

**Universidad Politécnica de Tulancingo**

**Reporte de Practica**

por

**Jorge Perales Diaz**

**Ingeniería en Sistemas Computacionales**

Asignatura:

Lenguajes y Autómatas

Nombre del Catedrático:

Rafael Stanley Núñez Cruz

Tulancingo de Bravo, Hidalgo

Enero – Abril 2017

**U**

**P**

**T**



**Índice**

**Introducción………………………………………………………………. 3**

**Objetivo……………………………………………………………………. 3**

**Materiales…………………………………………………………………. 3**

**Planteamiento……………………………………………………………. 4**

**Desarrollo………………………………………………………………… 4**

**Resultados……………………………………………………………….. 4**

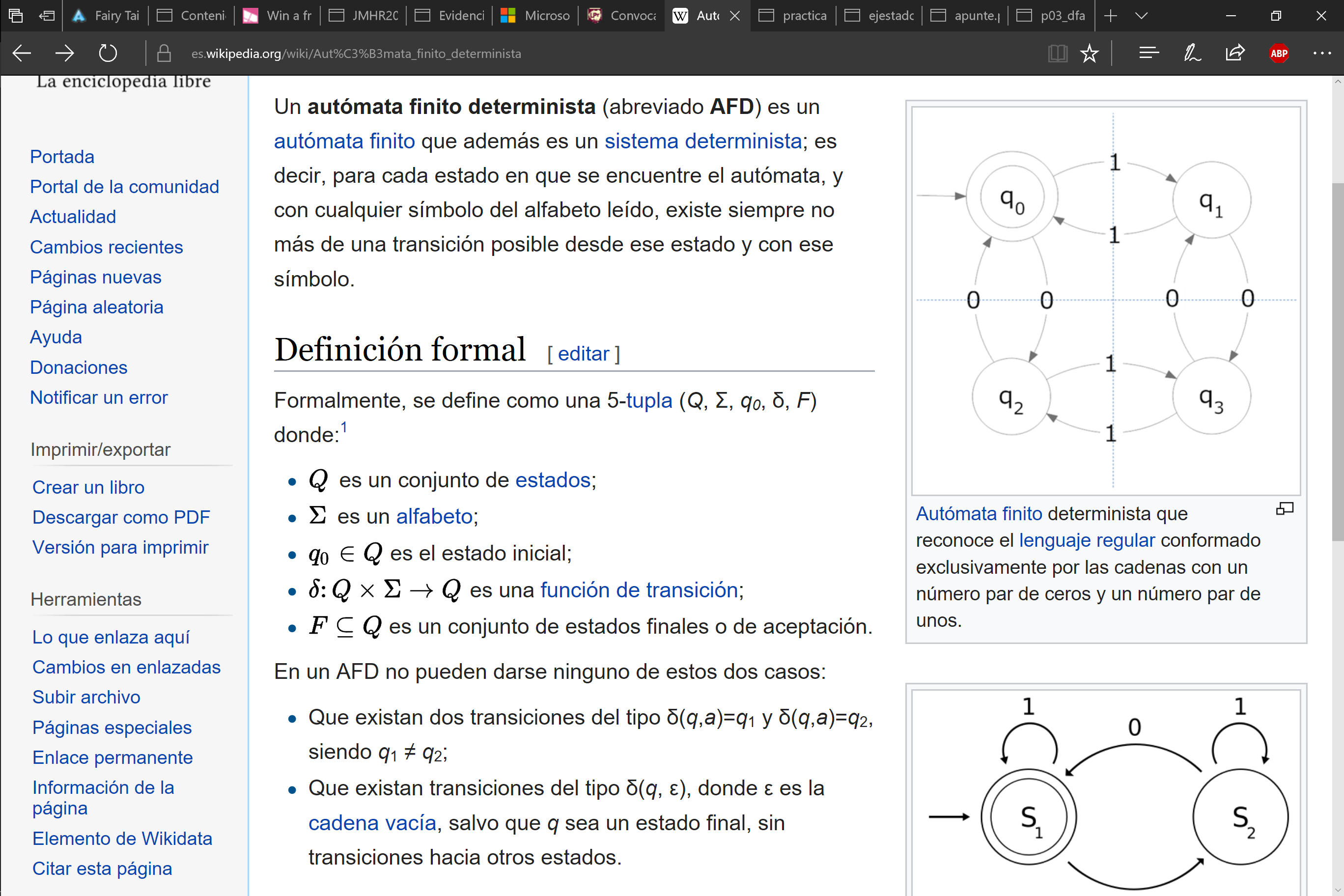
