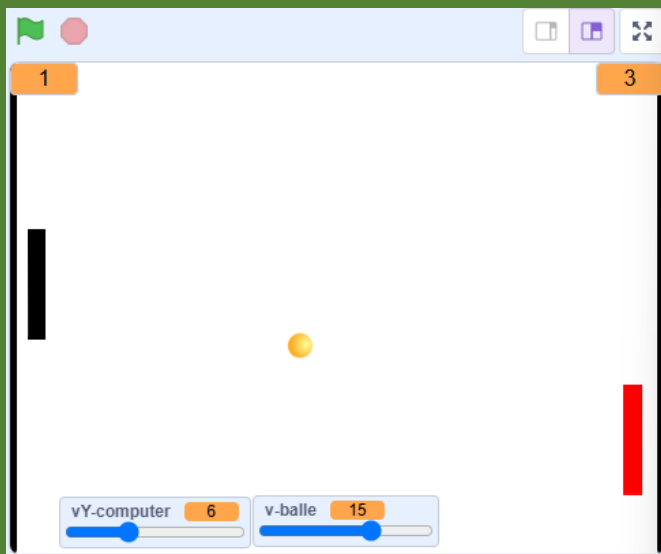




## 3 Jouer à Pong contre l'ordinateur



Programme un jeu de Pong contre l'ordinateur.  
Améliore ton jeu pour qu'il soit passionnant et réaliste !



▼ Vidéo

<https://jeunesingenieurs.be/vid-s3-3>



▼ Proposition de solution

<https://scratch.mit.edu/projects/1099399426>





# En mode défi

## ▼ Tu te lances en mode Défi ?

Voici une proposition de méthode.

1. .Crée les sprites : une balle, deux raquettes et deux murs de fond de jeu.
2. Fais circuler la balle de façon à ce qu'elle rebondisse sur les murs.
3. Actionne la raquette du joueur en suivant le pointeur de la souris et celle de l'ordinateur en suivant la position verticale de la balle.
4. Programme le rebond sur les raquettes.
5. Programme le scoring et la fin du jeu.
6. Améliore le comportement de la raquette de l'ordinateur de manière à ce que celui-ci puisse être battu.
7. Améliore ce jeu à l'aide de tes propres trouvailles.

En manque d'inspiration ? Il y a quelques idées en fin de ce jeu et ce projet :

<https://scratch.mit.edu/projects/1097781862>



## 3.1 LES SPRITES - LA BALLE SE DÉPLACE

### 🎯 Objectif : prépare les sprites

Charge et/ou dessine les objets à programmer : une balle, deux raquettes (une joueur et une ordinateur) et les deux murs. Place-les comme indiqué sur l'illustration de la carte 3-1 . Les raquettes font +/- 14 x 80 pixels.



Réduis la taille de la balle à 40 %

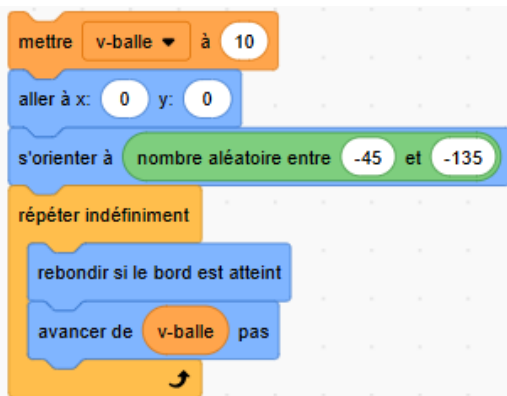
### 🎯 Objectif : lancer la *balle*

- La *balle* se place au centre,
- elle s'oriente vers le bord, côté joueur. Sa direction est fixée aléatoirement de façon à atteindre le mur.
- elle rebondit sur les bords.

Crée et initialise une variable

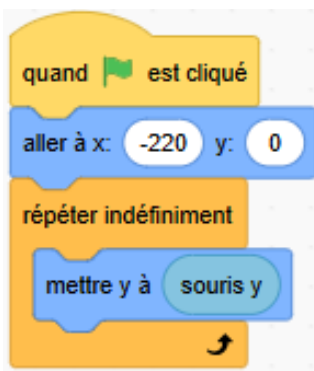
*v-balle*.

