1 | Simulations

Dans ce projet, nous avons développé deux simulations interactives en utilisant le langage de programmation Python, principalement en exploitant la bibliothèque `Pygame`.

`Pygame` offre une gamme d'outils pour créer des applications multimédias interactives, y compris des jeux et des simulations graphiques, ce qui en fait un choix adapté pour nos besoins.

1.1. Simulation de la bille en mouvement

1.1.1. Algorithme de la simulation

L'algorithme de la simulation de la bille en mouvement est conçu pour créer une interface utilisateur graphique où une balle se déplace horizontalement à travers l'écran. Voici les étapes clés de l'algorithme :

- Créer une fenêtre graphique avec une balle et un bouton "Démarrer".
- Lorsque le bouton "Démarrer" est pressé, démarrer une minuterie pour mettre à jour la position de la balle.
- Calculer la vitesse de la balle en fonction du nombre de va-et-vient prévus et de la durée totale du mouvement.
- Changer la direction de la balle lorsqu'elle atteint les bords de l'écran.
- Utiliser des minuteries pour afficher des images fixes à des moments spécifiques pendant le mouvement de la balle.
- Arrêter la balle après la durée totale du mouvement.

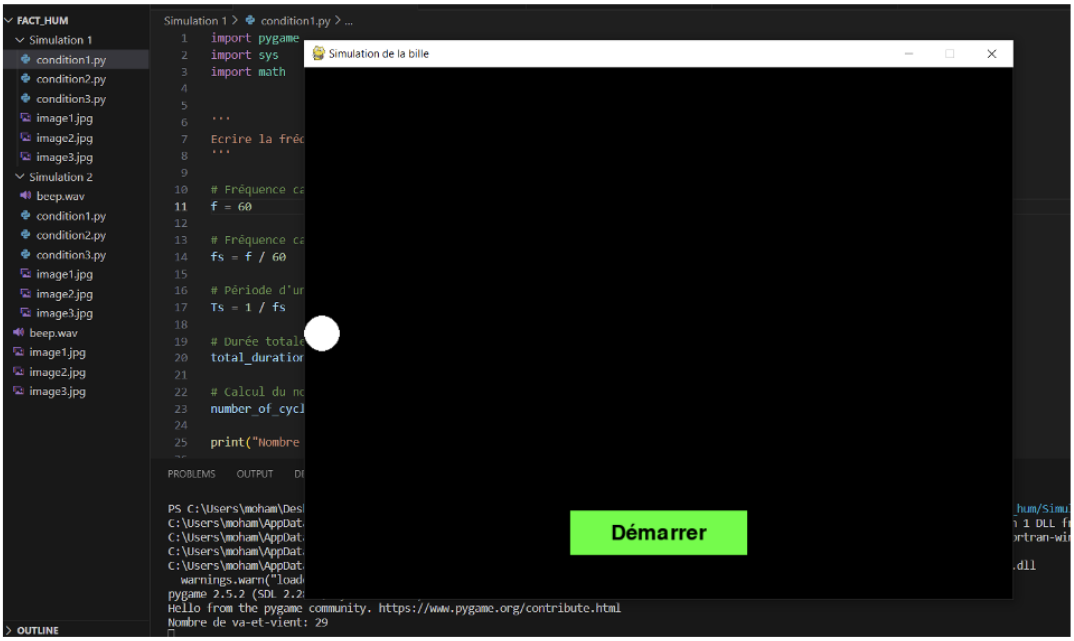

FIGURE 1.1 – Interface de démarrage pour la simulation 1

1.1.2. Explication du programme

Le programme commence par initialiser les variables nécessaires en fonction des paramètres spécifiés, notamment la fréquence cardiaque (f) en battements par minute (BPM). Cette fréquence cardiaque est ensuite convertie en fréquence par seconde (f_s) et utilisée pour calculer la période d'un va-et-vient (T_s) en secondes. La durée totale du mouvement de la balle (`total_duration`) est définie à 29 secondes.

Le nombre de va-et-vient (nombre de cycles) est calculé en divisant la durée totale de déplacement par la durée d'un va-et-vient (T_s). Cette formule mathématique est représentée comme suit :

$$\text{nombre de cycles} = \frac{\text{durée totale de déplacement}}{\text{durée d'un va-et-vient}} = \frac{29}{T_s}$$

Pour chaque condition expérimentale, le programme calcule la vitesse de la balle en fonction du nombre de va-et-vient prévus, de la taille de la fenêtre (pixels), et de la durée totale du mouvement. La formule de calcul de la vitesse (v) est donnée par :

$$v = \frac{\text{nombre de va-et-vient} \times 2 \times (\text{pixels} - 2 \times \text{ball\_size})}{\text{durée totale}}$$

- Dans la condition 1 (périodique synchrone), la fréquence cardiaque (f) est déterminée après avoir mesuré la fréquence du sujet. Cette fréquence est utilisée pour calculer la vitesse de la balle selon la formule précédente.

