Identification /5

Prénom: .....

Nom: .....

Code permanent: .....

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

Q1. Qu'est-ce qui sera affiché sur la sortie standard par le fragment de code suivant?

```
int t[] = {10, 20, 30, 40, 50};
int *p = t + 1, *q = t + 4;
printf("%d %d %d %d\n", *(p + 1),
       q[-1], (int)(q - p), *p + 1);
```

☐ `30 40 12 21`
☐ `40 40 3 30`
☐ `30 40 3 21`
☐ `30 50 3 21`

Q2. Considérez le programme `salut.c` suivant:

```
int main(int argc, char *argv[]) {
  if (argc != 2) {
    fprintf(stderr,
            "usage: salut NOM\n");
    return 2;
  }
  printf("Bonjour %s\n", argv[1]);
  return 0;
}
```

Une fois le programme compilé, qu'est-ce qui sera affiché dans le terminal par les commandes suivantes?

```
$ ./salut Alice Bob 2> erreurs.txt
$ echo $?
```

☐ `2` seulement
☐ `usage: salut NOM`, puis `2`
☐ `Bonjour Alice Bob`, puis `0`
☐ `Bonjour Alice`, puis `0`

Q3. Considérez la fonction suivante:

```
int suivant(int pas) {
  static int total = 10;
  total += pas;
  return total;
}
```

Que retourneront les appels `suivant(1)`, `suivant(2)` et `suivant(3)`, dans cet ordre?

☐ `11 12 13`
☐ `11 13 16`
☐ `1 3 6`
☐ `10 12 15`

Q4. Qu'affichera le fragment de code suivant?

```
struct Pays { char code[3]; int pop; };

int compare(const void *a,
            const void *b) {
  const struct Pays *x = a, *y = b;
  return strcmp(x->code, y->code);
}
// ...
struct Pays t[] = {{"BE", 12},
  {"CA", 40}, {"FR", 68}, {"JP", 124}};
struct Pays cle = {"FR", 0};
struct Pays *p = bsearch(&cle, t, 4,
  sizeof(struct Pays), compare);
printf("%d %d\n",
       (int)(p - t), p->pop);
```

☐ `2 0`
☐ `2 68`
☐ `3 68`
☐ `68 2`

Q5. Complétez le corps de la fonction suivante, qui retourne le nombre d'éléments du tableau `t`, de taille `n`, pour lesquels `predicat` retourne `true`:

```
unsigned int nombre_si(
  const int *t, unsigned int n,
  bool (*predicat)(int)) {
  // TODO
}
```

Exemple: si `est_pair` retourne `true` lorsque son argument est pair, alors

```
int v[] = {1, 2, 3, 4, 6};
nombre_si(v, 5, est_pair); // 3
```

Note: utilisez le verso au besoin.