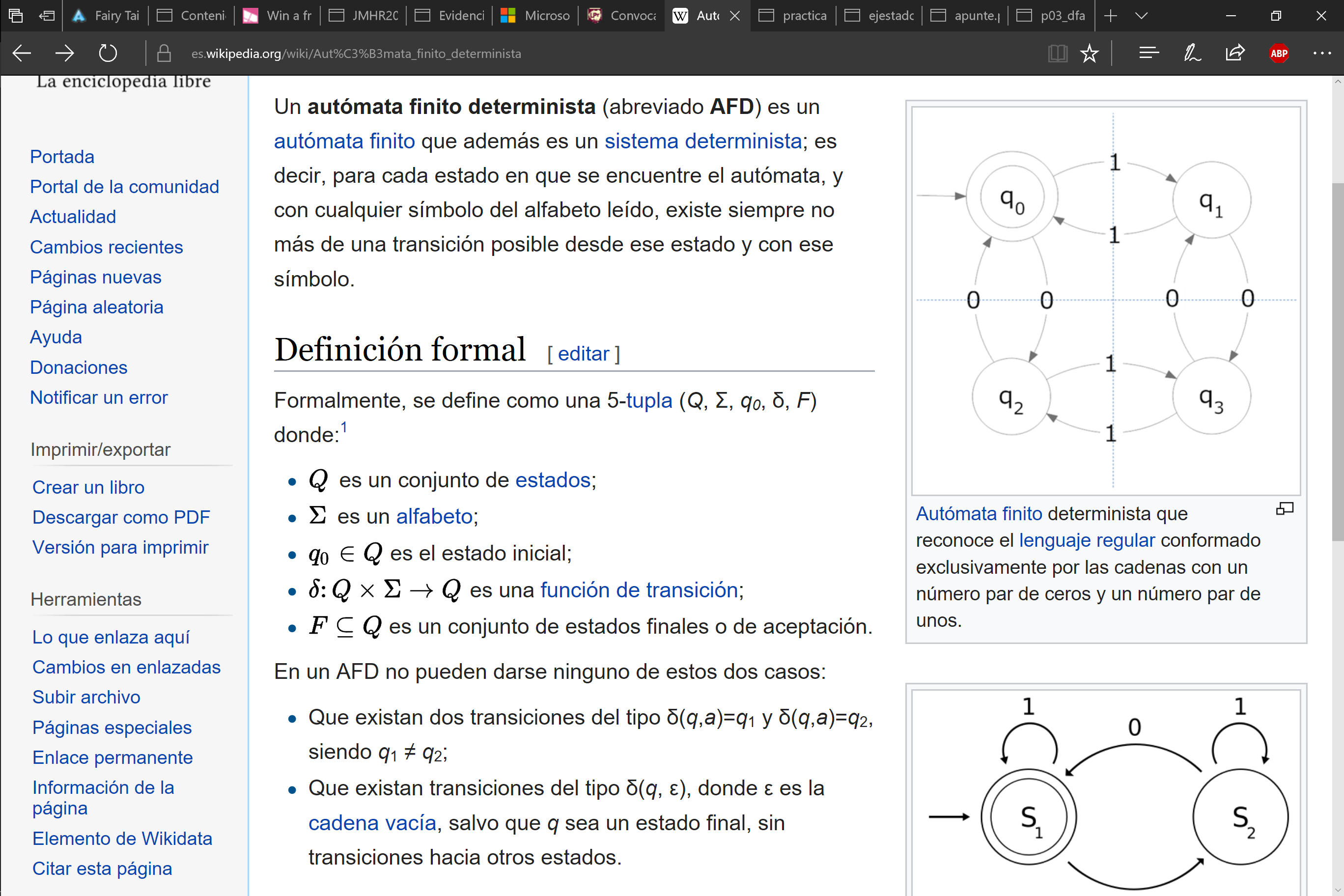
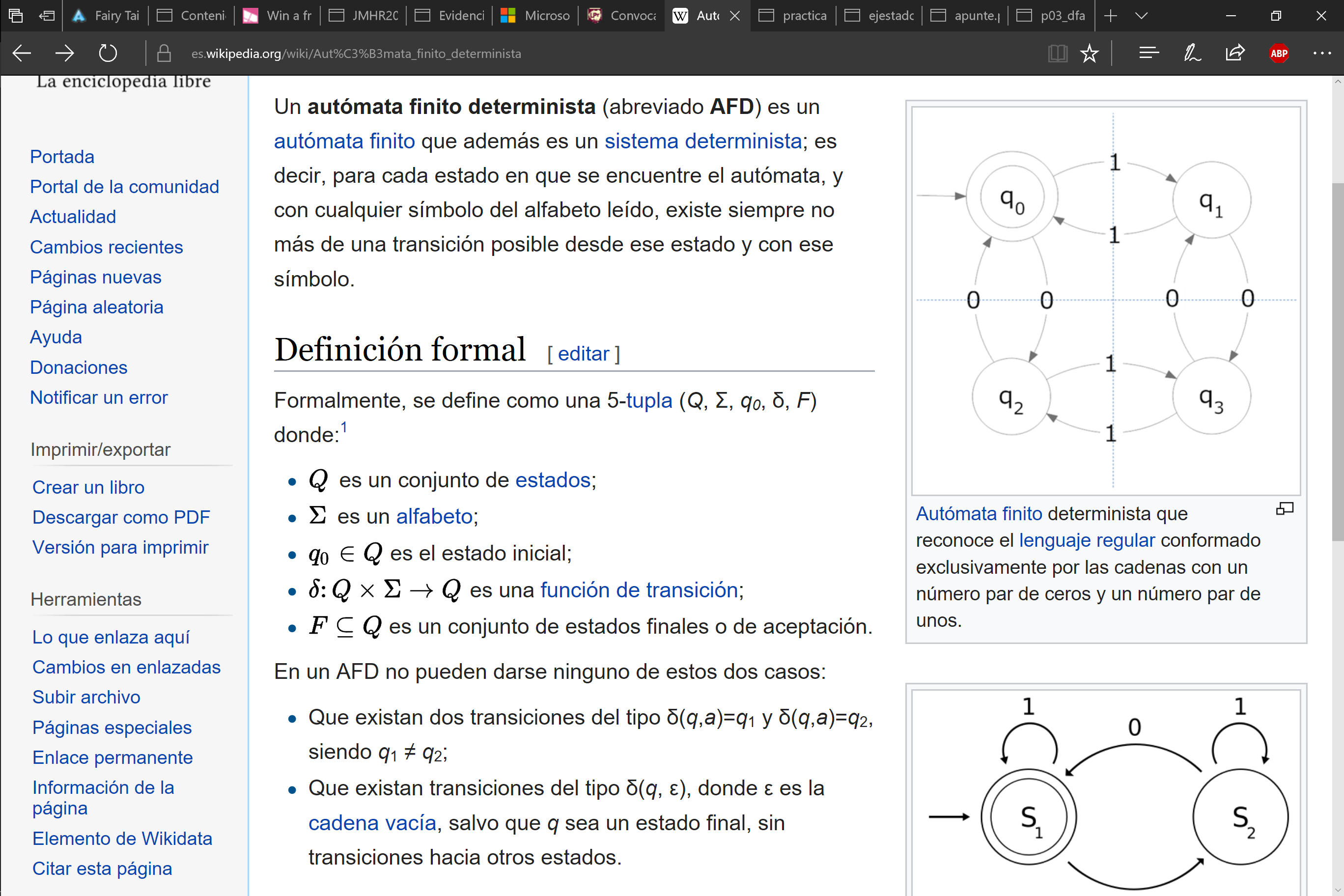
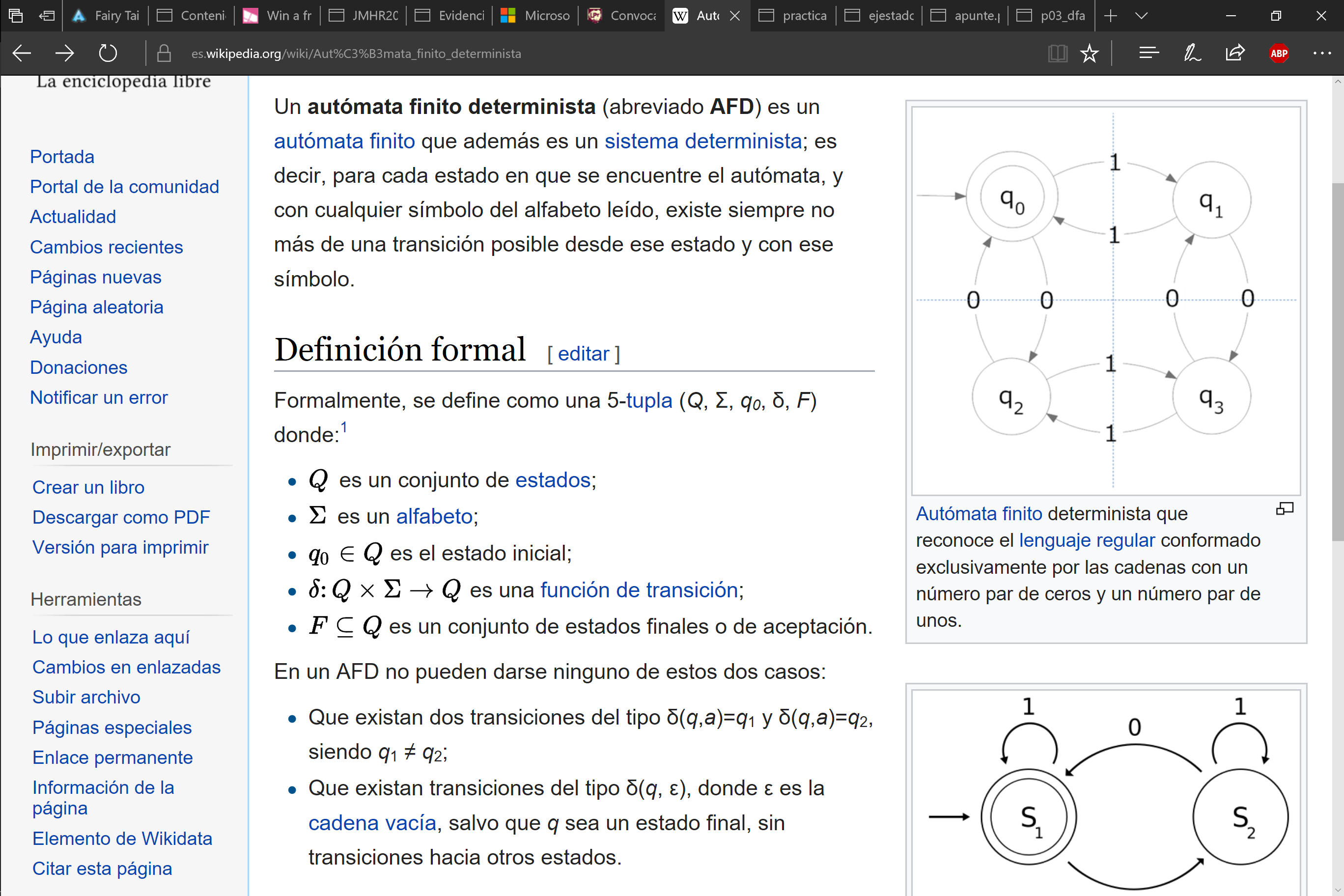
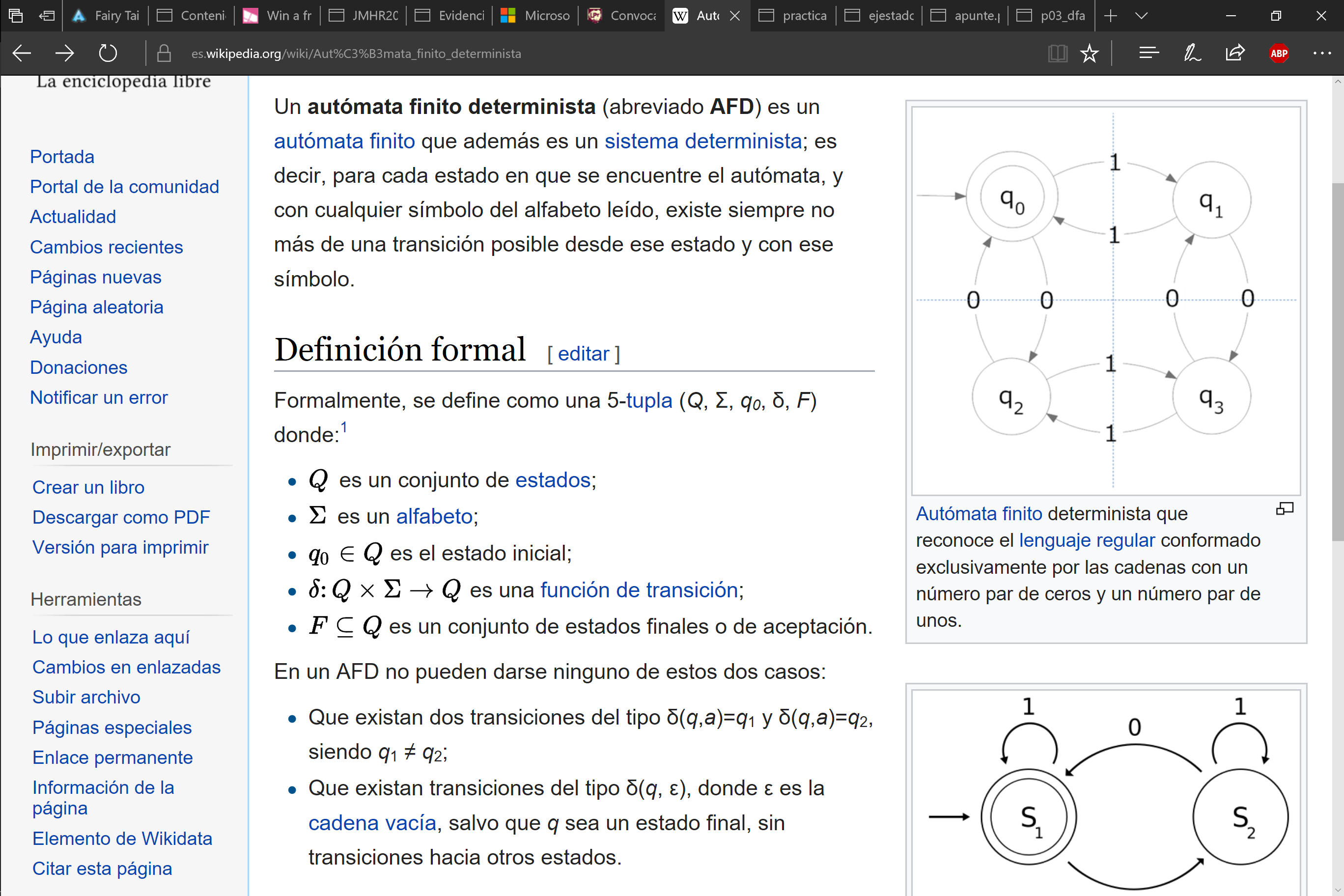
**Conclusiones……………………………………………………………. 5**

