REPUBLIQUE ET CANTON DE GENÈVE
Département de l'instruction publique, de la formation et de la jeunesse
Ecole supérieure d'informatique de gestion de Genève (ESIG GE)
Enseignement tertiaire B

ÉPREUVE REGROUPÉE
Janvier 2026

Branche	:	ALGORITHMES ET LANGAGE DE PROGRAMMATION (ALP)
Groupe	:	ESIG 1, groupe 1, 2 et 3
Date	:	14 janvier 2026
Durée	:	180 minutes

Consignes générales

1. L'épreuve est **individuelle**.
2. Seule votre **documentation personnelle** est autorisée. Si elle est électronique, elle doit avoir été copiée sur un emplacement local de votre machine avant le début de l'épreuve.
3. Vous devez utiliser l'IDE **Thonny** pour développer votre solution. Aucun autre environnement de développement (IDE) n'est autorisé.
4. **Toute communication externe est strictement interdite**. Cela inclut :
    a. Toute tentative de contact avec un camarade
    b. Le partage de ressources via le réseau
    c. L'utilisation de l'intelligence artificiel (IA) sous n'importe quelle forme
    d. La présence de dispositifs connectés à proximité immédiate (téléphone portable, montre connectée, lunettes intelligentes, clé USB ou tout autre support de stockage ou de communication). *Vous devez déposer ces objets à l'endroit indiqué par le surveillant.*

⚠ Tout manquement à ces règles sera considéré comme une fraude et sanctionné par la note minimale.

Préparation et installation

Avant le début de l'épreuve

1. **Connectez-vous au réseau** sur le poste de travail qui vous a été attribué.
2. **Copiez votre documentation personnelle** (clé USB, Google Drive, etc.) vers un emplacement local, par exemple dans un répertoire à votre nom dans `C:\ESIGUsers`.
3. **Copiez l'archive de l'épreuve**
    a. Ouvrez le répertoire `Lanschool Files` situé sur le `Bureau`.
    b. Déplacer l'archive `ALP-janvier-26.7z` à la racine de `C:\ESIGUsers`.
    Cette archive est protégée par un mot de passe qui vous sera communiqué au démarrage de l'épreuve.

Une fois le mot de passe reçu

1. **Décompressez l'archive** pour obtenir le répertoire suivant : `C:\ESIGUsers\ALP-janvier-26`
2. **Commencez votre travail** en complétant le script fourni afin de répondre aux spécifications décrites dans cet énoncé.

⚠ Assurez-vous de **travailler sur les fichiers extraits** et non sur l'archive compressée

Soumission du travail

1. **Sauvegardez votre travail** et **fermez votre IDE** (Thonny) avant toute autre action
2. **Renommez** le fichier `Mariokart.py` situé dans `C:\ESIGUsers\ALP-janvier-26` en y ajoutant vos **nom et prénom** à la fin.
3. **Effectuez le rendu de votre travail**
    a. Double-cliquez sur le raccourci "Rendu ALP Jan 26" présent dans `C:\ESIGUsers\ALP-janvier-26`.
    b. Déposez le fichier renommé dans votre répertoire de rendu.

⚠ **Seul le travail déposé dans le répertoire de rendu sera évalué.** Il est de votre responsabilité de vérifier que tous les fichiers nécessaires y sont bien présents.

⚠ **Soyez ponctuel lors de votre reddition** afin d'éviter une éventuelle pénalité.

Mario Kart Cup 2026

1. Contexte

Vous allez analyser les résultats d'un petit championnat du jeu Mario Kart, dans lequel plusieurs pilotes s'affrontent sur différents circuits.

Chaque pilote tente de parcourir ces circuits le plus rapidement possible.

Pour rendre la compétition plus dynamique, ils disposent également d'objets, c'est-à-dire de petites aides permettant d'accélérer, de se protéger ou de perturber un adversaire.

Au fil des circuits, certains pilotes franchissent la ligne d'arrivée, tandis que d'autres abandonnent ou sont disqualifiés à la suite d'un incident.

Votre travail consistera à analyser ce championnat afin d'en extraire plusieurs indicateurs sur les performances des pilotes et sur le déroulement des courses.

2. Données

Votre travail s'effectuera dans le fichier `mariokart.py`.

Voici les données fournies dans le fichier `mariokart_data.py` qui sont importées dans `mariokart.py` :

1. `LST_PILOTES` : Liste des pilotes participant au championnat. Chaque pilote correspond à une ligne de `TAB_TEMPS` et de `TAB_OBJETS`.
2. `DCT_CIRCUIT_VERS_COL` : Dictionnaire associant le nom de chaque circuit à son numéro, c'est-à-dire au numéro de colonne correspondant dans les tableaux 2D.
3. `DCT_CIRCUIT_VERS_LONGUEUR` : Dictionnaire associant le nom de chaque circuit à sa longueur (en mètres).
4. `TAB_TEMPS` : Tableau 2D représentant les résultats des pilotes sur les différents circuits.
   Chaque ligne correspond à un pilote et chaque colonne à un circuit.
   Les valeurs sont codées de la façon suivante :
   - 0 : le pilote n'a pas terminé la course (KO),
   - 1 : le pilote n'a pas terminé la course et a été disqualifié (DQ),
   - > 1 : temps réel du pilote sur le circuit (en secondes).
5. `TAB_OBJETS` : Tableau 2D représentant le nombre d'objets utilisés par les pilotes sur chaque circuit.
   Chaque ligne correspond à un pilote et chaque colonne à un circuit.
   La valeur est égale à 0 si le pilote n'a pas terminé la course ou a été disqualifié.

