

Examen de Info C

Durée : 1h

Aucun document autorisé

Mobiles interdits

Nom :

Prénom :

- 1. Écrivez de façon itérative la fonction `itos` qui renvoie la conversion d'un entier positif ou nul passé en paramètre, en sa représentation sous forme d'une chaîne de caractères. Aucune fonction standard de `string.h` n'est à utiliser.

```
/*
 * Antécédent : n >= 0
 * Rôle : renvoie la conversion de n en sa représentation sous forme
 * d'une chaîne de caractères.
 */
char *itos(unsigned int n) {
    // allouer un tableau de caractères mémoriser tous les chiffres
    char chiffres[sizeof (unsigned int)*3]; // log_2(sizeof(char)) = 3
    // décomposer l'entier n
    // note : les chiffres sont obtenus en ordre inverse
    int i = 0;
    do {
        chiffres[i++] = n%10 + '0';
        n/=10;
    } while (n!=0);
    // allouer la chaîne de caractères résultat de taille i+1
    // et y placer les chiffres dans le bon ordre
    char *s = malloc(i+1);
    int j=0;
    for (i--; i>=0; i--, j++) s[j] = chiffres[i];
    // mettre le caractère de fin de chaîne
    s[j+1] = '\0';
    return s;
}
```

Pour représenter les chaînes de caractères, on définit le type `String` suivant :

```
typedef struct string {
    char *s; // les caractères de la chaîne
    int lg; // longueur de la chaîne
} *String;
```

- 2. Écrivez la fonction `String string(const char *s)` qui renvoie une `String` initialisée à partir de la chaîne de caractères passée en paramètre.

```
String string(const char *s) {
    String ms = malloc(sizeof(struct string));
    ms->lg = strlen(s);
    // on alloue la place mémoire pour la nouvelle chaîne de car
    ms->s = malloc(sizeof(char)*ms->lg+1);
    // et on la copie
```

```
strcpy(ms->s, s);
return ms;
}
```

- 3. Écrivez la fonction `stringCat` qui renvoie une `String` concaténation de deux `String` passées en paramètre.

```
String stringCat(const String s1, const String s2) {
    String ms = malloc(sizeof(struct string));
    ms->lg = s1->lg+s2->lg;
    ms->s = malloc(ms->lg+1);
    ms->s = strcat(strcpy(ms->s, s1->s), s2->s);
    return ms;
}
```

- 4. Un fichier contient une suite d'entiers et de réels en alternance, d'abord un entier, puis un réel. on considère que le fichier est sans erreur. Attention, le fichier n'est pas un fichier de texte (pas de `fscanf`)!

Écrivez la procédure `somme` qui calcule la somme des entiers et celle des réels. Cette procédure devra posséder 3 paramètres. Un paramètre donnée, le nom du fichier, et deux paramètres résultats, les sommes des entiers et des réels.

```
void somme(const char *f, int *sInt, double *sDouble) {
    // ouvrir le fichier en lecture
    FILE *fp;
    if ((fp = fopen(f, "r"))==NULL) {
        perror(f);
        exit(errno);
    }
    // le fichier f est ouvert en lecture
    int x;
    double y;
    *sInt = *sDouble = 0;
    while (fread(&x, sizeof(int), 1, fp)>0) {
        // lire le réel
        fread(&y, sizeof(double), 1, fp);
        // sommer les entiers et les réels
        *sInt+=x;
        *sDouble+=y;
    }
    // fermer le fichier
    fclose(fp);
}
```

- 5. Dans le répertoire courant, le fichier qui contient la suite d'entiers et de réels se nomme monFichier. Écrivez la fonction `main` qui appelle la procédure `somme` précédente et qui écrit sur la sortie standard les deux sommes calculées.

```
int main(void)
{
    int sInt;
    double sDouble;
```

```
    somme("monFichier", &sInt, &sDouble);  
    printf("%d - %f\n", sInt, sDouble);  
  
    return  EXIT_SUCCESS;  
}
```

.....