TD10 : éléments de correction

Exercice 1

```
Structure Lotterie
{
        Champ nom : chaîne de caractères
        Champ nMaxSerie1 : entier
        Champ nSerie1 : entier
        Champ nMaxSerie2 : entier
        Champ nSerie2 : entier
        Champ probaRang1 : réel
}
```

```
Fonction : Factorielle
Précondition : n ≥ 0
Entrée :        - n : entier
Local :         - i : entier
Sortie :        - fact : entier
Début
|       Soit fact = 1
|       Pour i de 2 à n avec un pas de +1 Faire
|       |       Soit fact = fact * i
|       Fin Pour
|       Retourner fact
Fin
```

```
Fonction : PParmiN
Précondition : p ≥ 0, n ≥ 0
Entrée :        - p : entier
                - n : entier
Local :         /
Sortie :        - proba : entier
Début
|       Soit proba = Factorielle(n) / (Factorielle(p) * Factorielle(n-p))
|       Retourner proba
Fin
```

```
Fonction : ProbaRang1Loto
Entrée :        void
Local :         /
Sortie :        - proba : entier
Début
|       Soit proba = PParmiN(5, 49) * PParmiN(1, 10)
|       Retourner proba
Fin
```

```
Fonction : CreerListeJeux
Entrée :        - n : entier (modifiable)
Local :         - i : entier
                - nomJeu : chaîne de caractères
Sortie :        - listeJeux : tableau 1D de chaînes de caractères
Début
|       Soit n = -1
|       Tant Que n ≤ 0 Faire
|       |       Soit n = SaisirEntier()
|       Fin Tant Que
|       Soit listeJeux = tableau 1D de n chaînes de caractères
|       Soit nomJeu = ""
|       Pour i de 1 à n avec un pas de +1 Faire
|       |       Soit nomJeu = SaisirChaine()
|       |       listeJeux[i] = nomJeu
|       Fin Pour
|       Retourner listeJeux
Fin
```

```
Procédure : InitTabLotteries
Précondition : tailleTabLotteries == tailleListeJeux
Entrée :        - tabLotteries : tableau 1D de tailleTabLotteries structure Factorielle (modifiable)
                - tailleTabLotteries : entier
                - listeJeux : tableau 1D de tailleListeJeux chaînes de caractères
                - tailleListejeux : entier
Local :         - i : entier
Sortie :        void
Début
|       Pour i de 0 à tailleTabLotteries avec un pas de +1 Faire
|       |       Soit tabLotteries[i].nom = listeJeux[i]
|       Fin Pour
Fin
```

```
Procédure : CompleterTabLotteries
Entrée :        - tabLotteries : tableau 1D de tailleTabLotteries structure Factorielle (modifiable)
                - tailleTabLotteries : entier
Local :         - i : entier
                - nMaxSerie1 : entier
                - nSerie1 : entier
                - nMaxSerie2 : entier
                - nSerie2 : entier
Sortie :        void
Début
|       Pour i de 0 à tailleTabLotteries avec un pas de +1 Faire
|       |       Soit nMaxSerie1 = SaisirEntier()
|       |       Soit nSerie1 = SaisirEntier()
|       |       Soit nMaxSerie2 = SaisirEntier()
|       |       Soit nSerie2 = SaisirEntier()
|       |       Soit tabLotteries[i].nMaxSerie1 = nMaxSerie1
|       |       Soit tabLotteries[i].nSerie1 = nSerie1
|       |       Soit tabLotteries[i].nMaxSerie2 = nMaxSerie2
|       |       Soit tabLotteries[i].nSerie2 = nSerie2
|       Fin Pour
Fin
```

```
Procédure : AjoutProbaTabLotteries
Entrée :        - tabLotteries : tableau 1D de tailleTabLotteries structure Factorielle (modifiable)
                - tailleTabLotteries : entier
Local :         - i : entier
Sortie :        void
Début
|       Pour i de 0 à tailleTabLotteries avec un pas de +1 Faire
|       |       Soit tabLotteries[i].probaRang1 =
|       |               PParmiN(tabLotteries[i].nSerie1, tabLotteries[i].nMaxSerie1) *
|       |               PParmiN(tabLotteries[i].nSerie2, tabLotteries[i].nMaxSerie2)
|       Fin Pour
Fin
```

```
Procédure : SauverStructLotterie
Entrée :        - lotterie : structure Lotterie
                - chemin : chaîne de caractères
Local :         - f : flux d'écriture
Sortie :        void
Début
|       Soit f = flux d'écriture initialisé sur chemin
|       Écrire(f, "{lotterie.nom},")
|       Écrire(f, "{Chaîne(lotterie.nMaxSerie1)},")
|       Écrire(f, "{Chaîne(lotterie.nSerie1)},")
|       Écrire(f, "{Chaîne(lotterie.nMaxSerie2)},")
