ALGORITHME AVANCÉE		
TRAVAUX DIRIGERS N°1		
Classe : L3 SI	**Durée** :1 Heures	**Date : 3**0/10/2024
Enseignant : Dr GBAME Gbede Sylvain		**Ecole :UFHB**

Exercice 1 : ***Calcul du produit d'un nombre par une factorielle***

Soit g un nombre réel et n un nombre entier naturel, proposer un algorithme qui calcule g × n!

Exercice 2 **: Activités au choix**

Écrire un algorithme qui, à partir de la saisie d'un mot, détermine, selon le choix de l'utilisateur :

- son nombre de voyelles ;
- ou son anagramme renversée

Exercice 3 **:** ***Calcul de la puissance d'un nombre***

Écrire une procédure appelée Puissance qui prend en paramètres deux entiers, un nombre x et un exposant n, et retourne la valeur de x^n (le nombre xxx élevé à la puissance n). Ensuite, écrire un algorithme principal qui demande à l'utilisateur d'entrer un nombre et un exposant, appelle la procédure Puissance, puis affiche le résultat.

Exercice 4 : **Trouver le maximum dans un tableau**

Écrire une procédure appelée Maximum qui prend en paramètre un tableau d'entiers et retourne la valeur maximale présente dans ce tableau. Ensuite, écrire un algorithme principal qui demande à l'utilisateur d'entrer la taille du tableau et les éléments du tableau, appelle la procédure Maximum, puis affiche le résultat.

Exercice 5 : Calcul de l'aire d'un cercle

Écrire une fonction appelée AireCercle qui prend en paramètre le rayon r d'un cercle et retourne l'aire de ce cercle. L'aire d'un cercle est donnée par la formule A=π×r2A = \pi \times r^2A=π×r2. Ensuite, écrire un algorithme principal qui demande à l'utilisateur de saisir le rayon, appelle la fonction AireCercle, puis affiche le résultat.

Exercice 6 : Vérifier si un nombre est premier

Écrire une fonction appelée EstPremier qui prend un entier nnn en paramètre et retourne Vrai si nnn est un nombre premier, ou Faux sinon. Un nombre premier est un nombre entier supérieur à 1 qui n'a pas d'autres diviseurs que 1 et lui-même. Ensuite, écrire un algorithme principal qui demande à l'utilisateur de saisir un nombre, appelle la fonction EstPremier, puis affiche un message indiquant si le nombre est premier ou non.

EXERCICES SUR LES FONCTIONS ET LES PROCEDURES

Exercice 1

Écrire une fonction **Somme** qui permet de calculer la somme de n entiers saisis.

Exercice 2

Écrire une fonction **Petit** qui permet de retourner le plus petit élément d'un tableau de n entiers.

Exercice 3

Ecrire une fonction qui calcule la factorielle de n d'un nombre.

Exercice 4

Écrire une fonction **Somme1** qui permet de calculer la somme des éléments réels d'une matrice carrée comportant n lignes et n colonnes.

Exercice 5

Écrire une **fonction Puissance**, qui à partir d'un réel x et une valeur entière positive n, retourne x à la puissance n (x^n)

Exercice 6

Écrire une fonction PGCD qui calcule le Plus Grand Commun Diviseur de 2 entiers strictements positifs.

Exercice 7

Ecrire une fonction **fonmath** qui permet de retourner f(x,y) = xy + $5x^2$-2y avec x et y deux réels.

Exercice 8

Écrire une fonction **nbreoccur** qui permet d'afficher le nombre d'occurrence d'un caractère **c** dans une chaîne de caractères **ch**.

Exercice 9

Écrire une procédure **Suite** qui permet de chercher et d'afficher les n premiers termes de la suite V définie par :

$$V_n \begin{cases} 4 & \text{Si } n = 0 \\ 2\,V_{n-1} + 3 & \text{Si } n > 0 \end{cases}$$

Exercice 10

Écrire une procédure **Temps** qui permet de retourner le nombre d'heures, minutes etsecondes pour un temps donné T sous forme d'un nombre de secondes

EXERCICES SUR LES PILES, LES FILS, LES POINTEURS, LES LISTES CHAINES ET LES ENREGISTREMENTS

1. Exercices sur les Piles :

Exercice 1 : **Implémentation de base**

Écrivez une fonction en langage C (ou un autre langage) pour implémenter une pile en utilisant un tableau statique. La pile doit inclure les opérations suivantes :

- push pour ajouter un élément à la pile,
- pop pour retirer un élément de la pile,
- peek pour afficher l'élément au sommet de la pile,
- isEmpty pour vérifier si la pile est vide.

Testez ensuite votre pile avec un programme qui ajoute et retire des éléments.

