POLYTECHNIQUE MONTRÉAL
LE GÉNIE EN PREMIÈRE CLASSE

LOG 2810
STRUCTURES DISCRETES

TP2 : AUTOMATES ET LANGAGES

Session	Automne 2020
Pondération	10 % de la note finale
Lieu de réalisation	Session à distance
Taille des équipes	3 étudiants
Date de remise du projet	1 décembre 2020 (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 $C++$.
Les questions sont les bienvenues et peuvent être envoyées à: Aurel Josias Randolph (aurel.randolph@polymtl.ca), Saif-eddine Sajid (saif-eddine.sajid@polymtl.ca), Mohameth-Alassane Ndiaye (mohameth-alassane.ndiaye@polymtl.ca).	

1 Connaissances requises

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

2 Objectif

L'objectif de ce travail est de vous permettre d'appliquer les notions de théories du langage vues en cours, sur des cas concrets mais hypothétiques, tirés de votre quotidien. À cet effet, il est question dans ce travail de porter sur ordinateur un jeu de société à deux joueurs.

3 Mise en situation

On considère le jeu de société *Mastermind* dans lequel sont opposés deux joueurs, ou l'ordinateur et un seul joueur. Le premier joueur (ou l'ordinateur) définit un code secret, qui est un mot tiré d'un lexique et constitué des lettres d'un alphabet donné. Le second joueur, appelé le *mastermind*, doit alors deviner ce code secret en un nombre minimal d'étapes, en proposant, à chaque étape, un code de son choix. Après chaque proposition de code par le *mastermind*, le nombre d'erreurs commises par ce dernier (nombre de lettres mal placées) est affiché. La partie se termine si le *mastermind* a découvert le code secret, ou encore, si le nombre de coups permis a été atteint.

4 Description

Un étudiant du département de génie informatique et génie logiciel passionné de jeux de société émet l'idée de développer un jeu de *Mastermind* sur l'ordinateur. Il détermine que le jeu doit se diviser en 2 phases principales:

1. **Initialisation du jeu**
   Cette phase consiste à créer le lexique duquel sera tiré le code secret à deviner.
2. **Déroulement du jeu**
   Cette phase comprend 2 étapes:
   - Le choix du mot à deviner. Ce dernier est sélectionné, à partir du lexique, soit au hasard par l'ordinateur, ou par le premier joueur. Dans ce second cas, certaines suggestions peuvent être faites au premier joueur afin de l'assister dans sa tâche de sélection du code secret.
   - Le jeu. Une fois le code secret sélectionné, le *mastermind* (le second joueur) aura pour tâche de deviner ce dernier, en un nombre minimal d'essais.

L'étudiant se rappelle des notions de structures discrètes acquises, et décide de modéliser le problème à l'aide d'automates et de machines de Mealy. Toutefois, à court de temps, il sollicite votre aide pour l'aider avec l'implémentation des différentes fonctionnalités de son application. Lors de votre rencontre, il vous expose son projet et vous explique les termes suivants :

- **Implémentation du lexique et automate à états finis**
  L'ensemble des codes secrets forme un lexique qui, dans le contexte actuel, 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 au premier joueur lorsqu'il doit sélectionner le code à deviner. En d'autres termes, l'automate est utilisé pour énumérer toutes les formes possibles d'un mot donné.

  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***" et la touche `Entrée`, tous les mots figurant dans l'extrait doivent lui être affichés. Dès qu'il aura écrit successivement les lettres cas"***c***","***a***","***s***" et la touche `Entrée`, les mots "***case***", "***cases***", "***caser***", "***casier***", "***casiers***" doivent s'afficher à l'écran.

Mots
caisse
caisses
caissier
caissiers
cas
case
cases
caser
casier
casiers

Table 1: Extrait d'un lexique

- **Validité du code secret et machine de Mealy**
  Une machine de Mealy est une machine à états finis $M = (S, I, O, f, g, s_0)$ où :

- S est un ensemble fini d'états ;
- I est un alphabet de variables d'entrée ;
- O est un alphabet de variables de sortie ;
- $f : S \times I \to S$ est une fonction de transition qui attribue un nouvel état à chaque couple (état,entrée) ;
- $g : S \times I \to O$ est une fonction de transition qui attribue une sortie à chaque couple (état,entrée) ;
- s_0 est l'état initial.

Dans notre problème, le but de la machine de Mealy est de compter le nombre d'erreurs dans la proposition du joueur, en comparaison au secret. Si le joueur a trouvé le secret, la machine doit sortir 0 erreur ; sinon, chaque mauvaise lettre comptera pour une erreur dans le décompte final.

