

*La notation tiendra compte de la lisibilité du code et de la clarté des explications.
La barème n'est donné qu'à titre indicatif.*

Nom : Prénom :

Exercice 1 (5pts) Répondre par vrai ou faux aux questions suivantes. Barème: 0,5 pour une réponse correcte, -0,5 pour une réponse incorrecte et 0 en cas d'absence de réponse. Dans cet exercice, il n'est pas demandé de justification. Ici, **Vrai** signifie « toujours vrai » et **Faux** signifie « pas toujours vrai ».

- | | Vrai | Faux |
|--|--------------------------|--------------------------|
| 1. Une initialisation <code>int x = 1;</code> équivaut à la déclaration <code>int x;</code> immédiatement suivie de l'affectation <code>x = 1;</code> lorsque <code>x</code> est une variable locale dynamique..... | <input type="checkbox"/> | <input type="checkbox"/> |
| 2. On peut définir plusieurs variables de même nom à des niveaux différents de bloc. | <input type="checkbox"/> | <input type="checkbox"/> |
| 3. Plusieurs fonctions peuvent porter le même nom si elles diffèrent par leur liste de paramètres..... | <input type="checkbox"/> | <input type="checkbox"/> |
| 4. En l'absence d'initialisation explicite, une variable de type <code>int</code> est initialisée à 0.. | <input type="checkbox"/> | <input type="checkbox"/> |
| 5. Les définitions des fonctions de bibliothèque se trouvent dans des fichiers d'en-tête (extension <code>.h</code>)..... | <input type="checkbox"/> | <input type="checkbox"/> |
| 6. Une fonction peut prendre un nombre variable de paramètres..... | <input type="checkbox"/> | <input type="checkbox"/> |
| 7. L'expression <code>(++i + i++)</code> s'évalue en 2 si <code>i</code> vaut initialement 0..... | <input type="checkbox"/> | <input type="checkbox"/> |
| 8. La séquence suivante est incorrecte en C: <code>char c;</code>
<code>char p[10], *q;</code>
<code>p[1] = *q = c;</code> | <input type="checkbox"/> | <input type="checkbox"/> |
| 9. Dans la fonction <code>main()</code> , une instruction <code>return 0;</code> est toujours équivalente à une instruction <code>exit(0);</code> | <input type="checkbox"/> | <input type="checkbox"/> |
| 10. Pour créer le fichier objet <code>essai.o</code> à partir du fichier source <code>essai.c</code> , il est nécessaire que toutes les fonctions utilisées dans <code>essai.c</code> soient définies ou soient des fonctions de bibliothèque..... | <input type="checkbox"/> | <input type="checkbox"/> |

Exercice 2 (5pts) Un palindrome est une chaîne de caractères qui, lue à l'envers, est égale à elle-même. Par exemple, la chaîne vide, la chaîne "ici" et la chaîne "elle" sont des palindromes. On veut écrire une fonction de prototype

```
int est_palindrome(char *texte);
```

qui renvoie 1 ou 0 selon que la chaîne de caractères dont le début est pointé par l'argument `texte` est ou non un palindrome.

- Écrire une version itérative de `est_palindrome`.
- Écrire une fonction *récursive* `int est_palindrome_aux(char *texte, int longueur)`, qui renvoie 1 si la chaîne de longueur `longueur` commençant à l'adresse `texte` est un palindrome. Utiliser cette fonction pour écrire une nouvelle version de la fonction `est_palindrome`.

Exercice 3 (4pts) Écrire un programme qui lit un texte sur l'entrée standard et le réécrit sur la sortie standard, en supprimant toutes les voyelles.

Exercice 4 (6pts) On considère les fonctions suivantes :

```
void echange_1(int x, int y)
{
    int tmp;

    tmp = x;
    x = y;
    y = tmp;
}
```

```
void echange_3(int *x, int *y)
{
    int tmp;

    tmp = *x;
    *x = *y;
    *y = tmp;
}
```

```
void echange_2(int *x, int *y)
{
    int *tmp;

    tmp = x;
    x = y;
    y = tmp;
}
```

```
void echange_4(int *x, int *y)
{
    int *tmp;

    *tmp = *x;
    *x = *y;
    *y = *tmp;
}
```

Expliquer, de façon *détaillée* et *justifiée*, ce que l'on obtient après exécution de chacun des blocs suivants. Une réponse correcte mais non justifiée ne rapportera aucun point.

1. {int a = 1, b = 2; echange_1(a,b); printf("a = %d, b = %d\n", a, b);}
2. {int a = 1, b = 2; echange_2(&a,&b); printf("a = %d, b = %d\n", a, b);}
3. {int a = 1, b = 2; echange_3(&a,&b); printf("a = %d, b = %d\n", a, b);}
4. {int a = 1, b = 2; echange_4(&a,&b); printf("a = %d, b = %d\n", a, b);}

Dans chaque cas, dessiner l'état de la mémoire en fin d'appel des fonctions echange_1, echange_2, echange_3, et echange_4, et préciser ce qui sera affiché.