ENSEIRB-MATMECA – Filière Electronique

Examen Langage C - PG108
2020-2021 2ème session
Jeudi 1er Avril 2021
Durée de l'épreuve : 2h / documents autorisés – sans calculatrice

1) Fonctions mystère:

Que font les fonctions suivantes ?
Pour chaque fonction, maximum : une phrase pour expliquer ce que fait la fonction, éventuellement une phrase supplémentaire par paramètre pour en expliquer l'utilité.

```
int timon(int pumbaa) {
     if ((pumbaa % 2) == 0)
          return 0
     return 1;
}
```

```
int pim(char *pam) {
     for (int poum=0; poum < 10; poum++)
          if (pam[poum] == 0)
               return 0;
     return 1;
}
```

```
double marie(char *toulouse, char *berlioz) {
     while (*toulouse) {
          if (*toulouse++ != *berlioz++)
               return 0 ;
     }
     return 1 ;
}
```

```
void castor(char *pollux) {
     while (*pollux) {
          if ((*pollux >= 'a') && (*pollux <= 'z'))
               *pollux += 'A' - 'a';
          pollux++;
     }
}
```

2) Exercice:

Écrivez un programme complet comprenant les inclusions de bibliothèque, la fonction *main* et éventuellement les sous-fonctions. Ce programme récupère un entier n depuis la ligne de commande et affiche tous les nombres premiers qui lui sont inférieurs.

Par exemple, si le programme s'appelle `listprem` , la commande `./listprem 10` génèrera l'affichage suivant :

```
2
3
5
7
```

3) Le *jeu* des 3 erreurs:

La fonction suivante compte le nombre de valeurs négatives dans un tableau de doubles. 3 erreurs se sont glissées dedans.
- Mentionnez chacune de ces trois erreurs
- Re-écrivez la fonction en les corrigeant

```
int count_negs(double *tableau, int longueur) {
     float negs = 0;
     for (int indice == 0;  indice < longueur; indice)
          if (tableau[indice] < 0) {
               ++negs;
          }
     return negs;
}
```

Précisions :
- l'algorithme est correct, seule la syntaxe est en cause
- l'indentation est bonne également (pas de piège)

4) Problème:

Nous nous intéressons à la création d'un histogramme. Pour simplifier le problème, nous utiliserons un tableau d'octets en guise de données sources. Il s'agit donc de compter, pour chaque valeur que peut prendre un octet, le nombre de fois ou il apparaît dans le tableau à analyser.

a) Quel est le type de donnée qui stocke une valeur entière sur un octet ? Dans la suite du sujet, nous utiliserons sa version non signée.

La fonction `saisir()` reçoit un tableau d'octets (le tableau à remplir) ainsi qu'un nombre de valeurs. Elle demande à l'utilisateur de saisir ces valeurs une par une et les mémorise dans le tableau fourni.

b) Quelle valeur doit/peut renvoyer la fonction `saisir()` pour pouvoir récupérer les valeurs entrées par l'utilisateur ?

c) Pourquoi est-til nécessaire d'indiquer le nombre de valeur à saisir alors que le tableau est fourni ?

d) Donnez le prototype de la fonctoin `saisir()`.

e) Ecrivez la fonction `saisir()`.

Le nombre d'occurence de chaque valeur sera stockée dans un tableau de 256 entiers (un entier pour chaque valeur de l'octet). Nous appellerons `signal`, le tableau d'octets et `histogramme` le tableau de 256 entiers.

f) Ecrivez la fonction `count()` qui reçoit les tableaux `signal` et `histogramme` ainsi que le nombre d'éléments dans `signal`, qui ne renvoie rien, et qui remplit `histogramme` en fonction des valeurs trouvées dans `signal`.

g) pourquoi n'est-il pas nécessaire de donner la taille de `histogramme` ?

h) Ecrivez une fonction qui, à partir des deux fonctions précédentes, affiche, pour chaque valeur d'octet, le nombre de fois où il apparaît dans le tableau saisi par l'utilisateur.

i) Lorsqu'elle est appelée, la fonction `saisir()` récupère des informations venant de l'entrée standard. Comment, sous UNIX, fournir le contenu d'un fichier par l'entrée standard au lieu de tout taper au clavier ?

j) Ecrivez la fonction `sub_sample()` qui transforme un histogramme sur 256 valeurs en un histogramme sur 64 valeurs.