Université de Nice-Sophia Antipolis — POLYTECH
ELSE3-FISE — 2025–2026

Examen de Info C – DS3

Durée : 1h
Aucun document autorisé
Mobiles interdits

Notez que les affirmations (antécédents, conséquents, rôles, et invariants) dans vos codes C entreront pour partie dans la note finale.

▶ 1. En utilisant uniquement la notation de pointeur, écrivez la fonction `space2hyphen` qui remplace dans une chaîne de caractères tous les caractères espaces par un tiret (`-`). La fonction prend en paramètre la chaîne à modifier et renvoie la chaîne modifiée. Par exemple, la fonction appliquée à la chaîne :

```
"bonjour à  tous, \
et a++";
```

renvoie `"bonjour-à--tous,-et-a++"`.

Question sur 3 pts

```
/*
 * rôle : renvoie la chaîne de caractères s dans laquelle chaque
 *        espaces a été remplacé par un tiret
 */
char* space2hypen(char *s) {
  char *p = s;
  while (*s)  {
    if (isspace(*s))
       // remplacer l'espace par un tiret
       *s = '-';
    s++;
  }
  return p;
}
```

.....

▶ 2. Déclarez une variable `s` initialisée à la valeur donnée au-dessus et écrivez sur la S/S le résultat de l'appel de `space2hypen(s)`.

Question sur 1 pt

```
char s[] = "bonjour à  tous, \
et a++";

printf("%s\n", space2hypen(s));
```


.....

▶ 3. À l'aide de **typedef**, déclarez le type `durée` pour représenter des durées *normalisées*. C'est une structure (**struct**) formée de 3 champs de type **int** : `h` pour les heures ($h \geqslant 0$), `m` pour les minutes ($0 \leqslant m \leqslant 59$), et `s` pour les secondes ($0 \leqslant s \leqslant 59$).

Question sur 1 pt :

```
typedef struct {
  int h; // h ⩾ 0
  int m; // 0 ⩽ m ⩽ 59
  int s; // 0 ⩽ s ⩽ 59
} durée;
```

.....

▶ 4. Écrivez la fonction `initDurée` qui possède 3 paramètres, $h \geqslant 0$, $0 \leqslant m \leqslant 59$ et $0 \leqslant s \leqslant 59$. Elle renvoie une durée normalisée initialisée avec les 3 paramètres h, m, s.

Question sur 1 pt

```
/*
 * Antécédent :  h ⩾ 0, 0 ⩽ m ⩽ 59 et 0 ⩽ s ⩽ 59
 * Rôle : renvoie la durée (h, m, s)
 */
durée initDurée(const int h, const int m, const int s) {
  assert(h>=0 && m>=0 && m<=59 && s>=0 && s<=59);
  return (durée) { h, m, s };
}
```

.....

▶ 5. Écrivez la procédure `écrireLnDurée` qui prend en paramètre une `durée` normalisée et l'écrit sur la S/S. Par exemple, elle pourra écrire $(2h, 36m, 12s)$.

Question sur 1 pt

```
/*
 * Antécédent : d durée normalisée
 * Rôle : écrit sur la sortie standard la durée normalisée d
 */
void écrireLnDurée(const durée d)  {
  printf("(%dh, %dm, %ds)\n", d.h, d.m, d.s);
}
```

.....

▶ 6. Écrivez la fonction `normalise` qui possède 3 paramètres, `h` $\geqslant 0$, `m` $\geqslant 0$ et `s` $\geqslant 0$ et qui renvoie une durée normalisée telle que `h` $\geqslant 0$, $0 \geqslant$ `m` $\geqslant 59$ et $0 \geqslant$ `s` $\geqslant 59$. Par exemple, la normalisation de la durée $2h$ $86m$ $120s$ est $3h$ $28m$ $0s$.

Question sur 4 pts

```
/*
 * antécédent : h, m, s ⩾ 0
 * rôle : renvoie la durée normalisée de (h, m, s)
 */
durée normalise(const int h, const int m, const int s) {
  assert(h>=0 && m>=0 && s>=0);
  durée dn = {h, m, s};
  if (dn.s>=60) {
    // calculer le minutes
    dn.m = dn.m + s/60;
    dn.s = dn.s%60;
  }
  if (dn.m>=60) {
    // calculer les heures
    dn.h = dn.h + dn.m/60;
    dn.m = dn.m%60;
  }
  return dn;
}
```

.....

► 7. Écrivez la procédure `somme` qui prend 3 paramètres `d1`, `d2` et `d3` de type `durée` (donc normalisée) et dont le rôle est de calculer la somme de `d1` et `d2`. Le résultat est dans `d3`. Par exemple, $1h\ 59m\ 59s + 0h\ 40m\ 59s$ égal $2h\ 40m\ 58s$.

Question sur 3 pts :

```
/*
 * antécédent : d1 et d2 deux durées valides
 * conséquent : *d3 = d1 + d2
 */
void somme(const durée d1, const durée d2, durée *d3) {
  *d3 = normalise(d1.h+d2.h, d1.m+d2.m, d1.s+d2.s);
}
```

.....

► 8. Écrivez le fragment de code qui déclare 3 variables `d1`, `d2`, `d3` de type `durée`, qui initialise les 2 premières à $1h\ 59m\ 59s$ et $0h\ 40m\ 59s$ qui calcule leur somme dans `d3`, et enfin l'affiche sur la S/S.

Question sur 1 pt :

```
durée d1 = initDurée(1, 59, 59);
durée d2 = initDurée(0, 40, 59);
durée d3;
somme(d1, d2, &d3);
écrireLnDurée(d3);
```

.....

► 9. Écrivez le programme qui prend comme paramètres programmes une suite (éventuellement vide)

de triplets qui représentent chacun une durée (h, m, s) (non forcément normalisée) qui calcule la somme des durées de la suite, et l'écrit sur la S/S. Vous utiliserez uniquement la notation de pointeur. On considère aussi que la suite de triplets est sans erreur. Par exemple, l'exécution du programme suivant : `./durée 2 86 120 1 40 12 2 77 301` écrit sur la S/S ($8h$, $30m$, $13s$).

Question sur 5 pts

```
int main(int argc, char *argv[]) {
  if (argc-->1) {
    // suite de triplets non vide
    durée som = initDurée(0, 0, 0);
    // parcourir tous les triplets et les additionner
    while ((argc-=3)>=0) {
      int h = atoi(*++argv);
      int m = atoi(*++argv);
      int s = atoi(*++argv);
      somme(som, normalise(h, m ,s), &som);
    }
    //
    écrireLnDurée(som);
  }
}
```

.....