Nom :
Prénom :
Groupe :

I2 Devoir sur table

Attention à bien indiquer vos nom, prénom, groupe sur chaque feuille

Aucun document autorisé

1 Questions de cours (5 points)

1) Dans quels cas choisit-on d'écrire un sous-programme sous la forme d'une procédure ? D'une fonction ?

2) Quels sont les avantages et inconvénients d'un tableau trié par rapport à un tableau non trié ?

3) En quoi consiste l'étape de conception globale ?

4) Quelles informations trouve-t-on dans la signature d'un sous-programme ?

5) Soit une cédéthèque contenant des disques, chacun possédant un titre, un artiste (ou groupe) et une année de parution. Proposez un ou plusieurs types de données permettant de gérer ces informations.

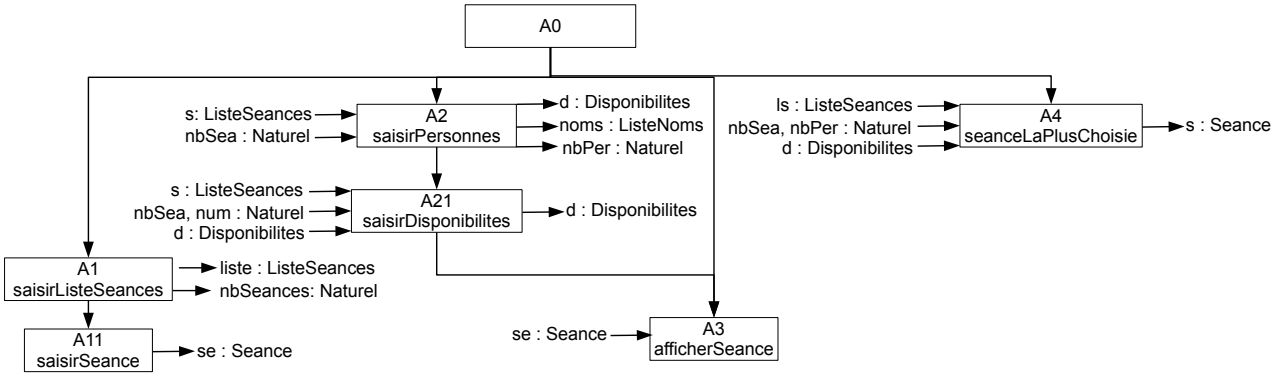
2 Écriture de code (9 points)

Vous devez réaliser la conception détaillée d'un programme permettant de définir plusieurs séances horaires pour recueillir les disponibilités de plusieurs personnes et trouver à quelle séance le plus grand nombre de personnes est disponible (une sorte de *Doodle* local sur une machine). Le programme demande dans un premier temps de saisir les séances possibles en indiquant pour chacune d'elle le jour (numéro de jour, de mois et l'année), une heure de début et une heure de fin. Seules les heures pleines (pas de minutes) seront considérées. Le programme doit s'assurer que le numéro de jour est bien entre 1 et 31, celui du mois entre 1 et 12, les heures entre 1 et 24 et que l'heure de fin n'est pas avant celle de début. Nous ne demandons pas de tester la cohérence entre le numéro de jour et celui du mois.

La deuxième étape consiste à saisir l'identité et la disponibilité de plusieurs personnes. Pour chacune, le programme demande un nom et si elle est disponible pour chacune des séances.

Enfin, il affiche la séance à laquelle le plus grand nombre de personnes est disponible. S'il y en a plusieurs, seule la première est affichée.

Une analyse descendante pour ce programme a été réalisée :



Les types de données suivants sont à définir :

- `Seance` pour représenter une séance horaire possible
- `ListeSeances` pour représenter plusieurs séances
- `ListeNoms` pour représenter les noms de tous les participants
- `Disponibilites` pour représenter les disponibilités de chaque participant à chaque séance

Les sous-programmes correspondent aux signatures suivantes :

procédure saisirListeSeances **(S** liste : ListeSeances ; **S** nbSeances : **Naturel)**

Cette procédure permet de saisir plusieurs séances proposées. Elle donne en sortie la liste des séances saisies et leur nombre.

procédure saisirSeance **(S** se : Seance**)**

Cette procédure remplit le paramètre de sortie `se` avec des informations entrées au clavier, en s'assurant qu'elle sont valides (cf ci-dessus).

procédure saisirPersonnes **(E** s : ListeSeances ; **E** nbSea : **Naturel** ; **S** d : Disponibilites ; **S** noms : ListeNoms ; **S** nbPer : **Naturel)**

