

Composition d'Informatique n°5

Remarques fréquentes

Remarques générales

- 0A. En OCaml, les chaînes de caractères `string` ne sont pas modifiables. Pour créer une chaîne particulière ne contenant pas plusieurs copies du même caractère (ce qui se fait avec `String.make`), on peut soit enchaîner les concaténations (`^`, au programme), soit construire une liste de chaînes qu'on concatène (`String.concat`, pas au programme).
 - 0B. Certaines questions de programmation **nécessitent** des commentaires en plus du code. C'est le cas lorsque :
 - * le code proposé utilise des fonctions auxiliaires non mentionnées par l'énoncé ;
 - * le code est long (compter 10 lignes en OCaml, 15 lignes en C).
- Si le code est trop compliqué, le correcteur ne arrêtera d'essayer de comprendre au bout d'un temps relativement court et ne mettra aucun point.
- 0C. On lève une exception `E` avec la syntaxe `raise E`, pas avec `failwith "E"`.

Remarques par question

- 2A. Même si cela ne change pas la borne sur $K(y_0)$, l'énoncé demande un algorithme de type « diviser pour régner », donc une solution qui fait inutilement deux appels récursifs au lieu d'un seul ne répond pas à la question.
- 2B. Il faut renvoyer une expression de type `string`, donc faire un appel à `string_of_int` comme le suggérerait l'énoncé.
- 3A. Avec une solution efficace (de type diviser pour régner), il n'y a pas de débordement de pile ! Le nombre d'appels récursifs est $\log_2(10^{10}) = 10 \log_2 10 \simeq 40$. De même, le temps de calcul n'est pas un problème, car chaque multiplication s'effectue en temps constant (et il y en a moins d'une centaine).
- 4A. La chaîne vide `""` doit avoir un antécédent par `phi`.
- 4B. La chaîne de taille 1 ne contenant que le caractère d'indice 0, `"\0"`, doit avoir un antécédent par `phi`.
- 6A. Il est préférable d'admettre un résultat intermédiaire honnêtement que de tenter une arnaque sortie du chapeau ! Une telle réponse sera sanctionnée.
- 12A. On s'attend, d'après les rappels de l'énoncés, à une solution utilisant un appel à `Hashtbl.find_opt`.
- 15A. Il faut utiliser le vocabulaire de variable locale/globale. Attention, la portée fait référence à l'identifiant, pas à l'emplacement mémoire où est stocké l'objet informatique.
- 19A. On attend une justification de la construction de l'automate de Glushkov.
- 19B. Il faut apprendre son cours. On n'attend pas la construction d'un automate quelconque, mais celui de Glushkov explicitement.
- 20A. Il est préférable et beaucoup plus simple d'écrire une fonction directement récursive. Utiliser une fonction auxiliaire avec un accumulateur, ou faire le travail avec une boucle ne crée que des complications.
- 20B. Il faut vérifier que le préfixe renvoyé est bien un mot de $\{a, b\}^*$ (et ne contient pas d'autres lettres).
- 25A. Il faut vérifier que le suffixe est vide après avoir lu le mot.
- 27A. Le cas de base est le cas $n = 0$, pas $n = 1$.
- 27B. Attention au filtrage masquant des variables. Lorsqu'on écrit `match x with |y ->`, la variable `y` est une nouvelle variable, qui masque toute autre variable pouvant s'appeler `y`.

Répartition des notes

