Computabilidad

La idea es reducir la MT a Halt, para ello creamos un método IDDLE que equivale a una MT en el que relacionemos lo que reconoce la MT inicial pero involucramos otra MT, con la cual, llegaremos al absurdo.

Ejercicio 1

> Demuestra que no existe una MT que dado un código de otra MT diga si hay fragmentos de código sin utilizar

Sea $U(C)$ la MT que determina que código no se utiliza en C, y el método E que recibe una MT y una cadena, que representa una MT con el siguiente seudocódigo:

```
E(M,w):
    x = 0
    M(w)
    x = x+1
    return
```

Luego, notemos que la variable x declarada se utiliza $\iff$ $M(w)$ se detiene, por lo que, si ejecutamos $U(E(M,w))$ y no retorna código sir usar $\implies$ $M(w)$ se detuvo, de lo contrario sabremos que $M(w)$ no para,lo cual equivale a resolver $Halt$, contradicción.

Ejercicio 2

> Diga se es computable la MT que reconoce las MT que reconocen un lenguaje regular.

Sea C la MT que reconoce las MT que reconocen si una cadena $w \in L$, siendo L un lenguaje regular. Como es regular el lenguaje $\implies$ $\exists$ un autómata A que reconoce las cadenas de este, luego, definamos el método E de la siguiente forma:

```
E(w):
    x = A(w)
```

```
    M(w)
    return x
```

Nótese que la MT que representa a E retorna si el autómata del lenguaje L reconoce la cadena w, lo cual hace $\iff M(w)$ se detiene.

Al ejecutar $C(E(w))$, este retorna una respuesta $\iff M(w)$ se detiene, y esto es equivalente a resolver $Halt$, lo cual es absurdo.