▀ Ajoute ce code à la *balle* et teste-le



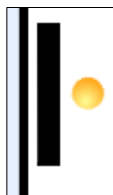


## 3.2 ACTIONNER LES RAQUETTES



🎯 Objectif : déplacer la raquette du joueur

La raquette du joueur suit le pointeur de la souris.

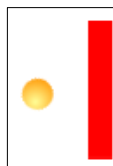


▼ Ajoute ce code à la raquette joueur et teste-le

La raquette va accompagner le mouvement vertical de la souris.

🎯 Objectif : positionner la raquette de l'ordinateur

Dans un premier temps, on veille à ce que la raquette de l'ordinateur se positionne de façon à intercepter la balle dans tous les cas



▼ Ajoute ce code à la raquette de l'ordinateur

Avec ce code, l'ordinateur sera imbattable...

Nous y reviendrons plus loin.



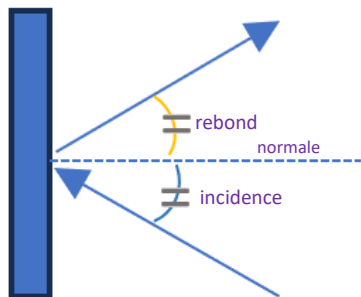
## 3.3 PROGRAMMER LE REBOND

🎯 **Objectif** : quand la balle touche une raquette, elle rebondit.

Le rebond doit respecter les lois de la Physique : l'angle d'incidence égale l'angle de rebond.

La référence est le plan de la raquette et sa perpendiculaire : la normale.

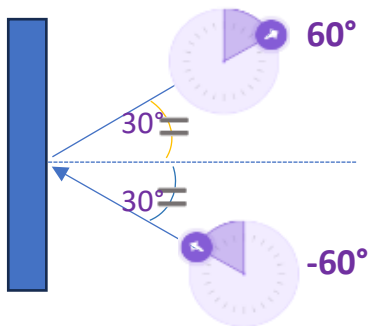
C'est la même règle pour un rayon lumineux réfléchi par un miroir.



### ▼ Trop simple...

L'orientation du sprite est l'angle complémentaire de l'angle de rebond.

Quand une balle arrive sur la raquette orientée à  $-60^\circ$ , elle rebondit orientée à  $+60^\circ$ ...



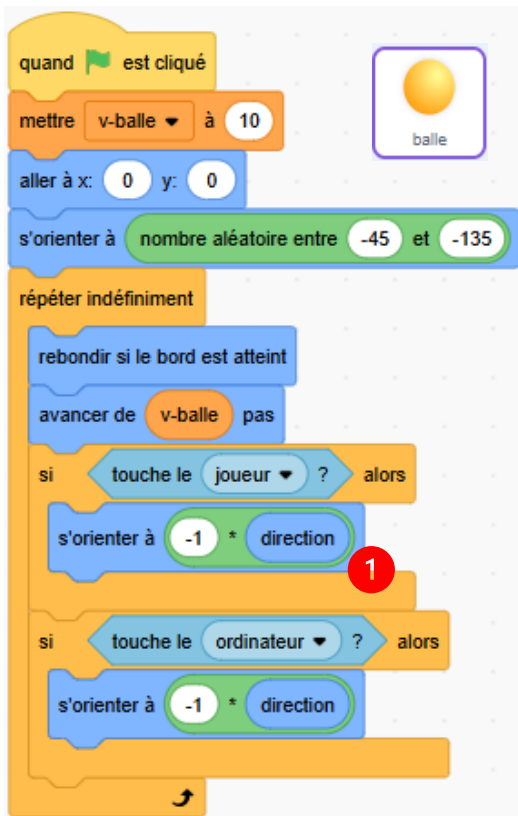


## 3.3 PROGRAMMER LE REBOND

### Modifie le code de la balle

Le calcul sera donc très simple. L'orientation avant le rebond égale l'opposé de l'orientation après le rebond.