- Dans la condition 2 (périodique asynchrone), la fréquence cardiaque (f) est fixée à 100 BPM. La vitesse de la balle est calculée de la même manière que dans la condition 1, en utilisant la fréquence fixe.

- Dans la condition 3 (apériodique synchrone), la fréquence cardiaque (f) est déterminée après avoir mesuré la fréquence du sujet, ce qui influence le nombre de va-et-vient prévus. Pour introduire de l'aléatoire et assurer l'apériodicité, la vitesse de la balle peut être ajustée en ajoutant ou soustrayant une fluctuation à la vitesse standard calculée.

  $$V = V_{\text{standard}} \times (1 + \alpha)$$

  Avec α un nombre aléatoire entre -0.5 et 0.5.

Cette approche mathématique garantit que la vitesse de la balle est ajustée en fonction de la fréquence cardiaque du sujet et des spécifications de chaque condition expérimentale.

Des images fixes sont affichées à des intervalles spécifiques pendant le mouvement de la balle, utilisant des minuteries pour contrôler leur affichage : La séquence d'affichage des images est la suivante : 5 secondes après l'appui sur le bouton « démarrer », afficher la première image pendant 5 secondes ; ne rien afficher pendant les 3 secondes suivantes puis afficher la deuxième image pendant 5 secondes ; enfin ne rien afficher pendant les 3 secondes qui suivent et afficher la dernière image pendant 5 secondes. La simulation se termine après 4 second. Ce qui donne en totalité 30s pour chaque simulation.

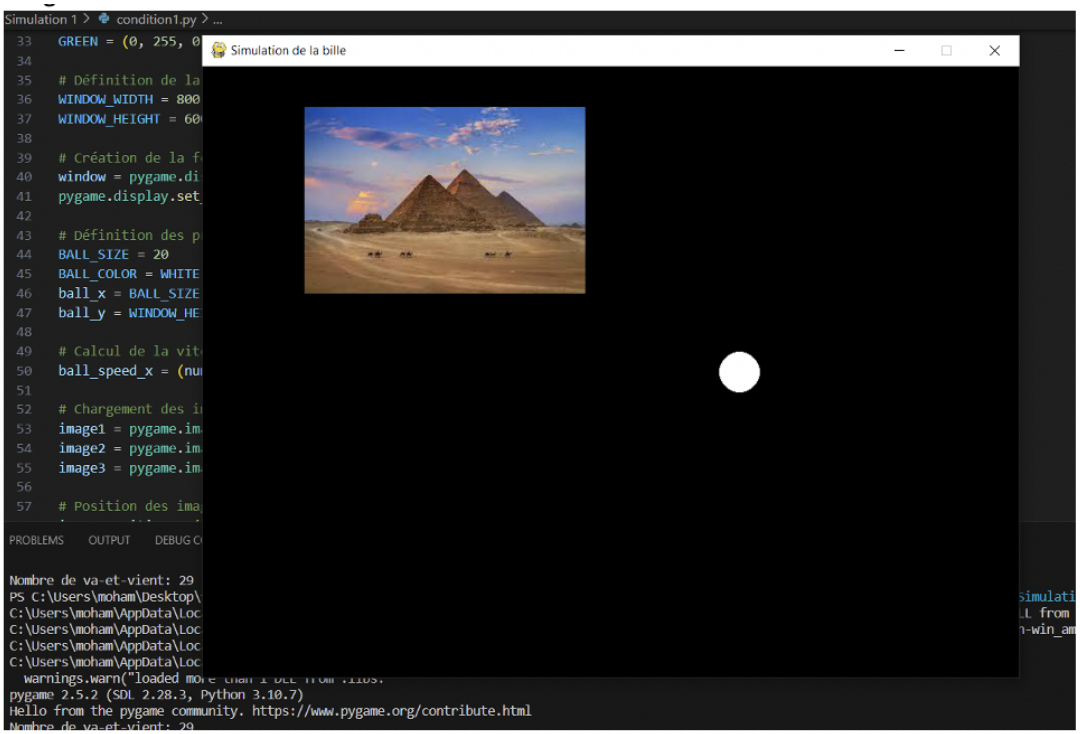

FIGURE 1.2 – Apparition d'une image dans la simulation 1

1.2. Simulation du carré clignotant et bipant

1.2.1. Algorithme de la simulation

L'algorithme de la simulation du carré clignotant et bipant vise à créer une interface utilisateur graphique présentant un carré qui clignote et émet un bip à des intervalles spécifiques. Voici les étapes clés de cet algorithme :

- Initialiser l'environnement graphique en créant une fenêtre avec un carré et un bouton "Démarrer".
- Lorsque le bouton "Démarrer" est pressé, démarrer une minuterie pour contrôler le clignotement du carré et les bips.
- Calculer le nombre de clignotements et de bips en fonction de la fréquence cardiaque du sujet.
- Définir la fréquence de clignotement et de bip en fonction du nombre de va-et-vient prévus.
- Contrôler le clignotement du carré en alternant entre l'affichage et la non-affichage à des intervalles spécifiques.
- Émettre un bip à chaque intervalle de clignotement du carré.

