

# **PROJET DE GROUPE**

**Présentation**

# NOTRE ÉQUIPE

Jules : Chef + Code

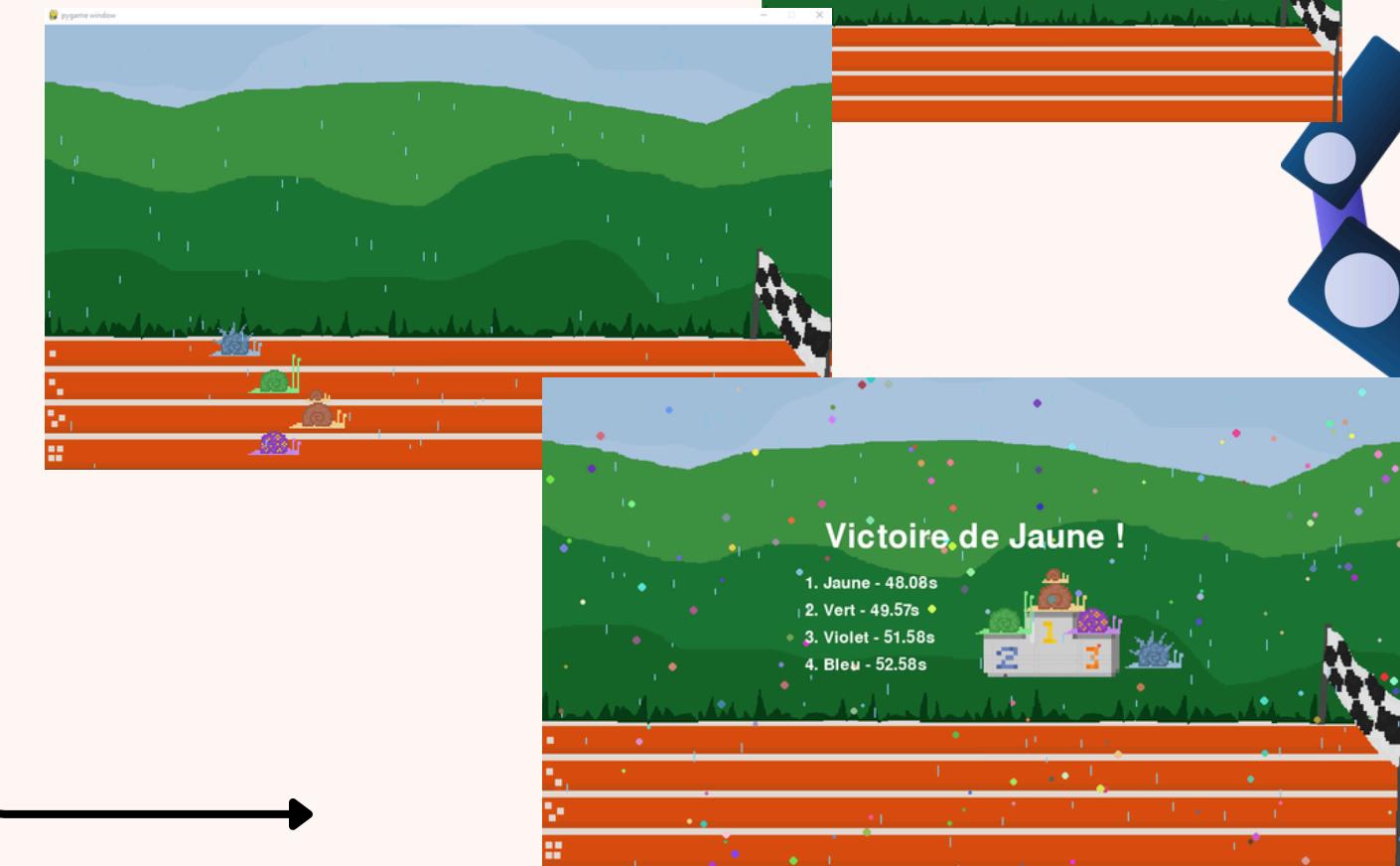
