Evaluation Algorithmique

A) QCM

1. En langage algorithmique, un algorithme commence par le mot :
    a. Variable
    b. Constante
    c. Début

2. Dans un algorithme, les variables et les constantes se déclarent :
    a. Dans l'en-tête de l'algorithme
    b. Entre "Début" et "Fin"
    c. Les variables sont dans l'en-tête et les constantes, entre "Début" et "Fin"

3. Dans un algorithme, une variable du type entier se nommant "i" est déclarée par l'instruction:
    a. Variable i <- entier
    b. Variable i en entier
    c. Variable entière : i

4. Dans un algorithme, une constante se nommant "k" et valant 97 est déclarée par l'instruction:
    a. Constante entière : k <- 97
    b. Constante k = 97
    c. Constante k <- 97

5. Dans la portion d'algorithme suivant, A, B, C et D sont des entiers. Si A vaut 1 avant l'exécution de la première ligne, combien vaut D après l'exécution de ces 3 lignes ?
   B <- 2 * A - 4
   C <- -3 * A – 2 * B
   D <- 15 * C + 21 * B / 3
    a. D vaut -1
    b. D vaut 0
    c. D vaut 1

6. Soit i une variable du type entier, contenant la valeur 9. Qu'affiche la portion d'algorithme suivant :
   Si i < 10 Alors
   Ecrire("C'est bon !")
   Sinon
   Ecrire("C'est mauvais !")
   Fin Si
    a. Cette portion d'algorithme affiche "C'est bon !"
    b. Cette portion d'algorithme affiche "C'est mauvais !"
    c. Cette portion d'algorithme n'affiche rien

7. Dans la portion d'algorithme suivant, k est un entier ayant pour valeur 7. Que vaut k après l'exécution de cette portion d'algorithme :

```
Si k modulo 2 = 1 Alors
  k <- k – 1
Fin Si
k <- k * 2
```

   a. La valeur de k est 12
   b. La valeur de k est 14
   c. La valeur de k est 16

8. Dans la portion de code suivante, i et j sont des entiers. Que vaut j après l'exécution de ces lignes :

```
j <- 0
Pour i <- 1 à 11 par pas de 3
  j <- j + i
Fin Pour
```

   a. j vaut 4
   b. j vaut 11
   c. j vaut 22

9. Dans la portion de code suivante, n’est un entier. Que vaut n après l'exécution de ces lignes :

```
n <- 0
tant que n > 10 faire
  n <- n + 2
Fin Tant que
```

   a. n vaut 0
   b. n vaut 20
   c. n vaut 22

10. Dans la portion de code suivante, m est un entier ayant pour valeur 7. Que vaut m lorsque l'exécution de ces lignes est terminée :

```
répéter
  si m modulo 2 = 1 alors
    m <- 3 * m + 1
  sinon
    m <- m / 2
  Fin si
jusqu'à m = 1
```

   a. m vaut 0
   b. m vaut 1
   c. m vaut 2

B) Exercices rapides

a. Écrivez un programme qui permute les valeurs de deux variables lues au clavier

b. Écrivez un programme qui effectue une permutation circulaire vers la droite de 4 variables lues au clavier

c. Écrivez un programme qui donne le maximum de 3 nombres lus au clavier

d. Écrivez un programme qui calcule la moyenne de 4 nombres lus au clavier et donne le minimum

e. Écrivez un programme qui calcule le prix TTC d'un prix HT entré au clavier

f. Écrivez un programme qui calcule le pourcentage d'un nombre, ce nombre ainsi que le pourcentage sont entrés au clavier

g. Écrivez un programme qui fournit une valeur en francs à partir d'un prix entré en euros

h. Écrivez un programme qui donne une température en degrés Celsius à partir d'une température Fahrenheit (C = (5/9) * (F - 32))

i. Écrivez un programme qui donne la mention d'un étudiant en fonction de sa note entrée au clavier

j. Écrivez une fonction `somme(x, y en entier) en entier` qui retourne l'addition de x et y.

