UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ciencias

Integrantes:
Marco Silva Huerta
Adrián Aguilera Moreno


# Lógica Computacional

# Práctica 6

- **Convertidor:**
  Todos los casos deberán ser listas, no se acepta hacer una conversión de un símbolo en solitario es decir *char_binario(a,L).* devuelve false.
  Comenzamos entonces por el caso base, si nos pasan una lista vacía para convertirla devolvemos la misma lista vacía.
  El segundo paso es ir caso a caso, de modo que toma el único char y lo convierte haciendo recursión sobre los hechos. Para la parte final recorremos toda la lista que nos estén pasando haciendo la recursión en el caso unitario y una vez echa la conversión se agrean a la lista convertida.
  Esto es lo mismo para cuando queremos ir de char a binario y de binario a char.

- **Cubos:**
  `sobre` nos indica que cubo esta sobre algún otro, *i.e.*, `sobre(X, Y)` indica que `X` esta sobre `Y`.

  `hastaArriba` nos indica quién ya no tiene un cubo sobre de el mismo.

  `bloqueado` nos dice si el cubo que se pasa como parámetro tiene un cubo arriba de el o no.

  `hastaAbajo` nos indica quién es el cubo que esta abajo en la pila de cubos.

  Las funciones auxiliares se especifican como comentario en el programa lógico.

- **Autómatas:**
  `estado_final` es el estado de aceptación de la lista que se le pasa al autómata si es que esa lista es aceptada por el lenguaje que genera el autómata.

  `estado_inicial` este estado es el primero por el que pasa la lista recibida y por donde inicia siempre (pues solo hay un estado inicial).

  `transicion` es nuestra función de transición donde están definidos todos los cambios de estados.

  `termina` esta función nos indica si la lista que se pasa como parámetro es aceptada por el autómata o no, después de sus respectivas llamadas recursivas.

  `aceptar` es la función que recibe una sola cadena y que por omisión llama a `termina` con estado inicial $q_0$.

- **Mezcla:**
  El primer método es un ordenamiento por selección, haciendo uso de las funciones *select* y *member* en el método auxiliar *menor* que se encarga de hacer las todas las comparaciones.

  Para el método mezcla hay 2 casos base:

  - Donde nos pasan dos listas vacías, devolvemos la lista vacía
  - Donde mezclamos una lista vacía y una lista con elementos (ordenados) darán únicamente la lista ordenada.

  Teniendo esto pasamos a realizar la mezcla tomando en cuenta 3 situaciones:

  $$X > Y, \quad X = Y, \quad X < Y$$

  Estas son la situaciones que se nos presentan al comparar las dos listas de modo que el operador de corte (**!**) se aplica después de hacer dicha comparación. Es decir que si se cumple que $X > Y$ el operador corta las ramas siguientes, caso contrario sigue.
  Para poder mezclar, únicamente podemos hacerlo con las dos listas ordenadas así que se creó un método auxiliar *ordenada* que devuelve true o false si es que la lista que le pasan esta ordenada. Aplicándolo al método *mezclar* verifica si estas listas están ordenadas.

Matriculas:

1. Marco Silva Huerta: 316205326.
2. Adrian Aguilera Moreno: 421005200.