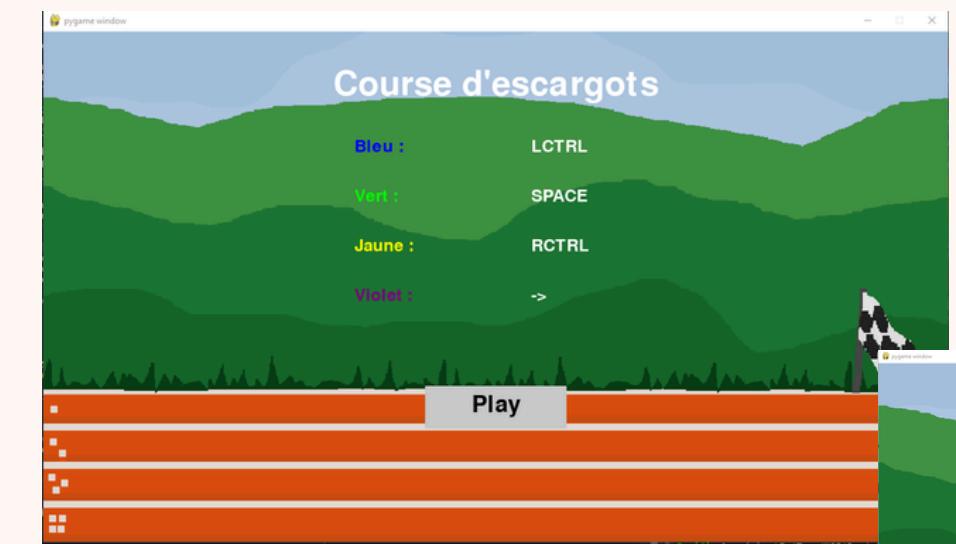
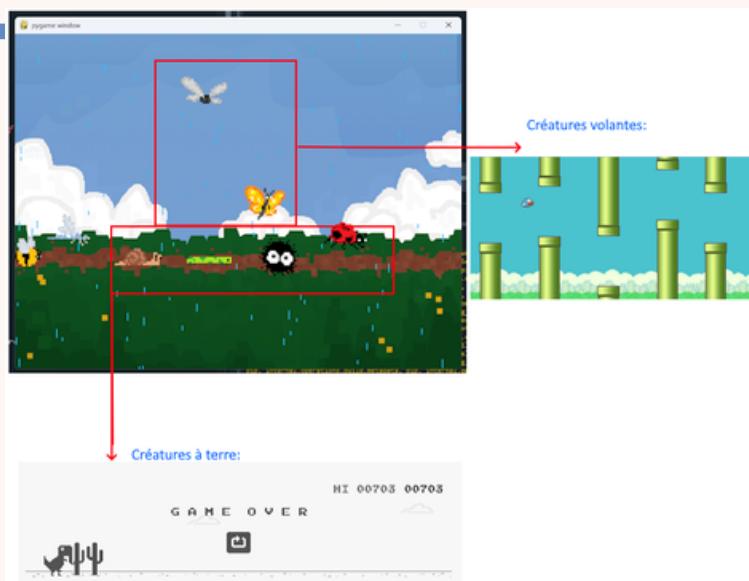
Pierre-Tengis : Code + ‘Ménage’