Et pour obtenir l'opposé d'un nombre, il suffit de le multiplier par -1 .



La balle arrive orientée à  $-60^\circ$  et au rebond, elle s'oriente à

①  $(-1) \times (-60) = 60$ .

### Teste ce code.

Le rebond est programmé de la même façon pour les deux raquettes.



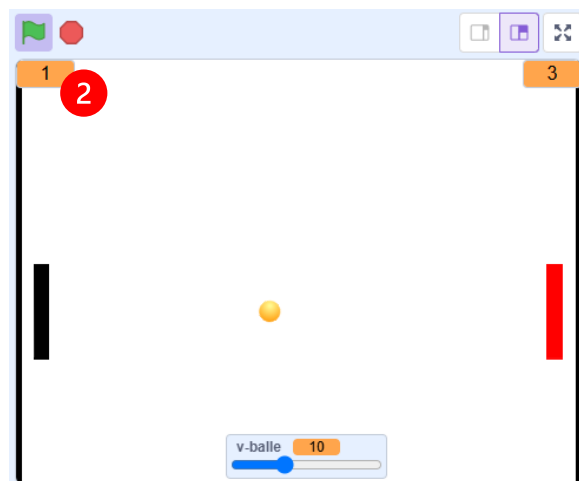
## 3.4 LE SCORE

### 🎯 Objectif : programmer un score

Chaque joueur gagne un point quand la balle touche le mur adverse. Le jeu se termine quand un des deux joueurs atteint le score 5. On relance le jeu chaque fois que le score est modifié.

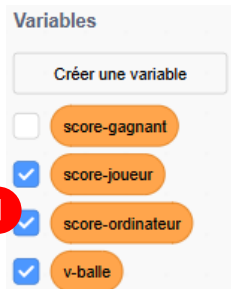
#### 📌 Crée et affiche les variables

Crée et affiche trois nouvelles variables :

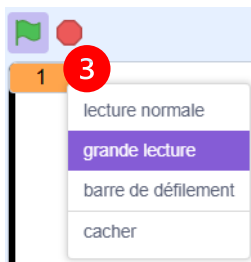


variable affichée sur la scène et ③ choisis le mode **grande lecture**.

La variable **score-gagnant** ne doit pas être affichée.



Pour modifier le mode d'affichage des variables, ① affiche-les à l'aide de la case à cocher, ② clique-droit sur la





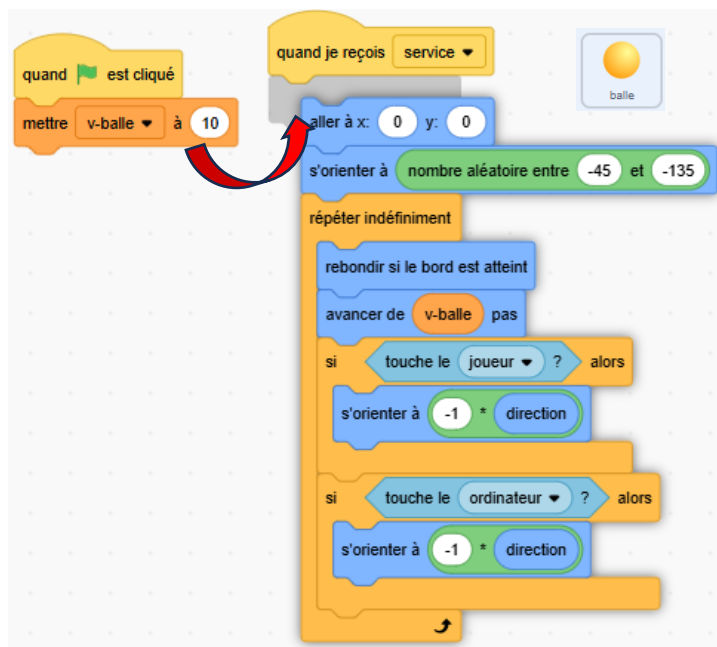
## 3.4 LE SCORE

### ■ Crée un script *service*

Le script *service* doit être démarré au début du jeu et chaque fois qu'un point est joué.