— Arrêter le clignotement et les bips après 29 secondes.

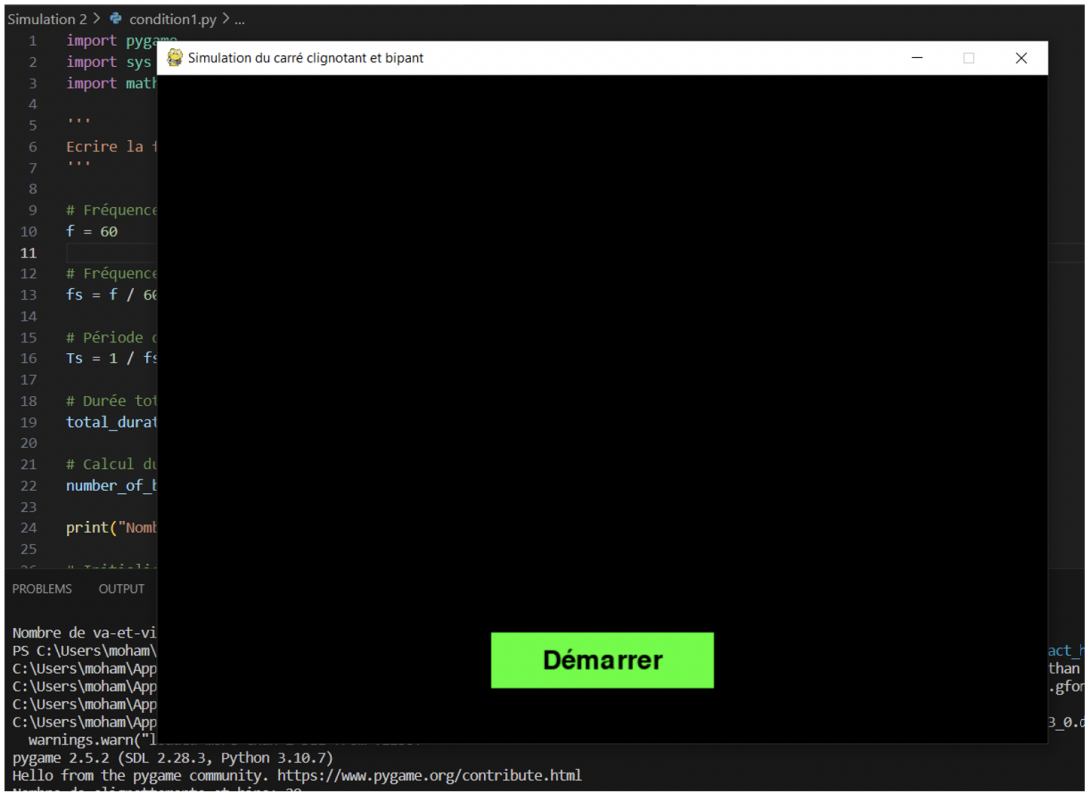

FIGURE 1.3 – Interface de démarrage pour la simulation 2

1.2.2. Explication du programme

Le programme commence par initialiser les variables nécessaires en fonction des paramètres spécifiés, notamment la fréquence cardiaque (f) en battements par minute (BPM). La fréquence cardiaque est ensuite convertie en fréquence par seconde (f_s) pour permettre le calcul de la période d'un battement en secondes (T_s).

Le nombre de va-et-vient (nombre de bips) du carré est calculé en divisant la durée totale de l'expérience par la période d'un battement (T_s). Cette formule mathématique est représentée comme suit :

$$\text{nombre de bips} = \frac{\text{durée totale de l'expérience}}{\text{Période d'un battement}} = \frac{29}{T_s}$$

Pour chaque condition expérimentale, la fréquence cardiaque (f) est utilisée pour déterminer le nombre de bips prévus. Ensuite, la fréquence de clignotement et de bip est ajustée en fonction de ce nombre.

L'interface graphique affiche un carré qui clignote à des intervalles spécifiques contrôlés par une minuterie. À chaque intervalle de clignotement, un bip est émis pour correspondre au rythme cardiaque.