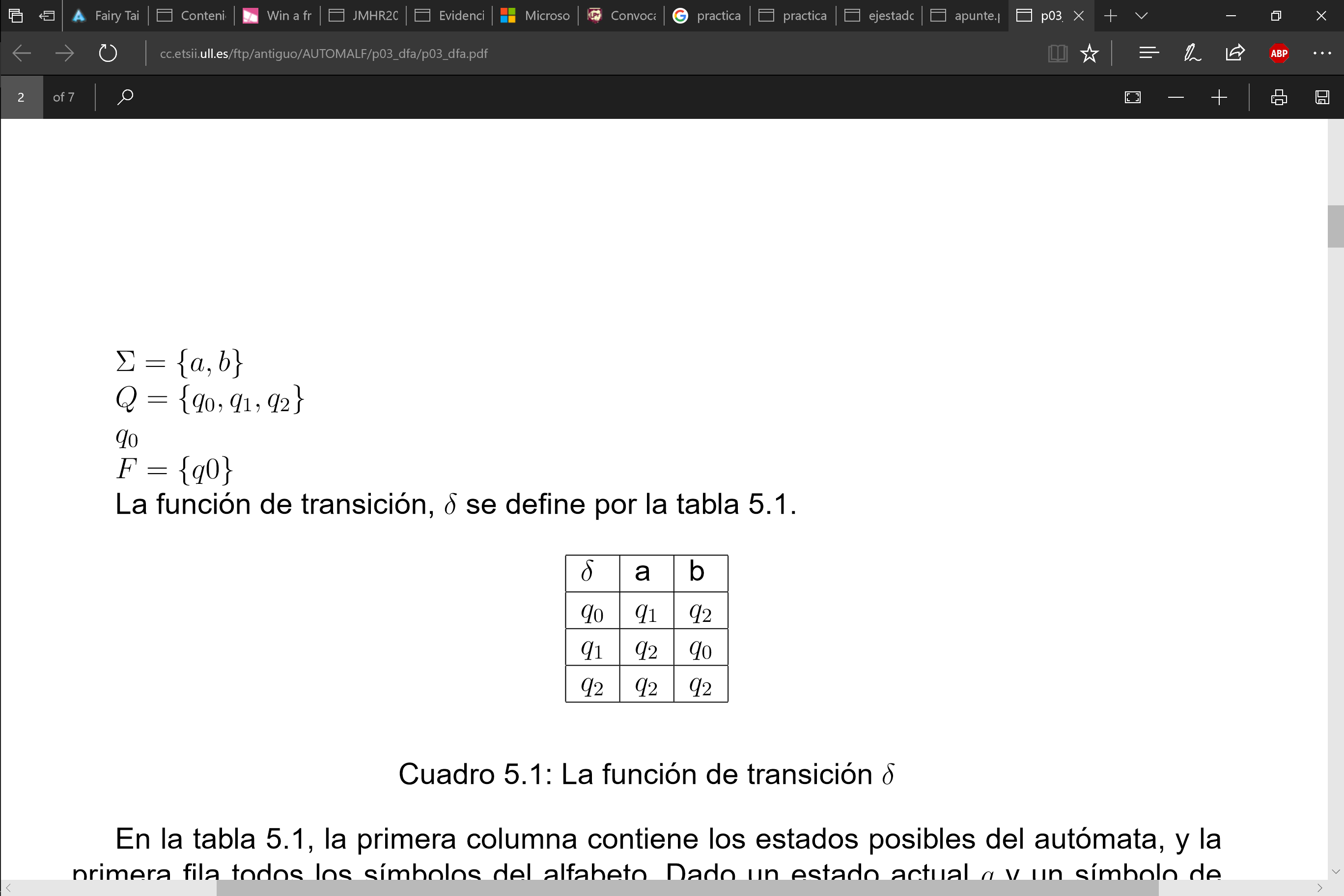
**Repositorio………………………………………………………………. 5**