3. Consignes sur la programmation en Python

a. Interdiction d'utiliser les fonctions prédéfinies comme `max`, `min`, `index`, `count`, `sum`, `mean`, `filter` ou équivalent dans des modules fournis par python.

b. Les éléments suivants sont explicitement autorisés : les fonctions `len` et `append`, les slices, ainsi que le mot-clé `in`, mais uniquement dans le contexte d'une boucle `for … in …` et dans les dictionnaires. En cas de doute, demandez au responsable de l'énoncé.

c. Pas de résultats ou de données codés « en dur » : votre code doit pouvoir s'adapter à d'autres jeux de données.

d. Les seules variables globales autorisées sont les constantes.

e. L'affichage produit par vos programmes doit correspondre exactement aux affichages attendus.

4. Questions

4.1. Courses terminées et objets utilisés (30 minutes)

a. Écrivez une fonction `nb_courses_terminees` qui prend en paramètre l'indice d'un pilote et retourne le nombre de courses qu'il a terminées.

b. Écrivez une fonction `total_objets` qui prend en paramètre l'indice d'un pilote et retourne le total d'objets qu'il a utilisés.

c. Complétez la procédure `question1` afin d'afficher, pour chaque pilote, le nombre de courses terminées ainsi que le total d'objets utilisés. L'affichage doit utiliser les deux fonctions précédentes.

```
=== Question 1 ===
- Léo : 4 courses terminées - 22 objets utilisés
- Mia : 3 courses terminées - 12 objets utilisés
- Noah : 4 courses terminées - 17 objets utilisés
- Sara : 4 courses terminées - 17 objets utilisés
- Yanis : 4 courses terminées - 21 objets utilisés
- Emma : 0 courses terminées - 0 objets utilisés
```

4.2. Récapitulatif par circuit (30 minutes)

Complétez la procédure `question2` afin d'afficher, pour chaque circuit, le résultat obtenu par chacun des pilotes.

Le résultat doit indiquer soit le temps en secondes, soit l'un des codes suivants :

- KO : le pilote n'a pas terminé la course
- DQ : le pilote a été disqualifié

```
=== Question 2 ===
Circuit n°0
  - Léo : 95s
  - Mia : KO
  - Noah : 100s
  - Sara : 98s
  - Yanis : 96s
  - Emma : KO

Circuit n°1
  - Léo : 102s
  - Mia : 110s
  - Noah : KO
  - Sara : 105s
  - Yanis : 103s
  - Emma : DQ
```

```
Circuit n°2
  - Léo : KO
  - Mia : 118s
  - Noah : 112s
  - Sara : 111s
  - Yanis : 109s
  - Emma : KO

Circuit n°3
  - Léo : 130s
  - Mia : DQ
  - Noah : 120s
  - Sara : KO
  - Yanis : 115s
  - Emma : KO

Circuit n°4
  - Léo : 140s
  - Mia : 135s
  - Noah : 128s
  - Sara : 130s
  - Yanis : DQ
  - Emma : DQ
```

4.3. Première course terminée (40 minutes)

a. Écrivez une fonction `premiere_course_terminee` qui reçoit l'indice d'un pilote et retourne le numéro de la première colonne correspondant à une course qu'il a terminée.
Si le pilote n'a terminé aucune course, la fonction doit retourner -1.

b. Écrivez une fonction `dico_premiere_course_terminee` qui retourne un dictionnaire associant chaque pilote au nom de sa première course terminée, ou à "Aucune" si aucune course n'a été terminée.
Cette fonction doit utiliser `premiere_course_terminee`.

c. Complétez `question3` afin d'appeler `dico_premiere_course_terminee` et d'afficher le dictionnaire retourné.

```
=== Question 3 ===
{'Léo': 'Circuit Mario', 'Mia': 'Plaines Yoshi', 'Noah': 'Circuit Mario',
'Sara': 'Circuit Mario', 'Yanis': 'Circuit Mario', 'Emma': 'Aucune'}
```

4.4. Circuit le plus consommateur d'objets (60 minutes)

a. Écrivez une fonction `circuit_plus_gros_consommateur` qui détermine quel circuit est celui où les pilotes utilisent en moyenne le plus d'objets.

   Pour chaque circuit, seuls les pilotes ayant effectivement terminé la course doivent être pris en compte dans le calcul.

   La fonction doit retourner un dictionnaire contenant deux paires clé-valeur :

   1. « circuit », dont la valeur est le nom du circuit identifié ;
   2. « nb_objets_moyen », dont la valeur correspond à la moyenne d'objets utilisés sur ce circuit (arrondie au dixième).

b. Complétez `question4` afin d'appeler cette fonction et afficher le résultat.

```
=== Question 4 ===
Château de Bowser (moyenne : 5.7 objets)
```