R101 : Initiation au développement

TD 11 : File statique en langage C

… en vous appuyant sur votre cours sur les piles :

- Dans cet exercice, on vous demande de gérer une file de messages.
- Chaque message sera une chaîne de caractères intégrée à une structure (dans un prochain TD, vous ajouterez un composant "date" à la structure qui correspond à la date de saisie du message).

1. Définitions des constantes, des types et des structures :
   a. *Vous utiliserez les constantes symboliques suivantes :*
      - MAX_MESSAGES : le nombre maximal de messages de la file (exemple : 20)
      - MAX_CAR : nombre maximal de caractères d'un message (exemple : 80)
   b. *Vous définirez 3 types utilisateurs :*
      - t_message : tableau de MAX_CAR caractères
      - t_element : une structure comportant UN message de type t_message (mais pas de date à ce stade)
      - t_file : une structure comportant 2 champs :
        tabElt : un tableau de t_element
        nb : le nombre d'éléments valides du tableau.
   c. *Vous définirez une constante appelée ELT_VIDE correspondant à un élément vide.*
      - Par exemple, l'élément comprend le message suivant : "------ce message est vide-------"

2. En vous appuyant sur les algorithmes et programmes présentés en cours, vous allez développer les procédures et fonctions suivantes :

initialiser()	La fonction retourne une file initialisée (tous les éléments seront initialisés avec l'élément vide)
estVide()	La fonction eetourne 1 si la file est vide, 0 dans le cas contraire
estPleine()	La fonction retourne 1 si la file est pleine, 0 dans le cas inverse
enfiler()	La procédure ajoute un nouvel élément à la file.
defiler()	La fonction permet d'enlever un élément de la file
vider()	La procédure permet de vider la file
tete()	La fonction retourne l'élément en tête de file, mais sans le retirer de la file.

3. … vous disposer également une procédure secrète afficherTous() qui affichera à l'écran tous les éléments. Celle-ci vous aidera dans la mise au point de votre programme.

4. Écrivez les prototypes des différentes procédures et fonctions dans le fichier "TD12_File_etu.c" donné.

5. Remarque : voici une portion de code qui vous permettra de saisir une chaîne de caractères comportant des espaces :

```
char msg[MAX_CAR] ;
printf("saisir un nouveau message : ");
fgets(msg,MAX_CAR, stdin);        // vider le buffeur de caractères (si nécessaire).
fgets(msg, MAX_CAR, stdin);       // saisie d'une chaine de caractères de MAX_CAR caractères au maximum.
msg[strlen(msg)-1]='\0';  // suppression du caractère '\n' de validation de fin de saisie.
```

6. Dans un second temps, vous compléterez votre programme avec les fonctions ou procédures suivantes :

supprimer_trop_anciens()	Supprimer les éléments les plus anciens de la file (le nombre d'éléments à supprimer sera un paramètre)
sauvegardeFichier()	Sauvegarde des messages de la file dans un fichier texte (le nom du fichier sera un paramètre) et vide la file.
lectureFichier()	Lecture des messages (le nom du fichier sera un paramètre). La file initiale sera écrasée

7. Le main() du programme est partiellement donné :

```
// déclaration des constantes symboliques
// définition des types
// définition des constantes
// prototypes des fonctions
// programme principal
int main(){
   // Declaration des variables
   t_file maFile;
   t_element elt;
   t_message msg;
   int choix;
   // initialisation
   maFile = initialiser();
   // ajouter quelques elements
   for (int i = 0 ; i<4 ; i++){
      sprintf(elt.message,"message %d", i);
      enfiler(&maFile, elt);
   }
   do{ // menu
      printf("-------------------------------------------------------\n");
      printf("0 : quitter\n");
      printf("1 : afficher le nombre d'elements dans la file ?\n");
      printf("2 : ajouter un element a la file\n");
      printf("3 : retirer un element et afficher le message \n");
      printf("4 : afficher le message de la tete de file\n");
      printf("5 : vider la file\n");
      printf("6 : la file est-elle vide ?\n");
      printf("7 : la file est-elle pleine ?\n");
      printf("8 : supprimer les messages trop anciens\n");
      printf("9 : sauvegarde dans un fichier texte et vider\n");
      printf("10: lecture des messages du fichier texte\n");
      printf("votre choix : ");
      scanf("%d", &choix);
      printf("-------------------------------------------------------\n");
      // traitement
      switch(choix){
         case -1 :   afficheTous(maFile); // fonction secrète ...
         case 0:     break;
         case 1:     // afficher le nombre d'éléments dans la file
                     break;
         case 2:     // ajouter un élément (à donner aux étudiants)
                     break;
         case 3:    // retirer un élément et afficher le message
                    break;
         case 4:    // afficher le message de la tête de file
                    break;
         case 5:    // vider la file
                    break;
         case 6:    // la file est-elle vide ?
                    break;
         case 7:    // la file est-elle pleine ?
                    break;
         case 8:     // supprimer les messages trop anciens
                    break;
         case 9:    //sauvegarde dans un fichier texte et vider
                    break;
         case 10:  //lecture des messages du fichier texte
                     break;
         default :   printf("erreur de saisie\n");
      }
   }while(choix != 0);
   return EXIT_SUCCESS;
}
// définitions des fonctions
```