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.

Ejercicio 3 (Busy Beavers 🦫)

> Sea $BB(n)$ un método que dada una cantidad de estados retorna la mayor cantidad de unos que imprime una MT con esa cantidad de estados. Demuestre que $BB(n)$ no es computable.

Supongamos que $BB(n)$ es computable $\implies$ existe una MT que representa BB. Sea $S(T)$ una MT que dada una MT retorna su cantidad de estados, y $M = U(T, c)$ la MT universal *(una máquina que simula el comportamiento de otra máquina T)*, con la particularidad de que, se le pasa como parámetro una cinta vacía, y por cada transición de T, M imprime un 1 en la cinta c, luego, este número de unos debe ser menor o igual que $BB(n)$, o sea, la cantidad de transiciones de una MT con n estados debe ser menor o igual que la cantidad de unos máxima que puede imprimir una MT con esos estados.

Luego, creamos el método E con el siguiente comportamiento:

```
E(T,c):
    // c es la cinta vacía
    n = S(T) // cantidad de estados de T

    for i in U(T,c):
        // por cada iteracion de U
        x = c
        if(BB(n) < x):
            return false;

    return true;
```

Este método itera por las transiciones de $U(T, c)$ que es equivalente a que por cada transición de T revise si en c, o sea, si la cantidad de trransiciones que ha hecho es mayor que $BB(n)$, en cuyo caso indica que T no parará, lo cual es equivalente a resolver $Halts$, contradicción,$porloquenoexisteMTquerepresenteBB, por lo que este método no es computable.