TD 8 : Structures

(et révision sur le passage de paramètres)

1°) Un **article** est une structure qui pourra contenir une **référence** d'article (long), son **prix** unitaire et la **quantité** disponible en stock, une **désignation** (chaîne de 20 caractères).
Définissez le type `Article`

2°) On suppose que l'on dispose de la fonction main suivante qui permet par menu de créer une base de données d'articles, puis de gérer les quantités en fonction des ventes.

```
#define MAX 100

void main(void)
{ Article stock[MAX];
  int nbArticles, n, i, reference ;
  nbArticles = 0;
  do
  { printf("0 : quitter l'application \n");
    printf("1 : saisie d'un nouvel article  \n");
    printf("2 : affichage du stock\n");
    printf("3 : modifier la quantité d'un article \n");
    scanf("%d", &n);

    switch(n)
    { case 1 : stock[nbArticles++] = creerArticle() ;
               break;

      case 2 : for (i=0 ; i<nbArticles ; i++)
                 afficherArticle(&stock[i]);
               break;

      case 3 : printf("donnez la reference\n");
               scanf("%d", &reference);
               i = rechercher(reference, stock, nbArticles) ;
               if (i<0) break;
               modifier(&stock[i]);
               break;
    }
  }
  while (n != 0);  /* on aurait pu écrire while (n) */
}
```

Après avoir regardé précisément l'utilisation des fonctions `creerArticle`, `afficherArticle`, `rechercher` et `modifier`, écrivez ces 5 fonctions en respectant la nature des paramètres et des résultats éventuels. En particulier :

- la fonction `creerArticle` remplit les champs d'un nouvel article en lisant les informations au clavier. Elle retourne cet article comme résultat (ce choix sur la forme du résultat n'est pas le plus judicieux d'un point de vue temps d'exécution, mais il permet d'illustrer le renvoi d'une structure).
- Les articles étant stockés dans leur ordre d'arrivée, la recherche d'une référence dans le stock se fera de façon séquentielle
- la fonction `modifier` devra
  - afficher toutes les données de l'article
  - saisir le nombre d'articles vendus
  - modifier la quantité
  - et afficher de nouveau toutes les données de l'article

Correction TD 8 : Structures

```
/* version non optimisee (sans allocation dynamique)*/
#include <stdio.h>
#define MAXcar 20
#define MAX 100

typedef struct
{ int ref ;
  float prix ;
  int quantite ;
  char designation[MAXcar] ;
} Article ;

Article creerArticle(void)
{ Article e ;
  printf("reference ?\n") ;
  scanf("%d", &(e.ref)) ;
  printf("prix ?\n") ;
  scanf("%f", &(e.prix)) ;
  printf("quantite ?\n") ;
  scanf("%d", &(e.quantite)) ;
  printf("designation ?\n") ;
  scanf("%s", e.designation) ; /* pas de & (c'est une chaîne ! ) */
  return e ;
}


void afficherArticle(Article *e)
{ printf("%4d %5.2f %4d %s \n", e->ref, e->prix, e->quantite,
         e->designation) ;
}

void modifier(Article *p)
{ int q ;
  afficherArticle(p) ;
  printf("quantite vendue ?\n");
  scanf("%d", &q);
  p->quantite = p->quantite-q ;
  afficherArticle(p) ;
}

int rechercher(int r, Article S[], int n)
{ /* recherche sequentielle. retourne -1 si abs du tableau */
  int i;
  for(i=0 ; i<n ; i++)
    if (S[i].ref == r) return i ;
  return -1;
}

void main(void)
{ Article stock[MAX];
```

```
  int nbArticles, n, i, reference ;
  nbArticles = 0;
  do
  { printf("0 : quitter l'application \n");
    printf("1 : saisie d'un nouvel article  \n");
    printf("2 : affichage du stock\n");
    printf("3 : modifier la quantité d'un article \n");
    scanf("%d", &n);

    switch(n)
    { case 1 : stock[nbArticles++] = creerArticle() ;
               break;

      case 2 : for (i=0 ; i<nbArticles ; i++)
                 afficherArticle(&stock[i]);
               break;

      case 3 : printf("donnez la reference\n");
               scanf("%d", &reference);
               i = rechercher(reference, stock, nbArticles) ;
               if (i<0) break;
               modifier(&stock[i]);
               break;
    }
  }
  while (n != 0);  /* on aurait pu écrire while (n) */
}
```