TP 4 : Fonctions – Strucures itératives – Structures conditionnelles

1. **Ecriture d’une fonction**

Soit f la fonction définie par :

où, et sont des paramètres.

**Exercice 1 :**

Ecrire les commandes permettant de créer cette fonction.

**Exercice 2 :**

Comment faire si on veut définir les paramètres et par défaut à 2, 1 et 1 respectivement ?

**Exercice 2 :**

Soit x=c(1,10). Que retourne cette fonction si on l’appelle avec les paramètres par défaut ?

1. **Boucles for, while**

**Exercice 1 :**

Parcourir les entiers de 1 à 20 à l’aide d’une boucle for en affichant dans la console à chaque itération si le nombre courant est pair.

**Exercice 2 :**

À l’aide de la fonction while(), créer une boucle qui permet de calculer la factorielle d’un nombre.

**Exercice 3 :**

Réutiliser le code de la question précédente pour en faire une fonction qui, lorsqu’on lui donne un nombre, retourne sa factorielle. Comparer le résultat avec la fonction factorial().

**Exercice 4 :**

Choisir un nombre mystère entier entre 1 et 100, et le stocker dans un objet que l’on nommera nombre\_mystere. Ensuite, créer une boucle qui à chaque itération effectue un tirage aléatoire d’un entier compris entre 1 et 100. Tant que le nombre tiré est différent du nombre mystère, la boucle doit continuer.

À la sortie de la boucle, une variable que l’on appellera nb\_tirages contiendra le nombre de tirages réalisés pour obtenir le nombre mystère.

**Exercice 5 :**

Utiliser le code de la question précédente pour réaliser la fonction trouver\_nombre, qui, lorsqu’on lui donne un nombre compris entre 1 et 100, retourne le nombre de tirages aléatoires d’entiers compris entre 1 et 100 nécessaires avant de tirer le nombre mystère.

**Exercice 6 :**

En utilisant une boucle for, faire appel 1000 fois à la fonction trouver\_nombre() qui vient d’être créée. À chaque itération, stocker le résultat dans un élément d’un vecteur que l’on appellera nb\_essais\_rep. Enfin, afficher la moyenne du nombre de tirages nécessaires pour retrouver le nombre magique. nb\_essais\_rep <- rep(NA, 1000)

**Exercice 7 :**

Utiliser une boucle for pour reproduire la suite de Fibonacci jusqu’à son cinquantième terme.

1. **Etude de data frames**

**Exercice 8 :**

1. Trouver le jeu de données ECRIN développé par A.B. Dufour, S. Dray, D. Chessel & J.R. Lobry De l’Université de Lyon.
2. Lire les données dans un data frame nommé « ecrin ».
3. Visualiser les données et trouver les informations concernant les variables du data set.
4. Vérifier les dimensions du data set.
5. Calculer les statistiques élémentaires de la variable RIC.
6. Interpréter les graphiques obtenus par les commandes :

*plot(ecrin$RIC)*

*plot(ecrin$SEM,ecrin$STA)*

1. Tracer un histogramme des données RIC. Définir un titre précisant la moyenne et la variance de ces données.
2. La variable *ecrin$SEM* est-elle quantitative ou qualitative ?
3. Créer une nouvelle variable *sem.f* qui sera qualitative. De même ; créer une variable *heu.f .*
4. Calculer le nombre de relevés matinaux et le nombre de relevés vespéraux.
5. Calculer le nombre de relevés par semaine et par moment de la journée.
6. Quelle est la moyenne du nombre d’espèces observées par semaine ? Représenter les résultats graphiquement.
7. Mêmes questions avec la moyenne du nombre d’espèces observées par semaine et par moment de la journée. Représenter par des symboles graphiques les différents les courbes du matin et du soir, et afficher la légende correspondante Mettre un titre au graphique et nommer les variables de chaque axe.
8. Créer une disposition graphique permettant de tracer 2 lignes et de 3 plots. Représenter la moyenne du nombre d’espèces observées le soir par semaine en essayant les différents types de représentation graphique (type =….). Mettre un titre à chaque plot et un titre général à la feuille. Exporter le plot sur votre répertoire de travail.
9. Représenter sur un même graphique les boîtes à moustaches du nombre d’espèces observées par semaine.