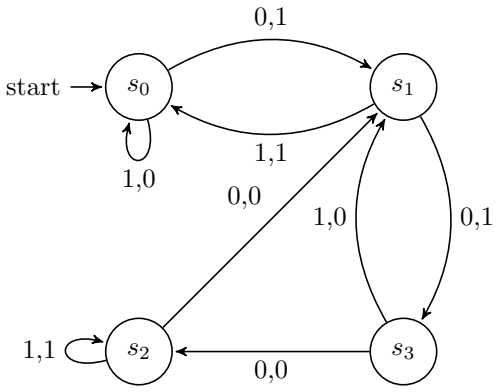

Figure 1: Diagramme d'états

S	f		g	
	0	1	0	1
s_0	s_1	s_0	1	0
s_1	s_3	s_0	1	1
s_2	s_1	s_2	0	1
s_3	s_2	s_1	0	0

Table 2: Table d'états

5 Composants à implémenter

C1. Écrire une fonction `creerLexique()` qui permet de créer un automate à partir d'un lexique, dont le nom est passé en paramètre. Le lexique est donné sous la forme d'un fichier .txt et comprend la liste des combinaisons disponibles pour les joueurs.

C2. Écrire une fonction `creerVerif()` qui permet de créer la machine de Mealy responsable de la vérification de la proposition du joueur.

C3. Écrire une fonction `modeAuto()` qui lance le jeu en mode *Un joueur*. Dans ce mode de jeu, l'ordinateur choisit au hasard un secret dans le lexique.

C4. Écrire une fonction `modeVersus()` qui lance le jeu en mode *Deux joueurs*. Dans ce mode de jeu, un premier joueur choisit un mot dans le lexique en demandant à l'ordinateur d'afficher des suggestions suivant les premières lettres entrées par le joueur - *cf.* ci-dessous.

C5. Créer une interface qui affiche le menu suivant :

- 1. Initialisation du jeu
- 2. Partie contre l'ordinateur
- 3. Deux joueurs
- 4. Quitter

Notes

- L'option 1 permet de lire le lexique afin de créer l'automate représentant les différentes combinaisons possibles.
- Les options 2 et 3 permettent de démarrer le jeu, respectivement en `modeAuto` et `modeVersus`.
- L'option 1 (l'initialisation du jeu), doit être appelée, au moins 1 fois, avant les options 2 et 3.
- Lors du mode de jeu *Deux joueurs*, la première étape consiste à choisir le secret. L'interface doit demander à l'utilisateur quel mot il désire choisir. L'utilisateur entre une chaîne de caractères, puis il tape sur la touche `Entrée`. L'automate doit alors définir si le mot est bien dans le lexique : si c'est le cas, l'interface demande si l'utilisateur veut réellement sélectionner ce mot. Dans tous les cas, s'il y a lieu, l'automate doit afficher une liste de suggestions, *i.e.* les mots qui commencent par la chaîne entrée.

  Exemple de sélection d'un code secret (sur la base du lexique présenté dans la table 1)

  > Quel code?
  Réponse: ba
  Ce mot n'existe pas dans le lexique

  Suggestion(s)
  Aucune...

  > Quel code?
  Réponse: ca
  Ce mot n'existe pas dans le lexique

  Suggestion(s)
  caisse
  caisses
  caissier
  caissiers
  cas
  case
  cases
  caser
  casier
  casiers

  > Quel code?
  Réponse: case

  Suggestion(s)
  case
  cases
  caser

  Voulez-vous sélectionner ce code? (1:oui/0:non)
  Réponse: 0

> Quel code?
Réponse: caisse

Suggestion(s)
caisse
caisses

Voulez-vous sélectionner ce code? (1:oui/0:non)
Réponse: 1

Le code sélectionné est: caisse

Début de la seconde partie...

- Lorsque la seconde partie du jeu est lancée, l'interface doit afficher le nombre de lettres du secret à découvrir. Si la proposition du second joueur possède une erreur de longueur de chaîne, ou bien si elle contient une lettre qui n'est pas dans l'alphabet, il faut annuler la proposition et demander une nouvelle. Sinon, à chaque proposition du joueur, l'interface affiche le nombre d'erreurs qu'il a fait. Au bout de quinze tentatives ratées (propositions non rejetées ou pas), le jeu s'arrête et l'interface doit afficher le secret avant de retourner au menu principal.
- Les codes secrets n'ont pas d'espaces.

6 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.

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.

6.1 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. Ajoutez le diagramme de classes complet, contenant tous les attributs et toutes les méthodes ajoutés.

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.

6.2 Soumission du livrable

La soumission doit se faire uniquement par moodle.

7 Évaluation

Éléments évalués	Points
Qualité du rapport : respect des exigences du rapport, qualité de la présentation des solutions	2
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.	2
Composants implémentés : respect des requis, logique de développement, etc.	
C1	3
C2	3
C3	2
C4	4
C5	4
Total de points	20

8 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