|       Écrire(f, "{Chaîne(lotterie.nSerie2)},")
|       Écrire(f, "{Chaîne(lotterie.probaRang1)}\n")
|       FermerFlux(f)
Fin
```

```
Procédure : ChargerStructLotterie
Entrée :        - chemin : chaîne de caractères
Local :         - f : flux de lecture
Sortie :        - lotterie : structure Lotterie
Début
|       Soit f = flux de lecture initialisé sur chemin
|       Soit ligne = LireLigne(f)
|       Soit details = Séparer(ligne, ',')
|       Soit lotterie = structure Lotterie
|       Soit lotterie.nom = details[0]
|       Soit lotterie.nMaxSerie1 = Entier(details[1])
|       Soit lotterie.nSerie1 = Entier(details[2])
|       Soit lotterie.nMaxSerie2 = Entier(details[3])
|       Soit lotterie.nSerie2 =  Entier(details[4])
|       Soit lotterie.probaRang1 = Entier(details[5])
|       FermerFlux(f)
|       Retourner lotterie
Fin
```

```
Procédure : SauverTableauStructLotterie
Entrée :        - lotteries : tableau 1D de tailleTab structure Lotterie
                - chemin : chaîne de caractères
Local :         - f : flux d'écriture
Sortie :        void
Début
|       Soit f = flux d'écriture initialisé sur chemin
|       Écrire(f, "{tailelTab}\n")
|       Pour i de 0 à tailleTab − 1 avec un pas de +1 Faire
|       |       Écrire(f, "{lotteries[i].nom},")
|       |       Écrire(f, "{Chaîne(lotteries[i].nMaxSerie1)},")
|       |       Écrire(f, "{Chaîne(lotteries[i].nSerie1)},")
|       |       Écrire(f, "{Chaîne(lotteries[i].nMaxSerie2)},")
|       |       Écrire(f, "{Chaîne(lotteries[i].nSerie2)},")
|       |       Écrire(f, "{Chaîne(lotteries[i].probaRang1)}\n")
|       Fin Pour
|       FermerFlux(f)
Fin
```

```
Procédure : ChargerTabStructLotterie
Entrée :        - chemin : chaîne de caractères
Local :         - f : flux de lecture
Sortie :        - lotteries : tableau 1D de structure Lotterie
Début
|       Soit f = flux de lecture initialisé sur chemin
|       Soit ligne = LireLigne(f)
|       Soit taille = Entier(ligne)
|       Soit lotteries = tableau 1D de taille structure Lotterie
|       Pour i de 0 à taille − 1 avec un pas de +1 Faire
|       |       Soit ligne = LireLigne(f)
|       |       Soit details = Séparer(ligne, ',')
|       |       Soit lotteries[i].nom = details[0]
|       |       Soit lotteries[i].nMaxSerie1 = Entier(details[1])
|       |       Soit lotteries[i].nSerie1 = Entier(details[2])
|       |       Soit lotteries[i].nMaxSerie2 = Entier(details[3])
|       |       Soit lotteries[i].nSerie2 =  Entier(details[4])
|       |       Soit lotteries[i].probaRang1 = Entier(details[5])
|       Fin Pour
|       FermerFlux(f)
|       Retourner lotteries
Fin
```

```
Procédure : Main
Entrée :        void
Local :         - tabLotteries : tableau 1D de structure Lotterie
                - tailleTabLotteries : entier constant = 10
                - listeJeux : tableau 1D de chaînes de caractères
                - tailleListeJeux : entier
Sortie :        void
Début
|       Soit tabLotteries = tableau 1D de tailleTabLotteries structure Lotterie
|       Soit tailleListeJeux = 0
|       Soit listeJeux = CreerListeJeux(tailleListeJeux) // mise à jour de tailleListeJeux
|       Si tailleTabLotteries == tailleListeJeux Alors
|       |       InitTabLotteries(tabLotteries, tailleTabLotteries, listeJeux, tailleListeJeux)
|       |       CompleterTabLotteries(tabLotteries, tailleTabLotteries)
|       |       AjoutProbaTabLotteries(tabLotteries, tailleTabLotteries)
|       |       // trier le tableau par ordre décroissant du champ probaRang1
|       |       // en C# : Array.Sort(tabLotteries, (x, y) => x.probaRang1.CompareTo(y.probaRang1))
|       |       TrierTableau(tabLotteries, ordre=décroissant, critère=probaRang1)
|       |       Pour i de 0 à tailleTabLotteries - 1 avec un pas de +1 Faire
|       |       |       Afficher("tabLotteries[i].nom avec un proba de {1/tabLotteries[i].probaRang1}")
|       |       Fin Pour
|       Fin Si
|       // sauvegarder le tableau dans un fichier puis le relire
|       SauverTableauStructLotterie(tabLotteries, "lotteries.txt")
|       Soit testTab = ChargerTableauStructLotterie("lotteries.txt")
Fin
```