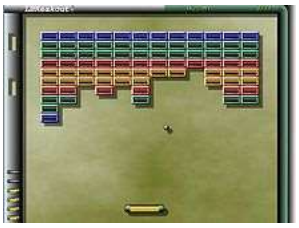
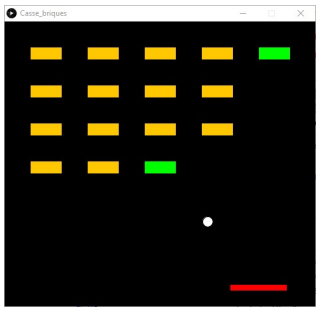
1. Introduction



Le **casse-briques** est un genre de jeu vidéo souvent classé dans la catégorie arcade, apparu en 1975 avec le jeu *Breakout*. Il est directement inspiré de *Pong*.

2. Jeu Casse-Briques

À l'aide du logiciel Processing, implémenter le jeu Casse-Briques.



2.1. Création du mur de briques

Les briques seront des objets.
✍ Créer une class Brique.
✍ Définir dans la méthode d'initialisation les dimensions de l'objet **`Brique`** (attributs **`largeur`** et **`hauteur`**).
✍ Définir une méthode **`dessine_brique`** qui dessine une brique à partir des attributs de dimensions de la brique et des variables de la position de la brique (cordonnées **`x`** et **`y`**).

Les objets **`Brique`** sont stockés dans un tableau à deux dimensions (**`tab[i][j]`**) en lien avec le mur de briques à créer.
✍ Créer un tableau vide de 4 lignes et de 5 colonnes.
✍ Dans chaque case du tableau, stocker un objet **`Brique`**.

✍ Dans la fonction **`draw()`**, dessiner le mur de briques :

```
for i in range(len(tab)):
        for j in range(len(tab[i])):
            tab[i][j].dessine_brique(j*90 + 65, i*60 + 50)
```

2.2. Balle et raquette

✍ Dessiner la raquette et la mettre en mouvement.
✍ Dessiner la balle et la mettre en mouvement (ne pas gérer les rebondissements sur les briques pour l'instant).

2.3. Modification de l'affichage du mur de briques

Lorsqu'une brique est touchée alors il ne faut plus l'afficher.
L'attribut **`etat`** (égal à 1 lors de l'initialisation) passera à zéro si la brique est touchée.
Par conséquent, l'affichage de la brique se fait si son attribut **`etat`** est égal à 1.
✍ Inclure un attribut **`etat`** dans la méthode d'initialisation égal à 1.
✍ Modifier la méthode **`dessine_brique`**, afin de dessiner la brique que si son attribut **`etat`** est différent de 0.

2.4. Gestion de la collision avec les briques

La méthode **collision** de la class **Brique** doit renvoyer comme paramètres la valeur de déplacement (sens) de la balle (déplacement en x et déplacement en y).

✍ La méthode collision répond à l'algorithme suivant :

```
si la balle a touché la brique et que etat ≠ 0 alors
    décrémenter etat
    min_x = distance mini entre la balle et la brique en x
    min_y = distance mini entre la balle et la brique en y
    si min_x < min_y alors
        changer la valeur de déplacement en y
    sinon
        changer la valeur de déplacement en x
    finsi
    retourner les valeurs de déplacement en x et en y
finsi
retourner les valeurs de déplacement en x et en y
```

✍ Dans la boucle for de la méthode **draw()**, inclure l'appel à la méthode collision sur les briques.

2.5. Améliorations

- La brique est effacée si elle est frappée plusieurs fois (se servir de son attribut **etat**).
- Inclure un compteur de points
- Inclure 3 vies
- Modifier l'angle de rebond sur la raquette en fonction de la position de collision de la balle sur la raquette.
- …