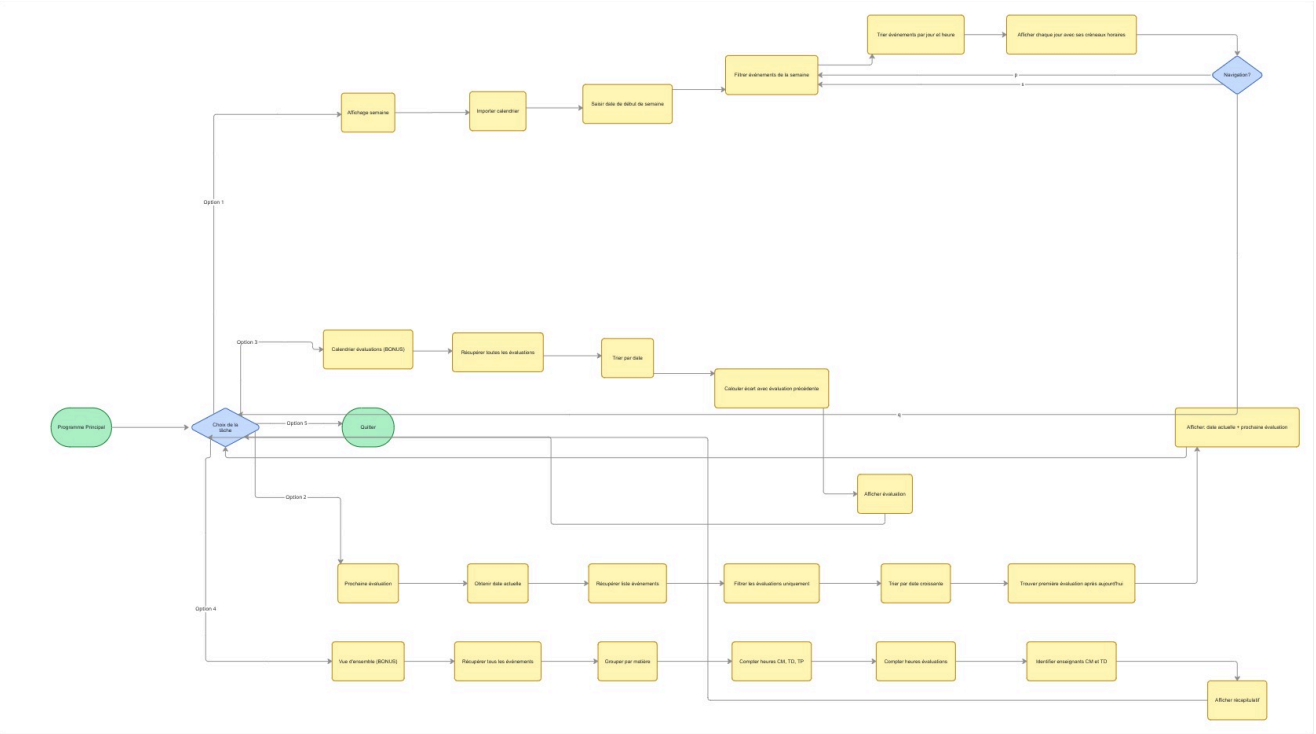
Groupe 1

1. Introduction

Ce document présente le découpage fonctionnel et la structure logicielle du projet "Calendrier Intelligent". L'objectif est de fournir un outil permettant la gestion hebdomadaire d'un emploi du temps et le suivi des évaluations.



2. Liste des fonctions et Docstrings

Voici l'ensemble des fonctions prévues pour la mise en œuvre du projet.

