Nom et Prénom : _____

Contrôle continu 2 (1h)

Hicham Janati

1 Tutoriel (10 pts)

IMPORTANT : Chaque question doit être traitée sans utiliser des notions présentées dans les questions qui viennent après ! Créez un fichier .c avec lequel vous allez effectuer ce tutoriel. Dans tous les bouts de codes suivants, nous supposons **stdlib** et **stdio** inclus.

1. **char est un type d'entier conçu pour stocker des variables (à 8 bits : soit $2^8 = 256$ valeurs) entre -127 et 127.**

```
1  int main(){
2      char a;
3      a = 66;
4      printf("La variable a = %d\n",a);
5      return 0;
6  }
```

Quelle est la sortie du programme ? Changez le "%d" par un "%c". Quelle est la sortie du programme ?

```
1 //
2 //
```

2. **Prenez 4 nombres voisins de a (par exemple 65,80,73 ..) et reportez la sortie que vous obtenez pour chacun. En déduire un moyen d'afficher la lettre K (en une ligne, mentionner le changement à effectuer au code).**

```
1 //
2 //
```

3. **Bien évidemment on ne va pas apprendre le numéro de chaque caractère. Pour stocker un caractère en mémoire, il suffit d'ajouter des guillemets :**

```
1  int main(){
2      char a;
3      a = 'B';
4      printf("La variable a = %d\n",a);
5      return 0;
6  }
```

Quelle est la sortie de ce programme ? Et si l'on change le '%d' en '%c' ?

4. **Ainsi, l'ordinateur interprète le type char comme étant un entier ou un caractère selon le type d'affichage donné à printf. Vous vous en doutiez sûrement : pour écrire en mémoire du texte, il suffit de créer un tableau de char : une chaîne de caractères ! Créez un tableau de char (non initialisé) qui contient un mot à 4 lettres de votre choix. (pas besoin de l'afficher)**

```
1  int main(){
2
3
4
5
6  }
```

5. **Comme avec les tableaux d'entiers où on initialisait avec des accolades comme *int T[4] = {0;1;3;4}*, on peut initialiser une chaîne de caractères avec des guillemets *char M[5]="hello"*. Complétez le code ci-dessous pour afficher les éléments de la chaîne un à un :**

```
1  int main(){
2     char M[15] = "Cool l'examen !";
3
4
5
6    }
```

6. **Compilez ce bout de code et expliquez l'erreur sortie dans le terminal :**

```
1  int main(){
2     char M[9] = "Yeah right";
3     return 0;
4     }
```

7. **C'est un peu pénible non ? J'ai dû compter à la main le nombre de lettres pour créer mon tableau de taille 15 et itérer sur toute la chaîne pour l'afficher. Eh ben les programmeurs y ont déjà pensé : l'une des différences majeures entre tableaux et chaînes est qu'il n'est pas nécessaire de connaître la taille d'une chaîne ! Encore mieux, pour afficher une chaîne de caractère, il suffit de la donner à printf avec l'option %s. Essayez par vous-même :**

```
1  int main(){
2     char M[] = "Cool l'examen !";
3     printf("%s", M);
4     return 0;
5     }
```

Mais comment C sait-il que le tableau est fini ? Eh ben en réalité, lorsqu'une chaîne est créée, C calcule le nombre de lettres, par exemple n mais réserve un tableau de taille n+1 en mémoire en ajoutant le caractère '\0' à la dernière case. Ce caractère signifie "STOP la chaîne se termine ici !". Ainsi, l'instruction *char H[5]="hello";* crée un tableau à 6 cases : | h | e | l | l | o | \0 | La case supplémentaire H[5] vaut alors '\0'.

Comment peut-on itérer sur un tableau de char ? Pour calculer la taille de la chaîne M ci-dessus par exemple ? (Donner l'idée)

```
1 //
2 //
3 //
```

8. **Complétez la fonction ci-dessous qui renvoie le nombre de caractère c trouvés dans une chaîne M :**

```
1 int Compteur(char M[], char c){
2
3
```

```
4
5
6 }
```

2 Exercices (10 pts)

Écrire une fonction qui inverse l'ordre d'un tableau. (3pts)
Écrire une fonction qui renvoie le max d'un tableau de manière récursive. (3pts)
Écrire une fonction qui renvoie le nombre d'éléments uniques d'un tableau *Exemple :* |4|5|3|3| *: 2* |1|1|0|0|1| *: 0* (4pts)

3 Problème (10 pts)

Le but de cet exercice est d'étudier les séries hyperharmoniques. On définit la série hyperharmonique d'ordre p par la somme :

$$S_p = \sum_{i=1}^{\infty} \frac{1}{i^p}$$

$p \leq 1 \Rightarrow S_p = +\infty$	$p > 1 \Rightarrow S_p < +\infty$	En particulier $S_2 = \frac{\pi^2}{6}$

Nous allons effectuer des approximations de cette somme avec une troncature à l'ordre N en sommant un tableau de taille N :

$$S_{p,N} = \sum_{i=1}^{N} \frac{1}{i^p}$$

Répondez aux questions suivantes en complétant les trois fichiers ci-dessous. (1pt pour la syntaxe)

1. Créez une fonction qui calcule la puissance p-ème d'un réel x. [bonus pour une forme récursive] (1.5pts)
2. Créez une fonction qui calcule la somme d'un tableau de réels. (2pts)
3. Créez une fonction qui prend un entier p, un entier N et vous renvoie une estimation de S_p. (3pts)
4. Créez un programme main qui demande à l'utilisateur de saisir p puis estime S_p avec N = 1000 et affiche sa valeur. (0.5pt)
5. Valeurs approchées de S_2 et S_3 ? (0.5pt)
6. En utilisant la fonction *log* de *math.h*, créez un programme qui donne une estimation de la constante gamma d'Euler (0.5pt) :

$$\gamma = \lim_{n \to \infty} (S_{1,N} - log(n))$$

7. En quoi l'utilisation des tableaux peut-elle être critiquée pour ce type de programme ? (1pt)

Fichier fonctions.h

```
1 float  Puissance(float x, int p);
2 ..... SommeTableau .............
3 ..... Estimation(int p, int N);
```

Fichier fonctions.c

```
1 float Puissance(float x, int p){
2
3
4
5
6
7 }
8
9
10
11
12
13
14
15
16
17
18
19
20
21
```

```
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46  .
```

Fichier main.c

```
1
2
3
4
5   int main(){
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20      }
```

```
1   //
2   //
3   //
4   //
```

Exercices

```
1   .
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
```

```
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46  .
```