Identification /5

Prénom: .....

Nom: .....

Code permanent: .....

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

- `sizeof(char) = 1`,
- `sizeof(int) = 4`,
- `sizeof(double) = 8` et
- `sizeof(T*) = 8` peu importe le type `T`.

Vous pouvez aussi supposer qu'aucun alignement n'est effectué par le compilateur.

Considérez le programme suivant:

```
#include <stdio.h>

struct S {
  double x;
  int* y[3];
};
union U {
  struct S* s;
  int* t;
  char c;
};

int main(void) {
  printf("%ld\n%ld\n", sizeof(struct S),
                       sizeof(union U));
  return 0;
}
```

Q1. Qu'est-ce qui sera affiché sur la 1re ligne de la sortie standard lors de son exécution?

☐ 8 ☐ 12 ☐ 20 ■ 32

Q2. Qu'est-ce qui sera affiché sur la 2e ligne de la sortie standard lors de son exécution?

■ 8 ☐ 9 ☐ 17 ☐ 32

Q3. Considérez le programme suivant:

```
#include <stdio.h>
#define i j

int main(void) {
  int i = 1;
  printf("%d\n", i + j);
  return 0;
}
```

Qu'est-ce qui se passera si on compile et on exécute le programme?

☐ La valeur `1` sera affichée
■ La valeur `2` sera affichée
☐ Le programme ne compilera pas
☐ Le programme plantera à l'exécution

Q4. Considérez le test Bats suivant:

```
@test "Un test Bats" {
  run echo 'abracadabra'
  # TODO
}
```

Par quoi doit-on remplacer `# TODO` pour que le test réussisse?

☐ `assert_failure`
☐ `assert_line --index 1 'abracadabra'`
☐ `assert_output --partial 'abrad'`
■ `assert_output --regexp 'a.*a'`

Q5. Considérez la fonction suivante:

```
bool p(const char* s) {
  for (int i = 0; s[i] != '\0'; ++i)
    if (isupper(s[i]))
      return true;
  return false;
}
```

Parmi les identifiants suivants, lequel est le plus approprié pour la fonction `p`?

☐ `isupper` ☐ `strisupper`
■ `strhasupper` ☐ `strtoupper`

Voir explications page suivante

Q1. Comme on suppose qu'il n'y a pas d'alignement, l'espace mémoire occupé par une structure peut être obtenu en calculant la somme des espaces mémoire occupés par chacun de ses champs. On obtient donc

$$\texttt{sizeof(struct S)} = \texttt{sizeof(double)} + 3 \cdot \texttt{sizeof(int*)} = 8 + 3 \cdot 8 = 8 + 24 = 32.$$

Q2. L'espace mémoire occupé par une union peut être obtenu en calculant le maximum des espaces mémoire occupés par chacun de ses champs. On obtient donc

$$\texttt{sizeof(union U)} = \max(\texttt{sizeof(struct S*)}, \texttt{sizeof(int*)}, \texttt{sizeof(char)}) = \max(8, 8, 1) = 8.$$

Q3. La directive `#define i j` remplace toutes les occurrences de `i` (en tant que mot) par `j`. Cela revient donc à compiler le programme suivant:

```
#include <stdio.h>

int main(void) {
  int j = 1;
  printf("%d\n", j + j);
  return 0;
}
```

Ce programme compile sans problème et s'exécute aussi sans problème. Il affiche donc la valeur `2` sur l'entrée standard.

Q4. Le test `assert_failure` échoue toujours. Le test `assert_line --index 1 'abracadabra'` échoue: il ne réussit que si la deuxième ligne (les indices commencent à 0) est exactement la chaine `abracadabra`, ce qui n'est pas le cas. Le test `assert_output --partial 'abrad'` échoue aussi: il ne réussit que si `abrad` apparaît comme sous-chaîne (caractères consécutifs) sur la sortie standard, or `abrad` n'est pas une sous-chaîne de `abracadabra`. Le test `assert_output --regexp 'a.*a'` réussit si l'expression régulière `a.*a` (le caractère `a`, suivi de 0 répétition ou plus d'un caractère quelconque, suivi du caractère `a`) apparaît sur la sortie standard. C'est bien notre cas ici, car `abracadabra` contient plusieurs fois le caractère `a`.

Q5. La fonction `p` parcourt la chaîne de caractères `s` jusqu'à ce qu'elle trouve un caractère majuscule et elle retourne alors `true`. Sinon, elle retourne `false`. Autrement dit, `p` retourne `true` si et seulement si la chaîne `s` contient au moins un caractère majuscule. Le nom `isupper` ne peut être utilisé pour renommer `p`, car il est déjà utilisé et C ne permet pas de surcharge de fonctions. Les noms `strisupper` (est-ce que la chaîne est majuscule?) et `strtoupper` (chaîne convertie en majuscules) ne sont pas appropriés, car ils ne reflètent pas le comportement de la fonction. C'est donc `strhasupper` (est-ce que la chaîne a une majuscule?) qui est le plus adapté.