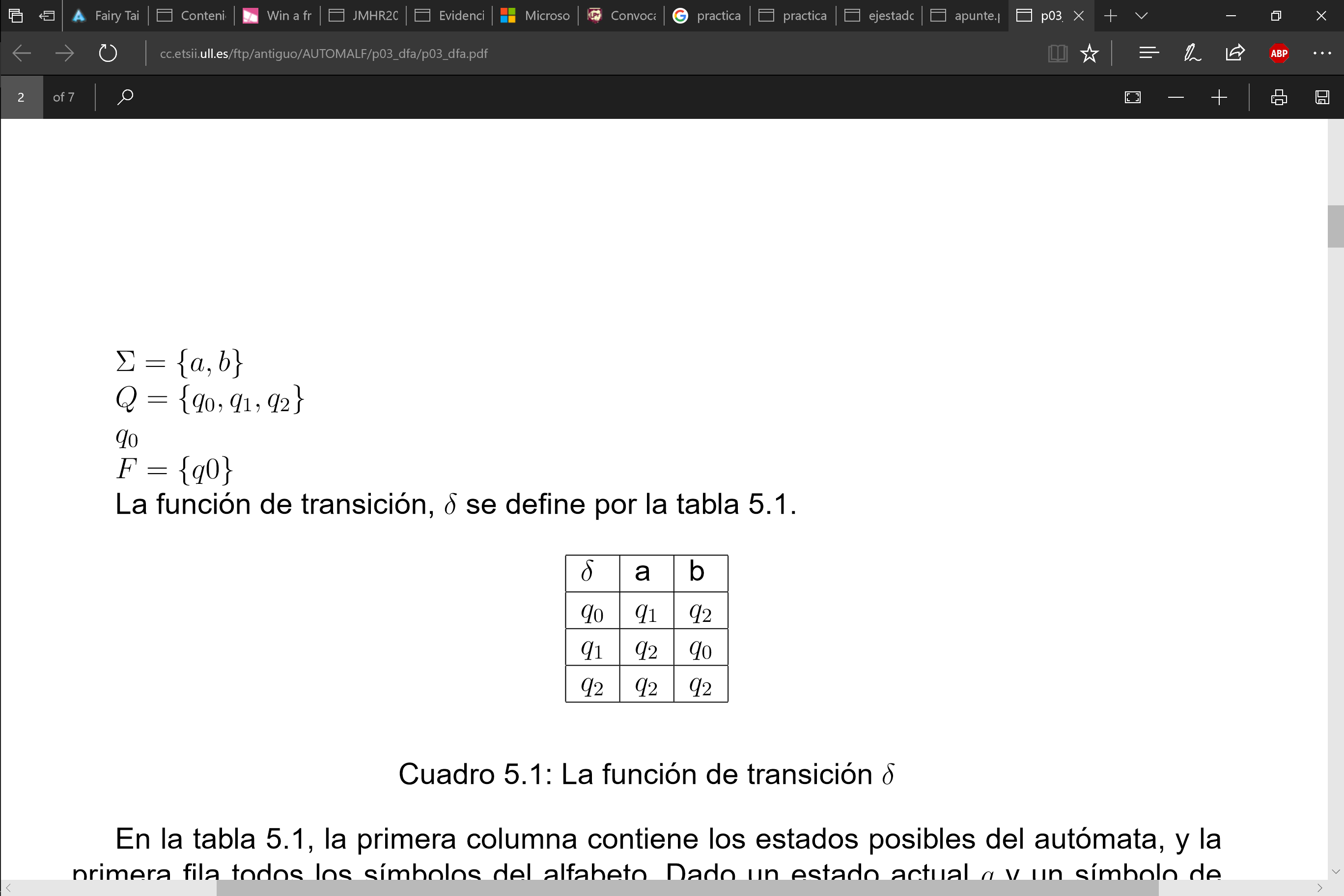
**Introducción**

Un **autómata finito determinista** (abreviado **AFD**) es un autómata finito que además es un sistema determinista; es decir, para cada estado en que se encuentre el autómata, y con cualquier símbolo del alfabeto leído, existe siempre no más de una transición posible desde ese estado y con ese símbolo.

