POLYTECHNIQUE MONTRÉAL
LE GÉNIE EN PREMIÈRE CLASSE

LOG 2810
STRUCTURES DISCRETES

TP2 : AUTOMATE D'ARBRE

Session	Automne 2018
Pondération	10 % de la note finale
Lieu de réalisation	L-2708; L-3712; L-3714
Taille des équipes	3 étudiants
Date de remise du projet	03 decembre 2018 (23h55 au plus tard)
Directives particulières	Soumission du livrable par moodle uniquement (https://moodle.polymtl.ca).
	Toute soumission du livrable en retard est pénalisée à raison de 10% par jour de retard.
	Programmation $C++$, Python ou Java
Les questions sont les bienvenues et peuvent être envoyées à: Les questions sont les bienvenues et peuvent être envoyées à : Mohameth Ndiaye (mohameth-alassane.ndiaye@polymtl.ca), Marc Anhoury (marc.anhoury@polymtl.ca), Saif Sajid (saif-eddine.sajid@polymtl.ca).	

1 Connaissances requises

- Notions d'algorithmique et de programmation C++.
- Notions d'automates et de langages.

2 Objectif

L'objectif de ce travail pratique est de vous permettre d'appliquer les notions théoriques sur les automates que nous avons vues en cours, sur des cas concrets mais hypothétiques, tirés de votre quotidien. A cet effet, il est question dans ce travail d'utiliser un automate pour encoder un lexique.

3 Mise en situation

L'ensemble des mots d'un langage forme un lexique. Dans les applications de correction automatique, le lexique qui peut être organisé sous la forme d'un automate à états finis, où chaque arc correspond à une lettre. L'objectif est ici de construire un automate à états finis capable de représenter le langage d'un lexique donné en entrée, puis d'utiliser cet automate pour faire la suggestion de mots à l'utilisateur d'une application offrant un service de correction automatique. Il s'agit donc de simuler l'une des fonctionnalités d'un correcteur automatique en utilisant l'automate correspondant au lexique, en mode génération. C'est-à-dire que l'automate est utilisé pour énumérer toutes les formes possibles d'un mot donné.
Vous avez été sollicité pour travailler sur le développement d'une application contenant un éditeur de

texte. Il vous a été assigné comme travail, de livrer une interface avec quelques fonctionnalités. Elle doit permettre à l'utilisateur de saisir du texte (à partir du clavier). Au fur et à mesure que l'utilisateur écrit chaque lettre (sauf le séparateur "espace" et les signes de ponctuation), votre application doit lui afficher, sur la base du lexique (automate), toutes les formes possibles du mot déjà constitué. Par exemple, sur la base du lexique dont l'extrait est présenté dans la table 1, dès que l'utilisateur aura écrit la lettre "***c***", tous les mots figurant dans l'extrait doivent lui être affichés. Dès qu'il aura écrit successivement les lettres cas"***c***","***a***","***s***", les mots "***case***", "***cases***", "***caser***", "***casier***", "***casiers***" doit s'afficher à l'écran.
Il vous a aussi été demandé de tenir des métadonnées sur les mots. Il s'agit de prendre les précautions permettant de connaître la fréquence d'utilisation de chaque mot, ainsi que les cinq mots les plus récemment utilisés. Ces données sont consultées sur demande.

Mots
caisse
caisses
caissier
caissiers
caissière
caissières
cas
case
cases
caser
casier
casiers

Table 1: Extrait d'un lexique

4 Tâches à réaliser

T1. Créer un automate à partir d'un lexique. Le lexique est donné sous la forme d'un fichier .txt.

T2. Créer une interface graphique qui permet à l'utilisateur de saisir du texte. Au fur et à mesure que l'utilisateur écrit un mot, votre application doit lui afficher, sur la base du lexique (automate), toutes les formes possibles du mot. Par exemple, sur la base du lexique dont l'extrait est présenté ci-haut, dès que l'utilisateur écrit la lettres "***c***", tous les mots figurant dans l'extrait doivent lui être afficher.

T3. Proposer puis implémenter une façon d'associer à chaque mot deux labels. Le premier est un entier qui indique le nombre de fois qu'il a été utilisé. Le second est un entier dont la valeur est soit 0, soit 1. L'entier 0 indique que le mot ne fait pas partie des cinq mots récemment utilisés. L'entier 1 indique que le mot fait pas partie des cinq mots récemment utilisés. Les labels sont mis 'jour dès que l'automate reconnaît le mot saisi par l'utilisateur.

T4. Afficher sur demande, pour chaque mot du lexique, ses labels correspondant.

5 Livrable

Le livrable attendu est constitué du code source et du rapport de laboratoire. Le livrable est une archive (ZIP ou RAR) dont le nom est formé des numéros de matricule des membres de l'équipe, séparés par un trait de soulignement (_). L'archive contiendra les fichiers suivants:

- les fichiers .cpp;

- les fichiers .h le cas échéant;
- le rapport au format PDF ou Word.

L'archive ne doit pas contenir de programme exécutable, de fichier de projet ou solution de Visual Studio, de répertoire Debug ou Release, etc. Les fichiers .cpp et .h suffiront pour l'évaluation du travail.

5.1 Rapport

Un rapport de laboratoire rédigé avec soin est requis à la soumission (format .pdf, maximum 8 pages). Sinon, votre travail ne sera pas corrigé (aussi bien le code source que l'exécutable). Le rapport doit obligatoirement inclure les éléments ou sections suivantes :

1. Page présentation : elle doit contenir le libellé du cours, le numéro et l'identification du TP, la date de remise, les matricules et noms des membres de l'équipe. Vous pouvez compléter la page présentation qui vous est fournie.
2. Introduction avec vos propres mots pour mettre en évidence le contexte et les objectifs du TP.
3. Présentation de vos travaux : une explication de votre solution.
4. Difficultés rencontrées lors de l'élaboration du TP et les éventuelles solutions apportées.
5. Conclusion : expliquez en quoi ce laboratoire vous a été utile, ce que vous avez appris, vos attentes par rapport au prochain laboratoire, etc.

Notez que vous ne devez pas mettre le code source dans le rapport.

5.2 Soumission du livrable

La soumission doit se faire uniquement par moodle.

6 Évaluation

Éléments évalués	Points
Qualité du rapport : respect des exigences du rapport, qualité de la présentation des solutions	1.5
Qualité du programme: compilation, structures de données, gestion adéquate des variables et de toute ressource (création, utilisation, libération), passage d'arguments, gestion des erreurs, documentation du code, etc.	1.5
Composants implémemtés : respect des requis, logique de développement, etc.	
T1	4
T2	5
T3	5
T4	3
Total de points	20

7 Documentation.txt

- http://www.cplusplus.com/doc/tutorial/
- http://public.enst-bretagne.fr/∼brunet/tutcpp/Tutoriel%20de%20C++.pdf
- http://fr.openclassrooms.com/informatique/cours/programmez-avec-le-langage-c