- Dans la condition 1 (périodique synchrone), la fréquence cardiaque (f) est déterminée après avoir mesuré la fréquence du sujet. Cette fréquence est utilisée pour calculer les intervalles de clignotement.
- Dans la condition 2 (périodique asynchrone), la fréquence cardiaque (f) est fixée à 100 BPM. les intervalles de clignotement sont calculés de la même manière que dans la condition 1, en utilisant la fréquence fixe.
- Dans la condition 3 (apériodique synchrone), la fréquence cardiaque (f) est déterminée après avoir mesuré la fréquence du sujet, ce qui influence les intervalles de clignotement. Pour introduire de l'aléatoire et assurer l'apériodicité, les intervalles de clignotement sont ajustés en ajoutant ou soustrayant une fluctuation à l'intervalle de clignotement standard calculée.

$$T = T_{\text{standard}} \times (1 + \alpha)$$

Avec α un nombre aléatoire entre -0.5 et 0.5.

Cet algorithme garantit que le carré clignote et émet des bips à des intervalles précis en fonction de la fréquence cardiaque du sujet, permettant ainsi de simuler différentes conditions expérimentales dans notre étude.

Des images fixes sont affichées à des intervalles spécifiques pendant le mouvement de la balle, utilisant des minuteries pour contrôler leur affichage : La séquence d'affichage des images est la suivante : 5 secondes après l'appui sur le bouton « démarrer », afficher la première image pendant 5 secondes ; ne rien afficher pendant les 3 secondes suivantes puis afficher la deuxième image pendant 5 secondes ; enfin ne rien afficher pendant les 3 secondes qui suivent et afficher la dernière image pendant 5 secondes. La simulation se termine après 4 second. Ce qui donne en totalité 30s pour chaque simulation.

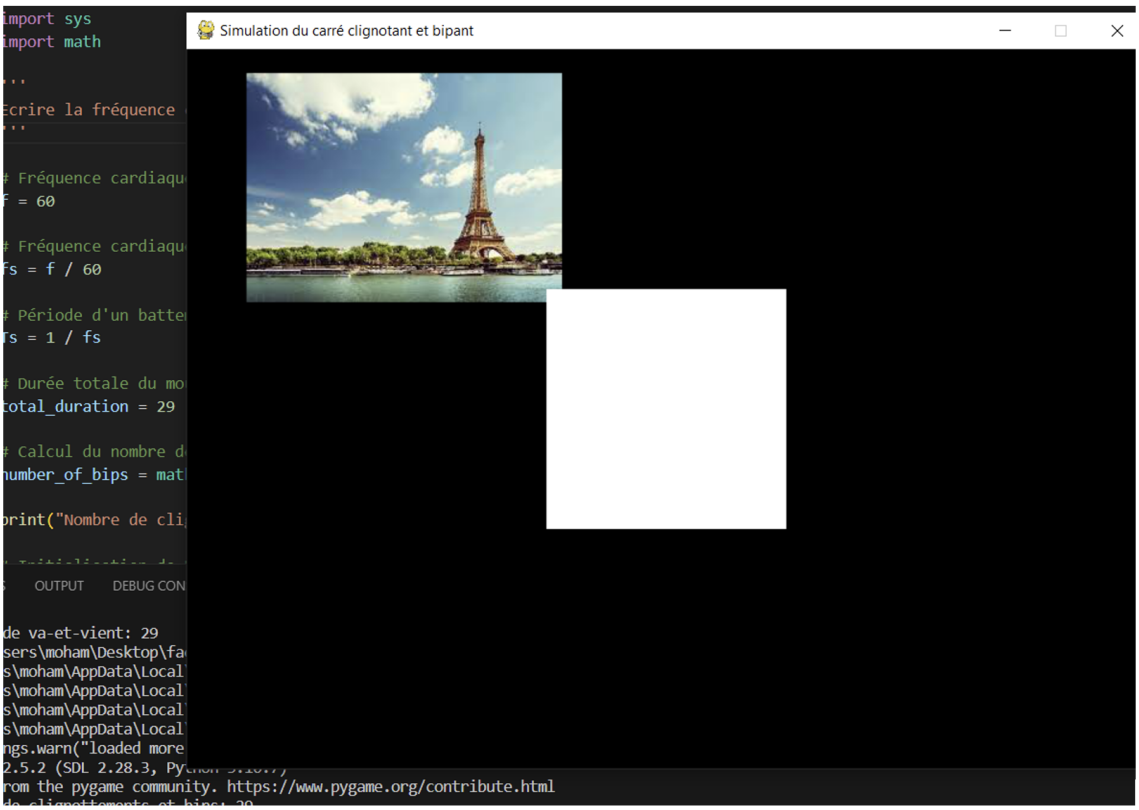

FIGURE 1.4 – Apparition d'une image dans la simulation 2

1.3. Image utisées pendant la simulation

FIGURE 1.5 – Image utisées pendant la simulation