

1

La codification des caractères est donnée par la table suivante :

Caractère	\0	;	=	?	{	}	+ -	%	\$	[0 - 9]	[A - F]	[G - Z]	[ \t]	autre
Code entier	0	1	2	3	4	5	6	7	8	9	10	11	12	13

### Travail à accomplir:

1. Implémenter l'analyseur lexical du mini-langage. Le programme doit, entre autres, inclure :
  - (a) la déclaration des tokens et de la variable globale d'attribut,
  - (b) la définition de la fonction *car\_suivan()*
  - (c) la définition de la fonction *get\_lexem()* permettant de retourner le dernier lexème reconnu,
  - (d) la définition d'une fonction *atoh()* permettant de convertir une chaîne composée d'une suite de chiffres hexadécimaux en la valeur numérique correspondante ; en d'autres termes, l'équivalent de la fonction *atoi()* mais pour des nombres hexadécimaux,
  - (e) la définition de la fonction *token\_suivan()*
  - (f) la fonction principale *mai()* qui, par appels répétés à *token\_suivant()*, obtient et affiche la liste des couples *< token, attribut >*.
2. Effectuer diverses exécutions sur des programmes sources avec ou sans commentaires, et avec ou sans erreurs lexicales.
3. Modifier l'analyseur pour permettre :
  - (a) la saisie de programmes sur plusieurs lignes;
  - (b) l'affichage des messages d'erreurs lexicales comme dans l'exemple suivant :  
*Ligne 5 position 13 : '#' caractère illégal !*  
 où 13 indique la position du caractère illégal dans la ligne 5 et non pas sa position dans le programme dans son ensemble.

### Indication :

On aura besoin pour cela :

- d'utiliser des variables supplémentaires pour contenir les numéros de ligne et de colonne (la position dans la ligne) du dernier caractère lu,
- de modifier légèrement les fonctions :  
*car\_suivan()*, *erreur\_lexicale()* et *reculer()* pour mettre à jour les nouvelles variables et modifier l'affichage.