IUT 'A' Paul SABATIER
Dépt INFORMATIQUE

M3101 : Principes des systèmes d'exploitation

TD : Compléments de langage C

TD1: les fonctions C

Objectifs :

Étude des fonctions du langage C et mise en œuvre dans le codage des fonctions et procédures du langage algorithmique.
Support de cours : Compléments de langage C - pages 16 à 20, §4.1 à §4.6.

Exercice 1 :

Soit l'algorithme suivant :

```
-- procédure de recherche de la première occurrence de l'élément élé dans le tableau tab
-- de n éléments ; si élé existe trouvé = VRAI et rang désigne la position de élé dans tab ;
-- sinon trouvé = FAUX et rang est indéfini.

procedure rechercherOccurrence ( tab : in TabEntiers ; n, élé : in Integer ;
                                 trouvé : out Boolean, rang : out Integer ) is

   fini : Boolean ; -- indicateur d'arrêt de la recherche
   i : Integer ; -- indice de parcours du tableau tab

begin
   i := 0 ;                    -- se positionner sur le premier élément du tableau
   fini := false ;
   while not fini loop
      if i < n then            -- tester si l'indice courant est valide
         if élé = tab(i) then
            trouvé := true ;   -- l'élément est trouvé,
            rang := i ;        -- fournir le résultat de la recherche,
            fini := true ;     -- arrêter la recherche
         else
            i := i + 1 ;       -- passer à l'élément suivant
         end if ;
      else
         trouvé := false;      -- l'élément n'est pas  dans le tableau,
         fini := true;         -- arrêter la recherche
      end if ;
   end loop ;
end rechercherOccurrence ;
```

1°)- Écrire la déclaration du type <Booleen> en utilisant le type **enum**
2°)- Écrire la déclaration du type <TabEntiers> en utilisant la directive **typedef** et la **déclaration** de la fonction C traduisant cette procédure
3°)- Écrire la **définition** de la fonction C traduisant cette procédure
4°)- Écrire l'appel de cette fonction avec les paramètres effectifs suivants :

```
TabEntiers  listeNotes ;
int  nbreNotes,  noteZero,  rangNote ;
Booleen  trouve ;
```

Exercice 2 :

Soit l'algorithme suivant :

```
-- procédure de recherche de la première occurrence de l'élément élé dans le sous-tableau tab
-- dont le premier  élément est pointé par adeb et le dernier par afin ;
-- si élé existe trouvé = VRAI et arang pointe sur élé dans tab ;
-- sinon trouvé = FAUX et arang est indéfini.

procedure rechercherSousTableau ( aDeb, aFin : in PtrEntier ; élé : in Integer ;
                                  trouvé : out Boolean ; aRang : out PtrEntier )

   fini : Boolean ; --indicateur d'arrêt de la recherche
   aCour : PtrEntier ; --pointeur sur l'élément courant du tableau

begin
   aCour := aDeb ;             -- se positionner sur le premier élément du sous-tableau
   fini := false;
   while not fini loop
      if aCour <= aFin then    -- tester si le pointeur courant est valide
         if élé = aCour.all then
            trouvé := true;    -- l'élément est trouvé,
            aRang := aCour ;   -- fournir le résultat de la recherche,
            fini := true;      -- arrêter la recherche
         else
            aCour := aCour + 1 ; -- passer à l'élément suivant
         end if;
      else
         trouvé := false;      -- l'élément n'est pas  dans le sous-tableau,
         fini := true;         -- arrêter la recherche
      end if;
   end loop;
end rechercherSousTableau ;
```

1°)- Écrire la déclaration du type <PtrEntiers> en utilisant la directive **typedef**
2°)- Écrire la **déclaration** de la fonction C traduisant cette procédure
3°)- Écrire la **définition** de la fonction C traduisant cette procédure
4°)- Écrire l'appel de cette fonction avec les paramètres effectifs suivants :

```
PtrEntier  premiereNote,  derniereNote, refNote ;
int  noteZero ;
Booleen  trouve ;
```

TD2 : l'allocation dynamique

Objectifs :

Étude de la bibliothèque standard du langage C, des fonctions d'allocation dynamique et mise en œuvre pour l'allocation mémoire des variables dynamiques.
Support de cours : Compléments de langage C - pages 51 à 53.

Exercice :

Écrire les déclarations et les instructions permettant de créer les variables dynamiques suivantes :

1) tableau de 30 entiers

2) tableau de 256 caractères

3) enregistrement de composé de 3 champs :
   - champ **codeProd** : entier
   - champ **designation** : tableau de 60 caractères
   - champ **puHT** : réel;

TD3 : les E/S sur fichiers

Objectifs :

Étude de la bibliothèque standard du langage C, de gestion des E/S sur fichiers et mise en œuvre avec des fichiers en organisation séquentielle.
Support de cours : Compléments de langage C - pages 34 à 47 , §3.1 à 3.8.

Exercice :

Développer une application interactive qui permet de constituer un annuaire téléphonique enregistré dans un fichier texte.

On structurera l'application de la façon suivante :

- une procédure de saisie d'une chaîne de caractères au clavier ,
- le programme principal qui ouvrira un fichier texte en écriture et qui répétera :
  - la saisie d'un nom, d'un prénom et d'un numéro de téléphone
  - l'enregistrement en format fixe de ces 3 chaînes dans une ligne du fichier

  et qui se terminera sur la saisie du nom égal à "stop"

Reprendre l'application en enregistrant les données dans un fichier binaire sous forme d'un enregistrement de 3 champs.

TD4 : les points de reprises

Objectifs :

Étude de la bibliothèque standard du langage C, des points de reprises et mise en œuvre pour le traitement des exceptions.
Support de cours : Compléments de langage C - pages 54 à 55.

Exercice :

Développer une application interactive qui permet de gérer les exceptions liées à l'ouverture d'un fichier en **lecture**.

On structurera l'application de la façon suivante :

- une procédure de saisie d'une chaîne de caractères au clavier
- une fonction qui ouvre un fichier dont la désignation est passée en paramètre.
  Si une exception est levée, un retour sera fait sur un point de reprise dans l'appelant
  sinon la fonction retournera le descripteur FILE* du fichier
- le programme principal met en place une procédure de reprise qui en cas d'erreur d'ouverture du fichier, permet de ressaisir la désignation du fichier pour ensuite l'ouvrir, dans la limite de **3 tentatives** infructueuses.
  S'il n'y as pas d'erreur à l'ouverture, le fichier sera simplement fermé car l'objectif n'est pas le traitement du fichier.