Python

```
# TD17 - CALENDRIER INTELLIGENT

# --- GESTION DES DONNÉES & UTILS ---


def ma_fonction_2(liste, colonne):

    """

    Trie le calendrier par ordre chronologique (année, mois, jour, puis
heure),

    car le fichier CSV n'est pas ordonné.
```

```
    Paramètres :
        - liste (list) : Liste d'événements brute extraite du fichier csv.
        - colonne (int) : Indice de la position de l'année dans chaque sous-liste.

    Return :
         - (list) : La liste triée prête pour un affichage séquentiel.
    """
    pass

def quelle_semaine(jour):
    """
    Calcule la date du lundi correspondant à la semaine du jour saisi par l'utilisateur.
    Parameters :
         - jour (list) : Date au format [année, mois, jour, heure, minute].
    Return :
         - (list) : Date du lundi à 00:00 [année, mois, jour, 0, 0].
    """
    pass

# --- TÂCHE 1 : AFFICHAGE DE LA SEMAINE ---

def affichage_calendrier(calendrier, date_pivot):
```

```
    """
    Affiche les événements de la semaine correspondant à la date choisie.
    Utilise le formatage f"{valeur:02d}" pour l'affichage (ex: 08h05).

    Paramètres :
        - calendrier (list) : Liste d'événements triée.
        - date_pivot (list) : Date de référence pour déterminer la
semaine à afficher.
    Sortie :
        - None (Affiche le planning du lundi au vendredi).
    """
    pass
def naviguer_semaine(commande, date_pivot):
    """
    Permet de reculer ou d'avancer d'une semaine (touches 'p' ou 's').
    Parameters :
        - commande (str) : 's' pour suivante, 'p' pour précédente.
        - date_pivot (list) : La date actuelle affichée.
    Return :
        - (list) : La nouvelle date décalée de 7 jours.
    """
    pass

# --- TÂCHE 2 : RAPPEL PROCHAINE ÉVALUATION ---
```

```
def trouver_IE(calendrier, date_actuelle):
    """
    Recherche la prochaine évaluation (type 'Evaluation') après
aujourd'hui.
    Parameters :
        - calendrier (list) : Liste d'événements triée.
        - date_actuelle (list) : Date système [année, mois, jour, heure,
minute].
    Return :
        - (list) : L'événement de l'évaluation la plus proche, ou None.
    """
    pass


def affichage_pense_bete(calendrier):
    """
    Affiche la date actuelle et les détails de la prochaine évaluation.

    Paramètres :
        - calendrier (list) : Liste complète des événements.
    """
    pass


# --- BONUS : VACANCES & STATISTIQUES ---


def affichage_liste_evaluations(calendrier):
    """
```

```
    Affiche toutes les évaluations du semestre et détecte les vacances
(écart > 7 jours).

    Paramètres :
        - calendrier (list) : Liste de tous les événements triée.
    """
    pass

def calcul_statistiques_matiere(calendrier):
    """
    Calcule le volume horaire par type (CM, TD, TP, Éval) et par matière.

    Paramètres :
        - calendrier (list) : Liste complète des événements.
    """
    pass

# --- CONTRÔLE INTERFACE ---

def retour_debut(retour):
    """ Permet de revenir au menu principal si retour est True. """
    pass

def quitter(quit=False):
    """ Arrête le programme si quit est True. """
    pass
```

```
# ALGORITHME PRINCIPAL (MAIN PROGRAM)


# 1. Initialisation

# import calendrier as cal

# data = cal.creer_calendrier("calendrier.csv")

# calendrier_trie = ma_fonction_2(data, 0)

# date_pivot = aujourdhui()



# 2. Affichage initial (Tâche 2)

# affichage_pense_bete(calendrier_trie)



# 3. Boucle de navigation (Tâche 1)

# tant que non quitter():
#     lundi = quelle_semaine(date_pivot)
#     affichage_calendrier(calendrier_trie, lundi)
#
#     commande = saisir("Appuyez sur 's' (suivante), 'p' (précédente) ou
'q' (quitter) : ")
#     si commande == 'q':
#         quitter(True)
#     sinon:
#         date_pivot = naviguer_semaine(commande, date_pivot)
```

3. Algorithme Principal (Esquisse)

L'exécution suit une boucle interactive :

1. Initialisation des données depuis le CSV.
2. Tri chronologique via `ma_fonction_2`.
3. Affichage du "Pense-bête" (Tâche 2).
4. Boucle de navigation hebdomadaire (Tâche 1) gérant les entrées utilisateur ('s', 'p', 'q').

---

4. Gestion de projet et Versioning (Git)

Dans une démarche d'ingénierie logicielle, notre groupe utilise l'outil **Git** pour la collaboration et le suivi des modifications.

Si vous souhaitez consulter l'historique de nos commits, l'évolution du code ainsi que la répartition du travail entre les membres de l'équipe, vous pouvez accéder à notre dépôt distant via le lien suivant :

https://github.com/Nikita-devel/INSA_Project_G1/tree/main/ISN