Marie G : Code + Planning

Marie L : Artiste + Code

Mélina : Artiste + code

# PROCESSUS



# JULES

```
keys = pygame.key.get_pressed()
if keys[pygame.K_TAB] and not prev_keys[pygame.K_TAB]:
    for escargot in listeescargots:
        escargot.x = 0
        escargot.vitesse = 10
    victoire = None
    leaderboard = None
    afficher_podium = False
    confettis_actifs = False
    compteur_frame = 0
    course_lancee = False
    decompte_actif = True
    index_decompte = 0
    compteur_pluie = 0
    pluie_active = False

if not afficher_podium:
    if keys[pygame.K_LCTRL] and not prev_keys[pygame.K_LCTRL]:
        listeescargots[0].bouger()
    if keys[pygame.K_SPACE] and not prev_keys[pygame.K_SPACE]:
        listeescargots[1].bouger()
    if keys[pygame.K_RCTRL] and not prev_keys[pygame.K_RCTRL]:
        listeescargots[2].bouger()
    if keys[pygame.K_RIGHT] and not prev_keys[pygame.K_RIGHT]:
        listeescargots[3].bouger()
prev_keys = keys
```

```
# Boucle principale
run = True
compteur_pluie = 0
compteur_frame = 0
leaderboard = None
pluie_active = False
confettis_actifs = False
prev_keys = pygame.key.get_pressed()
victoire = None
course_lancee = False
decompte = ['3', '2', '1', 'Go !']
index_decompte = 0
decompte_actif = True
font_decompte = pygame.font.SysFont(None, 100)
afficher_podium = False
podium = Podium(podium)
etat = "menu" # Peut être "menu" ou "jeu"
menu = Menu() # Instancie le menu
```

```
if etat == "menu":
    menu.dessiner()
else:
    screen.blit(bg, (0, 0))

# Gestion du décompte
if decompte_actif:
    texte_decompte = font_decompte.render(decompte[index_decompte], True, (255, 255, 255))
    screen.blit(texte_decompte, (L//2 - texte_decompte.get_width()//2, H//2 - 50))
    pygame.display.flip()
    pygame.time.wait(1000) # Attend 1 seconde
    index_decompte += 1
    if index_decompte >= len(decompte):
        decompte_actif = False
        course_lancee = True
        compteur_frame = 0
    continue # On saute le reste de la boucle pendant le décompte

keys = pygame.key.get_pressed()
if keys[pygame.K_TAB] and not prev_keys[pygame.K_TAB]:
    for escargot in listeescargots:
        escargot.x = 0
        escargot.vitesse = 10
    victoire = None
    leaderboard = None
    afficher_podium = False
    confettis_actifs = False
    compteur_frame = 0
    course_lancee = False
    decompte_actif = True
    index_decompte = 0
    compteur_pluie = 0
    pluie_active = False
```

# PIERRE-TENGIS

```
class Leaderboard():
    def __init__(self):
        self.font = pygame.font.SysFont(None, 36)
        self.resultats = []

    def ajouter(self, index, temps):
        for r in self.resultats:
            if r[0] == index:
                return
        self.resultats.append((index, temps))

    def afficher(self):
        posY = H//2 + (-70)
        for pos, (index, temps) in enumerate(self.resultats):
            sec = temps / 60
            texte = self.font.render(f"{pos+1}. {noms_escargots[index]} - {sec:.2f}s", True, (255, 255, 255))
            screen.blit(texte, (L//2 - 250, posY))
            posY += 40

class affiche_victoire():
    def __init__(self, index):
        self.index = index # On passe l'index de l'escargot gagnant
        self.font = pygame.font.SysFont(None, 72)
        self.texte = self.font.render(f"Victoire de {noms_escargots[self.index]} !", True, (255, 255, 255))

    def afficher(self):
        screen.blit(self.texte, (L//2 - self.texte.get_width()//2, 210))
```

```
class Confetti:
    def __init__(self):
        self.x = randint(0, L)
        self.y = randint(-H, 0)
        self.v = randint(2, 5)
        self.taille = randint(3, 6)
        self.couleur = (randint(50, 255), randint(50, 255), randint(50, 255))

    def bouger(self):
        self.y += self.v
        if self.y > H:
            self.y = -10
            self.x = randint(0, L)
            self.couleur = (randint(50, 255), randint(50, 255), randint(50, 255))

    def afficher(self):
        pygame.draw.circle(screen, self.couleur, (self.x, int(self.y)), self.taille)
```

```
# Dessine et déplace les gouttes si la pluie est active
if pluie_active:
    for goutte in gouttes:
        goutte.dessine()
        goutte.tombe()

if confettis_actifs:
    for confetti in confettis:
        confetti.afficher()
        confetti.bouger()
```



