Identification

Prénom: .....

Nom: .....

Code permanent: .....

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

Q1. Lorsque Bats est invoqué sur GitLab-CI, un rapport textuel est produit sur la sortie standard, similaire à celui-ci:

```
1..4
ok 1 - Input file opened
not ok 2 - First line of the input valid
ok 3 - Read the rest of the file
not ok 4 - Summarized correctly # TODO
```

Quel est le protocole décrivant la syntaxe respectée par ce flux de texte?

- ☐ BAP
- ■ TAP
- ☐ BATS
- ☐ TABP

Q2. Dans l'exemple précédent, quel nom donne-t-on à la première ligne (`1..4`) qui apparaît dans le flux de texte?

- ☐ Le cadre
- ■ Le plan
- ☐ Le décompte
- ☐ Le résumé

Q3. Parmi les casses (*case*) suivantes, laquelle est recommandée pour nommer les fichiers contenant du code source en C?

- ☐ `camelCase`
- ☐ `PascalCase`
- ☐ `kebab-case`
- ■ `snake_case`

Q4. Considérez la déclaration suivante:

```
double sum(const double* values,
           unsigned int num_values);
```

Quelle étiquette du standard Javadoc devez-vous utiliser pour documenter la variable `values`?

- ☐ `@arg`
- ■ `@param`
- ☐ `@input`
- ☐ `@return`

Les deux prochaines questions concernent la fonction suivante:

```
bool is_valid(const char* s) {
  while (*s != '\0') {
    if (!isalnum(*s) && *s != '_') return false;
    ++s;
  }
  return true;
}
```

Q5. Quelle valeur doit prendre `s` pour que la fonction `is_valid` retourne `false`?

- ☐ `""`
- ■ `"2?"`
- ☐ `"a"`
- ☐ `"_"`

Q6. Que se passera-t-il si on appelle la fonction `is_valid` avec la valeur `s = NULL`?

- ☐ La fonction va retourner `false`
- ☐ La fonction va retourner `true`
- ☐ Le programme ne s'arrêtera jamais
- ■ Le programme va planter

Q7. Qu'est-ce qui sera affiché sur la sortie standard par le programme suivant?

```
#include <stdio.h>
int main(void) {
  char s[] = "alpha";
  const char* p = s + 6;
  printf("%c%c\n", *(s + 1), *(p - 3));
  return 0;
}
```

- ☐ `ah`
- ■ `lh`
- ☐ `lp`
- ☐ `ap`

Q8. Qu'est-ce qui sera affiché sur la sortie standard par le programme suivant?

```
#include <stdio.h>
int f() {
  static int i;
  ++i;
  return i;
}
int main(void) {
  for (int i = 0; i < 3; ++i) printf("%d ", f());
  return 0;
}
```

- ☐ `1 1 1`
- ■ `1 2 3`
- ☐ Des valeurs numériques aléatoires
- ☐ Le comportement est indéterminé

Voir explications page suivante

Q1. Le protocole se nomme *test anything protocol* (TAP) et sa spécification est décrite à l'URL `https://testanything.org/`. L'acronyme BATS, qui signifie *Bash automated test system*, n'est pas un protocole mais bien un système informatique permettant de mettre en place des tests unitaires. Les deux autres acronymes n'ont pas de sens particulier dans le cours.

Q2. La première ligne d'un flux de texte respectant le protocole TAP est appelé *le plan*. Il décrit essentiellement le nombre de résultats de tests qui seront rapportés dans les lignes qui suivent.

Q3. Pour nommer les fichiers en C, c'est la syntaxe `snake_case` qui est encouragée, c'est-à-dire tout en lettres minuscules, en séparant les mots avec le caractère de soulignement `_`. On suggère aussi d'utiliser des noms de fichier le plus court possible.

Q4. Dans le standard Javadoc, les paramètres en entrée d'une fonction doivent être documentés avec l'étiquette `@param`. L'étiquette `@return` sert à documenter la valeur de retour. Les deux autres étiquettes ne font pas partie du standard Javadoc.

Q5. Le chaîne de caractères `s` est parcourue caractère par caractère jusqu'à ce qu'on atteigne le caractères de fin de chaîne `'\0'`. Si, en cours de route, on rencontre un caractère qui n'est pas alphanumérique (lettre ou chiffre) et qui n'est pas le caractère de soulignement `_`, alors on retourne `false`. Toutes les chaînes proposées sont bien formées et ne contiennent que des caractères autorisés, sauf `"2?"`, qui contient le caractère `'?'`.

Q6. Le programme va planter, car la condition d'entrée de la boucle `while` utilise l'opérateur de déréférencement, `*`, qui échoue lorsque le pointeur est nul.

Q7. Lors de l'affichage, `s + 1` pointe sur le deuxième caractère de la chaîne `s`, alors que `p - 3` pointe sur le quatrième élément (l'indice est $6 - 3 = 3$, et les indices commencent à 0). C'est donc le caractère `l` suivi de `h` qui seront affichés sur la sortie standard.

Q8. La variable `i` de la fonction `f` est statique. Elle est donc initialisée à `0` au début de l'exécution du programme et reste accessible jusqu'à la fin de l'exécution. Ainsi, au premier appel de la fonction `f`, la valeur retournée de `i` est `1`. Au deuxième appel, elle est incrémentée de `1`, donc elle passe à `2`. Finalement, au troisième appel, elle sera de `3`.