions suivantes :

```
- srand(time(NULL)); // Random seed initialization (must be called once)
- rand(); // Random int between 0 and RAND_MAX (excluded)
```

On utilisera enfin modulo pour obtenir une valeur entre 0 et la valeur maximale :

```
int valueToFind = randomValue % (maxValue + 1) ;
```

Pour les entrées et sorties, inclure `<iostream>`. On utilisera les fonctions suivantes :

```
- int playerInput;
  std::cin >> playerInput; // Wait for input. Cast and copy value to variable
- std::cout << "text" << std::endl; // Display to console
```

## II – Fonction :

Réaliser la fonction suivante :

- **CompareAndDisplay** : Compare deux valeurs entières. Affiche « Plus », « Minus » ou « Equal » en fonction de si la première valeur est inférieure, supérieure ou égale à la seconde.
  - Paramètres en entrée : Valeur à comparer, valeur de comparaison
  - Valeur de retour : 1 si la valeur à comparer est inférieure à la valeur de comparaison, -1 si elle est supérieure, 0 si elle est égale.

Adapter le code de la fonction **main** en utilisant la fonction **CompareAndDisplay**. Utiliser la valeur de retour de **CompareAndDisplay** pour vérifier si la partie est terminée.

## III – IA :

La stratégie optimale pour gagner le jeu du plus-moins est la déduction par dichotomie. Cette stratégie consiste à retenir les bornes maximales et minimales entre lesquelles le nombre à deviner est compris.

Au lancement du jeu :

- la borne minimale est initialisée à 0,
- La borne maximale est initialisée à la valeur maximale possible du tirage.

A chaque essai de déduction, l'IA doit :

- Proposer une valeur au milieu de ces deux bornes :  $(\text{borne min} + \text{borne max}) / 2$ ,
- En fonction du résultat du tour :
  - « Plus » : mettre à jour la borne minimale à la valeur proposée + 1,
  - « Minus » : mettre à jour la borne maximale à la valeur proposée – 1

En début de jeu, demander au joueur si le programme doit utiliser l'IA. Le joueur doit répondre par « y » ou par « n ». Si le joueur répond autre chose que « y », l'IA n'est pas utilisée.

Implémenter l'IA du jeu directement dans la fonction **main**. Remplacer les inputs du joueur par les inputs de l'IA dans le cas où l'IA est active.

## IV – Paramètres et références :

Implémenter une fonction permettant de mettre à jour les bornes de l'IA :

- **UpdateBounds** : Met à jour les bornes de l'IA à partir de la valeur proposée et du résultat de l'essai.
  - Paramètres en entrée : Valeur proposée, résultat de l'essai, borne min à mettre à jour, borne max à mettre à jour
  - Valeur de retour : /

Adapter le code de la fonction **main** en utilisant la fonction **UpdateBounds**.  
Pour réaliser cette fonction, utiliser le passage de variables en paramètre par références.

## V – Structure :

Les structures permettent d'organiser les variables du code en ensemble de variables.

Dans ce projet, nous souhaitons regrouper toutes les variables relatives au fonctionnement de l'IA dans une structure `AIData` contenant les variables membre suivantes :

- `isActive` : Vrai si l'IA est active, sinon faux
- `boundMin` : La borne minimale
- `boundMax` : La borne maximale

Implémenter le type `struct aiData` avec comme synonyme `AIData`. Adapter le code existant en utilisant cette structure.