# MARIE G

```
noms_escargots = ["Bleu", "Vert", "Jaune", "Violet"]
touches_escargots = ["LCTRL", "SPACE", "RCTRL", "->"]

# Classe pour le menu
class Menu:
    def __init__(self):
        # Polices pour le texte
        self.font_titre = pygame.font.SysFont(None, 72)
        self.font_texte = pygame.font.SysFont(None, 36)
        self.font_bouton = pygame.font.SysFont(None, 48)
        self.input_active = [False, False, False, False] # Liste pour savoir quel champ de texte est actif
        self.input_index = 0 # Index du champ actif
        self.noms = noms_escargots # Copie des noms des escargots

    def dessiner(self): # Affiche le fond et le titre
        screen.blit(bg, (0, 0))
        titre = self.font_titre.render("Course d'escargots", True, (255, 255, 255)) # Titre en Blanc
        screen.blit(titre, (L//2 - titre.get_width()//2, 50))

        for i in range(4): # Affiche les 4 escargots et leurs noms
            if i == 0:
                couleur = (0, 0, 255) # Bleu
            elif i == 1:
                couleur = (0, 255, 0) # Vert
            elif i == 2:
                couleur = (255, 255, 0) # Jaune
            else:
                couleur = (128, 0, 128) # Violet

            # Affiche le nom et la touche associée
            nom = self.font_texte.render(f"{self.noms[i]} : ", True, couleur)
            touche = self.font_texte.render(touches_escargots[i], True, (255, 255, 255)) # Touche en Blanc
            screen.blit(nom, (L//2 - 200, 150 + i*70))
            screen.blit(touche, (L//2 + 50, 150 + i*70))

            # Dessine un rectangle autour du champ actif
            if self.input_active[i]:
                pygame.draw.rect(screen, (255, 255, 255), (L//2 - 200, 150 + i*70, 100, 40), 2) # Rectangle en Blanc

        # Dessine le bouton Play
        pygame.draw.rect(screen, (200, 200, 200), (L//2 - 100, 500, 200, 60)) # En gris
        play = self.font_bouton.render("Play", True, (0, 0, 0)) # En noir
        screen.blit(play, (L//2 - play.get_width()//2, 510))

    def gerer_clics(self, pos): # Si on clique sur le bouton Play
        if (L//2 - 100 <= pos[0] <= L//2 + 100) and (500 <= pos[1] <= 560):
            return "play"

        # Si on clique sur un champ de nom
        for i in range(4):
            if (L//2 - 200 <= pos[0] <= L//2 + 100) and (150 + i*70 <= pos[1] <= 190 + i*70):
                # Désactive tous les champs, puis active celui cliqué
                self.input_active = [False]*4
                self.input_active[i] = True
                self.input_index = i
                return "menu"
        return "menu"

    def gerer_saisie(self, event):
        if event.type == pygame.KEYDOWN and any(self.input_active):# Si une touche est pressée et qu'un champ est actif
            i = self.input_index
            if event.key == pygame.K_RETURN: # Désactive le champ si on appuie sur Entrée
                self.input_active[i] = False
            elif event.key == pygame.K_BACKSPACE: # Supprime le dernier caractère si on appuie sur effacer (<-)
                self.noms[i] = self.noms[i][:i]
            else: # Ajoute le caractère tapé au nom
                self.noms[i] += event.unicode
```

```
# Boucle principale
run = True
compteur_pluie = 0
compteur_frame = 0
leaderboard = None
pluie_active = False
confettis_actifs = False
prev_keys = pygame.key.get_pressed()
victoire = None
course_lancee = False
decompte = ['3', '2', '1', 'Go !']
index_decompte = 0
decompte_actif = True
font_decompte = pygame.font.SysFont(None, 100)
afficher_podium = False
podium = Podium(podium)
etat = "menu" # Peut être "menu" ou "jeu"
menu = Menu() # Instancie le menu

while run:
    for evenement in pygame.event.get():
        if evenement.type == pygame.QUIT:
            run = False
        if etat == "menu":
            if evenement.type == pygame.MOUSEBUTTONDOWN:
                action = menu.gerer_clics(evenement.pos)
                if action == "play":
                    etat = "jeu"
                    decompte_actif = True
                    index_decompte = 0
                elif evenement.type == pygame.KEYDOWN:
                    menu.gerer_saisie(evenement)
        clock.tick(60)

        if etat == "menu":
            menu.dessiner()
        else:
            screen.blit(bg, (0, 0))
```