Cette procédure permet de saisir les disponibilités des participants au sondage. Les entrées donnent les séances proposées et leur nombre. À l'issue de cette procédure les paramètres de sortie `d`, `noms` et `nbPer` sont remplis avec, respectivement, les disponibilités de chaque participant à chaque séance, le nom de tous les participants et leur nombre.

procédure saisirDisponibilites **(E** s : ListeSeances ; **E** nbSea : **Naturel** ; **E/S** d : Disponibilites ; **E** num : **Entier)**

Cette procédure permet de saisir les disponibilités d'un participant pour chacune des séances. Les entrées donnent les séances proposées et leur nombre, pour qu'on puisse les afficher à l'écran quand on demande si le participant est disponible, et `num` indique le numéro du participant. Le paramètre `d` est modifié en fonction de ce qui est saisi.

procédure afficherSeance **(E** se : Seance**)**

Cette procédure affiche à l'écran les informations liées à la séance donnée en paramètre d'entrée.

fonction seanceLaPlusChoisie **(**ls : ListeSeances ; nbSea : **Naturel** ; d : Disponibilites ; nbPer : **Naturel) :** Seance

Cette fonction trouve la séance qui recueille le plus de disponibilités et la retourne.

Questions

Définir en pseudo code les types `Seance`, `ListeNoms`, `ListeSeances` et `Disponibilites` (les constantes limitant le nombre maximum de séances et de personnes sont données).

Constante MAX_SEANCES = 10
Constante MAX_PARTICIPANTS = 20

Écrire en pseudo code la conception détaillée de tous les sous-programmes.

procédure saisirListeSeances **(S** liste : ListeSeances ; **S** nbSeances : **Naturel)**
`Déclaration :`

`Début`

`Fin`

procédure saisirSeance **(S** se : Seance**)**
`Déclaration :`

Début

Fin

procédure saisirPersonnes **(E** s : ListeSeances ; **E** nbSea : **Naturel** ; **S** d : Disponibilites ; **S** noms : ListeNoms ; **S** nbPer : **Naturel)**

Déclaration :

Début

Fin

procédure saisirDisponibilites (**E** s : ListeSeances ; **E** nbSea : **Naturel** ; **E/S** d : Disponibilites ; **E** num : **Entier**)

`Déclaration :`

`Début`

`Fin`

procédure afficherSeance (**E** se : Seance)

`Déclaration :`

`Début`

`Fin`

fonction seanceLaPlusChoisie (ls : ListeSeances ; nbSea : **Naturel** ; d : Disponibilites ; nbPer : **Naturel**) **:** Seance

`Déclaration :`

`Début`

```
Fin
```

Programme principal

```
Déclaration :
```

```
Début
```

```
Fin
```

Vous pouvez utiliser l'espace restant pour compléter les pseudo-codes précédents si nécessaire (mais normalement la place fournie est largement suffisante pour écrire un programme efficace).

3 Analyse descendante (6 points)

Dans cet exercice, vous ne devez réaliser que la conception globale, c'est-à-dire écrire les **types de données** pertinents, une **analyse descendante** et **les signatures de fonctions et procédures**.

Le programme à concevoir est un programme de gestion de bibliothèque. Une **bibliothèque** est un ensemble de *livres*. Chaque **livre** est caractérisé par un code *identifiant* numérique, un *titre*, un *auteur*, le dernier *emprunteur* connu et son *statut* (emprunté ou non). On supposera qu'une bibliothèque contient au plus 1000 livres ; on doit pouvoir connaître à tout moment le nombre de livres au catalogue (i.e. le nombre de livres total empruntés ou non). Nous considérons dans cet exercice que nous ne gérons qu'une seule bibliothèque, et que celle-ci est sauvegardée dans un fichier.

Au lancement du programme, la bibliothèque en cours, stockée dans un fichier, est chargée. L'utilisateur peut alors choisir dans un menu :

- d'afficher tous les livres disponibles (non-empruntés) ;
- d'afficher tous les livres d'un même auteur ;
- d'afficher tous les livres empruntés par une même personne ;
- d'afficher un livre par son identifiant ;
- d'emprunter un livre par son identifiant ;
- de rendre un livre par son identifiant.

À la fin de l'exécution, la bibliothèque modifiée est sauvegardée dans le même fichier qu'initialement.

1) Constantes et Types

Définir par une constante le nombre de livres maximal que peut contenir une bibliothèque et deux types de données pour représenter un **livre** et une **bibliothèque**.

2) Analyse descendante

Dessiner une analyse descendante possible pour ce programme.

3) Donner les signatures des sous-programmes correspondant à cette analyse.