Exercice 2 : **Vérification de parenthésage**

Écrivez une fonction qui utilise une pile pour vérifier si une expression arithmétique contient un parenthésage correct (exemple : ((a + b) * c) est correcte, mais (a + b * c) est incorrect). La fonction doit retourner vrai si le parenthésage est correct et faux sinon.

2. Exercices sur les Files :

Exercice 1 : Implémentation de base

Créez une file en utilisant un tableau de taille fixe en C. Implémentez les opérations suivantes :

- enqueue pour ajouter un élément en fin de file,
- dequeue pour retirer un élément en début de file,
- isEmpty pour vérifier si la file est vide,
- isFull pour vérifier si la file est pleine.

Testez votre file en ajoutant et retirant des éléments.

Exercice 2 : Simulation de file d'attente

Imaginez une file d'attente dans une banque où chaque client est représenté par un numéro. Écrivez un programme qui utilise une file pour simuler l'arrivée de clients à un guichet. À chaque tour, un client est servi (retiré de la file) et un nouveau client peut entrer. Affichez la file après chaque modification.

3. Exercices sur les Listes Chaînées :

Exercice 1 : Création d'une liste chaînée simple

Implémentez une liste chaînée simple (liste simplement chaînée) en C. La liste doit inclure les opérations suivantes :

- ajouter_en_tete pour ajouter un élément au début de la liste,
- ajouter_en_fin pour ajouter un élément à la fin de la liste,
- supprimer pour supprimer un élément donné,
- afficher pour afficher tous les éléments de la liste.

Testez chaque opération.

Exercice 2 : Inversion d'une liste chaînée

Écrivez une fonction qui prend une liste chaînée comme argument et inverse son ordre. Par exemple, si la liste est [1 -> 2 -> 3], la liste inversée doit devenir [3 -> 2 -> 1].

4. Exercices sur les Pointeurs :

Exercice 1 : Manipulation de pointeurs

Déclarez deux variables entières, puis deux pointeurs pointant vers ces variables. Utilisez les pointeurs pour :

- Échanger les valeurs des deux variables,
- Afficher les valeurs des variables avant et après l'échange.

Exercice 2 : Allocation dynamique

Utilisez l'allocation dynamique de mémoire pour créer un tableau de taille donnée par l'utilisateur. Remplissez le tableau avec des valeurs, affichez-le, puis libérez la mémoire allouée.

5. Exercices sur les Enregistrements (Structures) :

Exercice 1 : Enregistrement de coordonnées

Définissez une structure appelée Point pour représenter un point dans un espace 2D avec des champs x et y. Créez une fonction qui prend deux points en argument et renvoie la distance euclidienne entre eux.

Exercice 2 : Gestion des employés

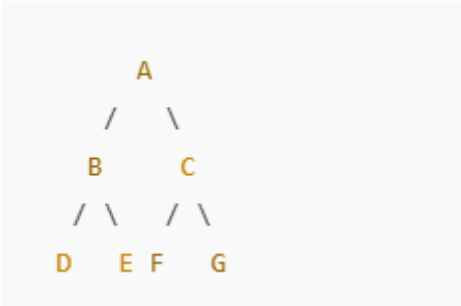
Définissez une structure Employe avec des champs pour le nom, l'âge et le salaire de l'employé. Créez un tableau dynamique d'employés et implémentez un programme qui permet :

- d'ajouter un employé,
- de modifier le salaire d'un employé,
- d'afficher les informations de tous les employés.

6. Exercices sur les Arbres (Structures) :

Exercice 1 : Parcours d'un arbre binaire

On vous donne la représentation suivante d'un arbre binaire :



Questions :

1. Écrivez les résultats des parcours suivants de cet arbre :
    - Parcours en profondeur préfixe (pré-ordre).
    - Parcours en profondeur infixe (in-ordre).
    - Parcours en profondeur postfixe (post-ordre).
    - Parcours en largeur (par niveaux).
2. Implémentez en pseudocode ou dans le langage de programmation de votre choix une fonction pour effectuer un parcours en profondeur préfixe.

Exercice 2 : Insertion et suppression dans un arbre binaire de recherche

On vous demande de manipuler un arbre binaire de recherche. Initialement, l'arbre est vide.

Questions :

1. Insérez successivement les éléments suivants dans l'arbre : 50, 30, 70, 20, 40, 60, 80. Dessinez l'arbre résultant.
2. Supprimez les éléments suivants de l'arbre dans l'ordre donné :
    - Supprimez le nœud avec la valeur 20 (cas d'une feuille).
    - Supprimez le nœud avec la valeur 30 (cas d'un nœud avec un enfant).
    - Supprimez le nœud avec la valeur 50 (cas d'un nœud avec deux enfants).
3. Décrivez et implémentez une méthode pour trouver le successeur immédiat (in-order successor) d'un nœud donné dans un arbre binaire de recherche.