Le nouveau message *service* est créé et on associe le script de la balle au bloc *quand je reçois service*.

Pour la balle, nous allons avoir deux scripts. Un qui démarre quand on clique sur le *drapeau vert* et un deuxième qui s'exécute quand le message *service* est reçu





## 3.4 LE SCORE

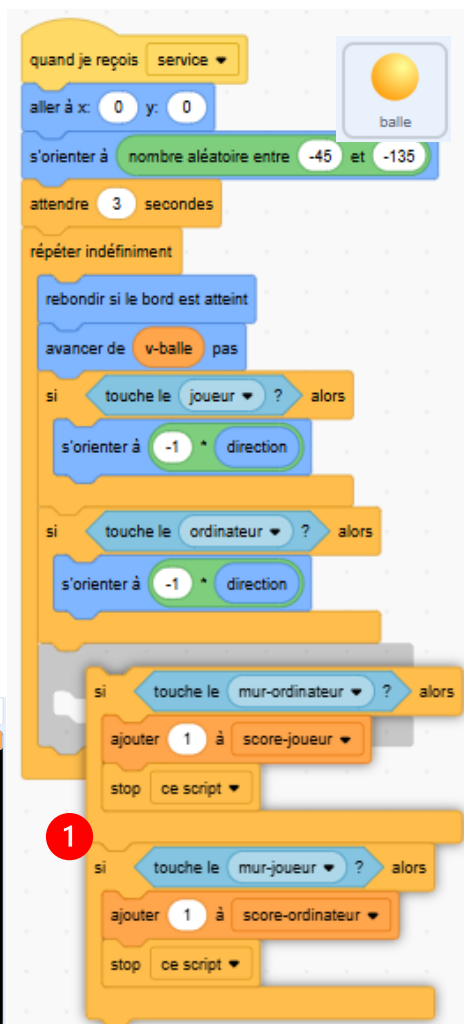
### Objectif

Les deux joueurs reçoivent un point quand le mur adverse est touché.

➤ 1 Ajoute ce code au script de la balle

➤ Prépare deux arrière-plans

L'arrière-plan 1 avec le texte « tu as gagné » et l'arrière-plan 2 avec le texte : « tu as perdu ».





## 3.5 FIN DU JEU

### Objectifs

Remettre la balle en service jusqu'à ce qu'un score gagnant soit atteint (par exemple 5) et terminer le jeu.

Complète le script **drapeau vert** pour finaliser la gestion du score



The image shows a Scratch script for the 'when green flag clicked' event. The script includes several 'set' blocks for variables: 'v-balle' to 10, 'score-gagnant' to 5, 'score-ordinateur' to 0, and 'score-joueur' to 0. A 'repeat until' loop contains a 'send to all service and wait' block, which is highlighted with a red circle and a '1' in the text. The loop is followed by an 'if-then-else' block: if 'score-joueur' equals 5, then 'switch to back layer arriere-plan2', else 'switch to back layer arriere-plan1'. Finally, there is a 'switch to costume ball-a' block and a 'stop all' block. A small 'balle' sprite is shown in the top right corner.

1 Cette boucle relance le jeu jusqu'à ce qu'un des deux joueurs obtienne le score de 5 points.

Le bloc **envoyer à tous ... et attendre** passe la main au script **quand je reçois...** et attend que celui-ci soit terminé pour s'exécuter à nouveau.



## 3.6 TROP FORT L'ORDINATEUR

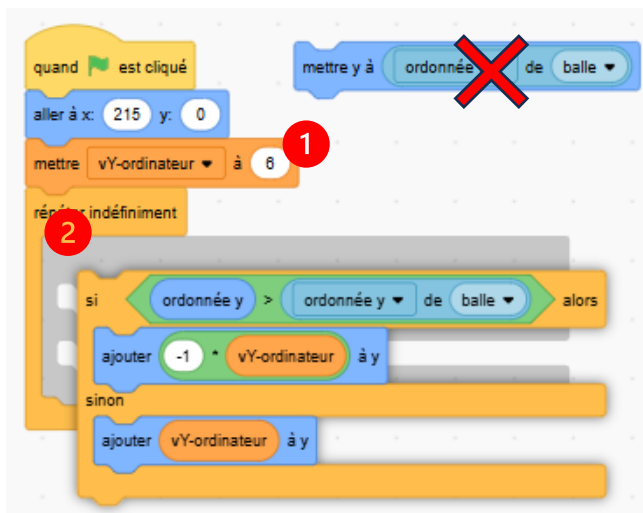
### 🎯 Objectif : affaiblir le jeu de l'ordinateur

