Identification /5

Prénom: .....

Nom: .....

Code permanent: .....

Veuillez choisir **une seule réponse** par question, avec une croix (×) ou un crochet (✓).

Q1. Sur une architecture 64 bits où un `int` occupe 4 octets et un `char`, 1 octet, qu'est-ce qui sera affiché par le fragment de code suivant?

```
char *s, c = 'x';
int n = 5, *p = &n;
double d, *q = &d;
printf("%zu %zu %zu %zu\n", sizeof s,
       sizeof c, sizeof *p, sizeof *q);
```

☐ `8 8 4 8`
☐ `1 1 4 8`
☐ `8 1 4 8`
☐ `8 1 8 8`

Q2. Considérez le fragment de code suivant:

```
int a = 1, b = 2;
int *const p = &a;
const int *q = &b;
```

Quelle instruction, placée après ce fragment, provoque une **erreur de compilation**?

☐ `*p = 3;`
☐ `q = &a;`
☐ `p = &b;`
☐ `b = *q + 1;`

Q3. Soit `x` une variable de type `int` et `p` une variable de type `int*` contenant l'adresse de `x`. Parmi les affirmations suivantes, laquelle est vraie?

- ☐ L'affectation `&x = p;` est valide, puisque `&x` désigne l'emplacement de `x`
- ☐ Après l'initialisation `int *r = p;`, l'instruction `*r = 7;` modifie la valeur de `x`
- ☐ L'expression `*x` est valide, puisque `x` est une *left-value*
- ☐ Si `q` est de type `const int*`, alors `int *r = q;` est acceptée sans avertissement

Q4. Qu'est-ce qui sera affiché sur la sortie standard par le fragment de code suivant?

```
struct Point { int x; int y; };
struct Point a = {1, 2}, b = a;
struct Point *p = &a, *q = &b;
b.x = 4;
p->y = q->x + 1;
q = p;
q->x = (*p).y * 2;
printf("%d %d %d %d\n",
       a.x, a.y, b.x, b.y);
```

☐ `1 5 10 2`
☐ `4 5 4 2`
☐ `10 5 10 5`
☐ `10 5 4 2`

Q5. La structure suivante permet de représenter une durée en minutes et en secondes:

```
struct Duree {
  unsigned int minutes;
  unsigned int secondes;
};
```

Écrivez une fonction C ayant la signature

```
void ajouter_duree(
  struct Duree *d,
  const struct Duree *e);
```

qui ajoute la durée pointée par `e` à la durée pointée par `d`, de telle sorte que le champ `secondes` de `d` demeure compris entre `0` et `59`. Vous pouvez supposer que c'est déjà le cas, pour les deux durées, avant l'appel, que le champ `minutes` n'est soumis à aucune contrainte (il peut dépasser `59`) et qu'il n'y a pas de débordement (*overflow*) à gérer. Si `d` ou `e` est `NULL`, alors la fonction ne fait rien. Par exemple, si `d` pointe vers la durée `{2, 50}` (2 minutes et 50 secondes) et `e`, vers la durée `{1, 15}`, alors `d` pointe ensuite vers la durée `{4, 5}`. *Note:* utilisez le verso au besoin.