k. Écrivez une fonction minuscule `minuscule(phrase en chaine) en chaine` qui retourne la chaine en minuscule (sinon elle retourne la chaine `phrase` sans le modifier).

l. Écrivez une fonction `compter_occurrences(phrase en chaine, mot en chaine)` qui retourne le nombre d'occurrences de mot dans chaine.

m. Écrivez un algorithme qui prend en entrée le nom du fichier contenant l'annuaire, ainsi que le nom, prénom, numéro de téléphone et l'e-mail d'une personne et qui sauvegarde le fichier annuaire en classant alphabétiquement la personne.
NB : on suppose que les entrées de l'annuaire existant sont classées alphabétiquement.

n. Ecrivez un algorithme qui affiche le numéro de téléphone du nom d'une personne passé sur la ligne de commande. Si ce nom n'existe pas dans l'annuaire, il affichera un message d'erreur.

C) Plus de réflexion

Vous allez écrire un programme qui permet à un joueur de faire une partie de Black Jack. L'objectif principal de ce jeu de cartes est d'avoir une main qui vaut plus que celle de la banque, sans toutefois dépasser la valeur 21.

Lorsque le jeu commence, l'ordinateur distribue deux cartes au joueur ainsi qu'à lui-même. Une des deux cartes de l'ordinateur est visible pour le joueur. Le joueur peut demander une nouvelle carte ou s'abstenir. Au coup suivant, l'ordinateur peut lui-même décider de prendre une nouvelle carte ou de s'abstenir.
Le jeu cesse lorsque le joueur ou l'ordinateur ont dépassé 21, ou s'ils ont atteint 21 ou encore s'ils ont passé tous les deux leur tour.
Les cartes marquées de 2 à 10 gardent leur valeur, les figures (reines, rois, et valets) valent 10 points, l'as vaut 11 points ou 1 point. L'as prend la valeur 1 lorsque le score est supérieur à 21 et qu'il avait la valeur 11.
Ex : 3 de coeur + As de pique + 10 de carreau = 3 + 11 + 10 = 24 lorsque l'as vaut 11 et prend en fait la valeur 14 puisqu'on a dépassé 21 avec la valeur 11.

Lorsque le jeu commence, 2 cartes sont attribuées à la banque, la première étant cachée, et 2 cartes sont attribuées au joueur. Le joueur peut, une fois les cartes distribuées, soit demander une carte, soit décider de s'arrêter (par ex. si son score est 20, il s'arrêtera). L'ordinateur qui représente la banque, peut lui aussi demander une carte (une fois que le joueur a fait son choix) ou s'arrêter. La banque demandera une carte si son score est inférieur à 17. Si le joueur a décidé ne plus demander de carte et que la banque a un score supérieur au joueur, la banque décidera de s'arrêter. La partie se poursuit jusqu'à ce qu'un des joueurs réalise un black jack (21), ou dépasse 21, ou que les deux joueurs aient décidé de s'arrêter.

Conseil : procédez par étapes

1) Découper bien votre programme en différente fonctions
2) Commencez par distribuer 2 cartes à la banque, et au joueur, affichez les mains et le score.
3) Puis écrivez la partie qui dit si quelqu'un a gagné et qui affiche la main de la banque avec la carte cachée.
4) Lorsque ces deux étapes fonctionnent écrivez une boucle qui tire une carte une fois pour le joueur, une fois pour la banque et s'arrête lorsqu'il y a un black jack ou que le score est dépassé.
5) Ajoutez des instructions pour demander au joueur s'il souhaite tirer une carte ou arrêter, l'ordinateur continue à jouer jusqu'à ce qu'il dépasse le score
6) Ajoutez les instructions pour que l'ordinateur décide s'il tire une carte ou s'il s'arrête.