Le jeu est peu intéressant car l'ordinateur est imbattable. Il faudrait que son comportement soit plus réaliste et que le joueur puisse le battre. Nous proposons deux adaptations pour atteindre cet objectif. On pourrait (tu pourrais... 😊) en trouver plein d'autres.

1. Limiter la vitesse de déplacement de la raquette de l'ordinateur
2. Empêcher l'ordinateur de voir et de suivre la balle quand elle est dans le camp adverse

### 📌 Limiter la vitesse de la raquette de l'ordinateur

Créer une variable vY-ordinateur et l'initialiser à 6. Supprimer le bloc **mettre y à** ... et le remplacer par **2** ce code qui a le même effet mais permet de modifier la vitesse.



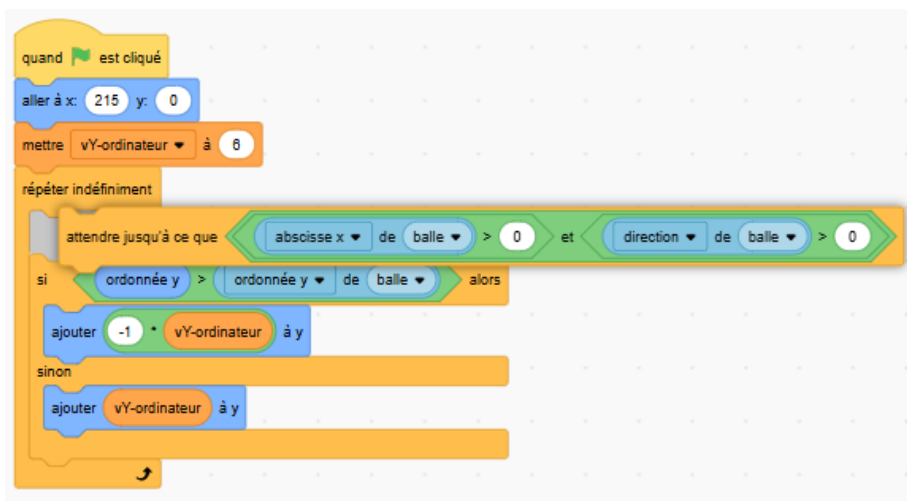


## 3.6 TROP FORT L'ORDINATEUR

### ■ L'ordinateur ne peut voir la balle que quand elle arrive dans son camp

Une deuxième option consiste à n'autoriser la raquette de l'ordinateur à réagir que quand la balle arrive dans son camp.

Ajoute un bloc :



Il aura pour effet de laisser l'ordinateur sans réaction jusqu'au moment où la balle reviendra vers lui et passera dans son camp.



## 3.7 À TOI DE JOUER

### 🎯 Apporter quelques améliorations

Tu as sûrement des idées pour améliorer ce jeu.  
N'hésite pas à t'y mettre !

Voici quelques suggestions :

- accélérer progressivement la vitesse de la balle pour mettre les joueurs sous pression.
- ajouter une option « faux bond » qui modifie de temps en temps et de façon aléatoire la trajectoire de la balle lors du rebond.
- améliorer le départ du jeu avec un décompte.
- enregistrer les résultats de joueurs dans une liste et les afficher en début de jeu.
- améliorer la gestion du contact de la balle avec les raquettes



Si tu stoppes le déplacement de la balle quand elle touche la raquette, tu constateras qu'elle ne rebondit pas sur le bord mais pénètre dans la raquette. Si on augmente la vitesse, la balle peut dans certains cas passer à travers.

Trouve le moyen de résoudre ce problème en améliorant la gestion de la collision.