# MARIE L

```
class Goutte:  
    def __init__(self, x, y, vitesse, longueur):  
        self.x = x  
        self.y = y  
        self.vitesse = vitesse  
        self.longueur = longueur  
  
    def dessine(self):  
        pygame.draw.line(screen, (135, 206, 235), (self.x, self.y), (self.x, self.y + self.longueur), 2)  
  
    def tombe(self):  
        self.y += self.vitesse  
        if self.y > H:  
            self.y = randint(-20, 0)  
            self.x = randint(0, L)  
  
# Initialisation des gouttes de pluie  
gouttes = []  
for _ in range(100):  
    x = randint(0, L)  
    y = randint(-H, 0)  
    vitesse = randint(15, 30)  
    longueur = randint(5, 15)  
    gouttes.append(Goutte(x, y, vitesse, longueur))  
  
confettis = [Confetti() for i in range(150)]
```

```
bg = pygame.transform.scale(pygame.image.load(os.path.join("images", "bg.png")), (1280, 720))  
podium = pygame.transform.scale(pygame.image.load(os.path.join("images", "podium.png")), (300, 300))  
bleu = pygame.transform.scale(pygame.image.load(os.path.join("images", "snail_1.1.png")), (110, 110))  
vert = pygame.transform.scale(pygame.image.load(os.path.join("images", "snail_2.1.png")), (110, 110))  
jaune = pygame.transform.scale(pygame.image.load(os.path.join("images", "snail_3.1.png")), (120, 120))  
violet = pygame.transform.scale(pygame.image.load(os.path.join("images", "snail_4.1.png")), (110, 110))
```

```
# Incrémente le compteur de pluie  
compteur_pluie += 1  
if course_lancee:  
    compteur_frame += 1  
  
# Active la pluie  
if compteur_pluie >= 500 and not pluie_active:  
    pluie_active = True  
  
# Si la pluie est active, incrémente le compteur de durée  
if pluie_active:  
    compteur_pluie += 1  
    if compteur_pluie >= 1000 and pluie_active:  
        pluie_active = False  
        compteur_pluie = 0  
  
if victoire:  
    victoire.afficher()  
  
if leaderboard:  
    leaderboard.afficher()  
  
pygame.display.flip()  
  
pygame.quit()  
sys.exit()
```

# MÉLINA

```
class Podium():
    def __init__(self, img):
        self.img = img
        self.positions = {0: (700, 260), 1: (630, 300), 2: (760, 300), 3:(850, 350)}
        self.font = pygame.font.SysFont(None, 40)
        self.texte = self.font.render("Appuyez sur tab pour rejouer !", True, (255, 255, 255))

    def afficher(self, leaderboard, listeescargots):
        screen.blit(self.img, (600, 230))
        screen.blit(self.texte, (L//2 - self.texte.get_width()//2, 470))
        for i in range(min(4, len(leaderboard.resultats))):
            index, temps = leaderboard.resultats[i]
            escargot = listeescargots[index]
            pos = self.positions[i]
            screen.blit(escargot.img, pos)
```

```
class escargot:
    def __init__(self, img, y=680):
        self.x = 0
        self.y = y
        self.vitesse = 10
        self.img = img
        self.position_podium = None

    def bouger(self):
        self.vitesse += 0.1
        self.x += self.vitesse

    def afficher(self):
        screen.blit(self.img, (self.x, self.y))
```

```
# Listes des escargots ainsi que leurs positions
listeescargots = [
    escargot(img=bleu, y=460),
    escargot(img=vert, y=520),
    escargot(img=jaune, y=570),
    escargot(img=violet, y=620)
]
```

**MERCI !**