Formalmente, se define como una 5-tupla (*Q*, Σ, *q0*, δ, *F*) donde:[]

* Q {\displaystyle Q\,} es un conjunto de [estados](https://es.wikipedia.org/wiki/Estado_(inform%C3%A1tica));
* Σ {\displaystyle \Sigma \,} es un alfabeto;
* q 0 ∈ Q {\displaystyle q\_{0}\in Q} es el estado inicial;
* δ : Q × Σ → Q {\displaystyle \delta \colon Q\times \Sigma \to Q} es una función de transición;
* F ⊆ Q {\displaystyle F\subseteq Q} es un conjunto de estados finales o de aceptación.





**Objetivo**

El objetivo es que el alumno con esta práctica pueda familiarizarse en el entorno practico y pueda visualizar aplicaciones en la vida cotidiana, además de adquirir experiencia en el uso de los autómatas finitos deterministas y que pueda quedar consolidado este tema, así como el saber identificar las partes de la construcción de un AFD que son; estados, estado inicial, estados finales y transiciones.

**Materiales**

* Procesador de textos: Sublime Text 3.
* Compilador: minGW.

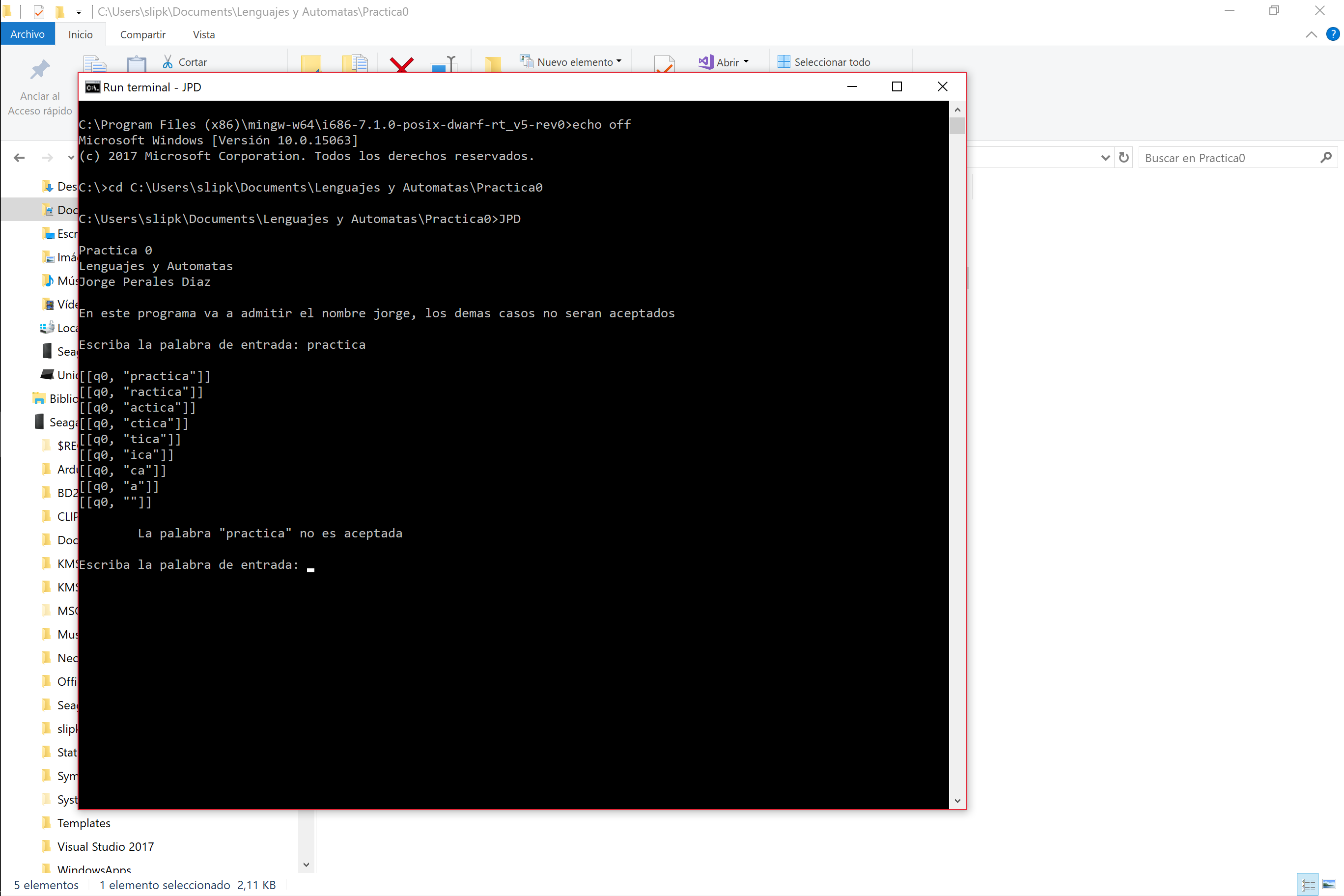
**Planteamiento**

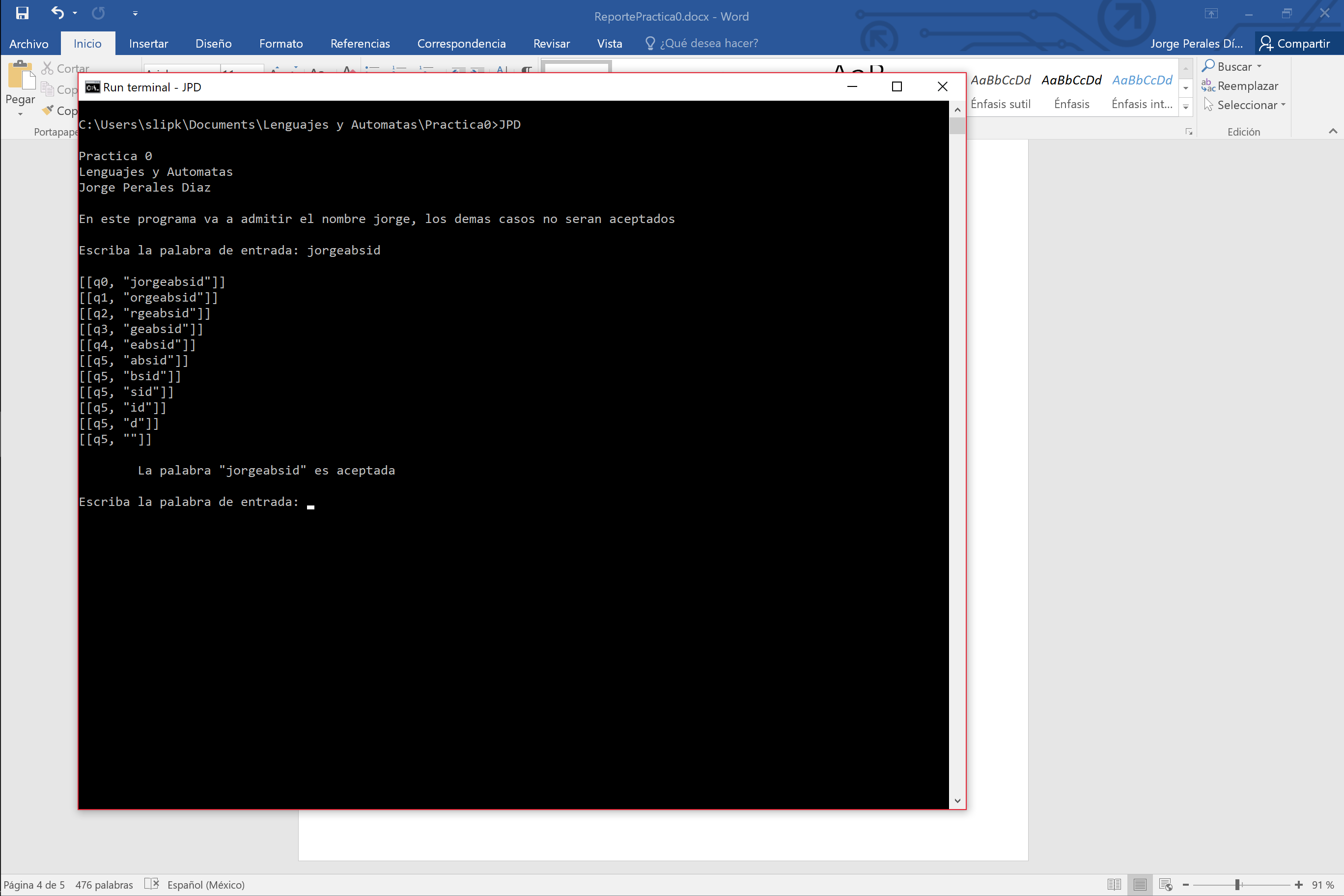
La práctica consistirá en la realización de un programa desarrollado en C++, el cual debe definir la quíntupla de un autómata ﬁnito determinista, que solamente acepte “un nombre”, en este caso el nombre a aceptar es “jorge”, después de haber consolidado la quíntupla se debe definir la matriz de relaciones que formaran el grafo y la definición de este autómata, una vez hecho esto, se procederá a pedir la palabra a evaluar, luego el programa mostrara las configuraciones que se vayan haciendo en el proceso de leer a palabra ingresada y finalmente decir si la palabra ingresada es aceptada o no en el autómata desarrollado.

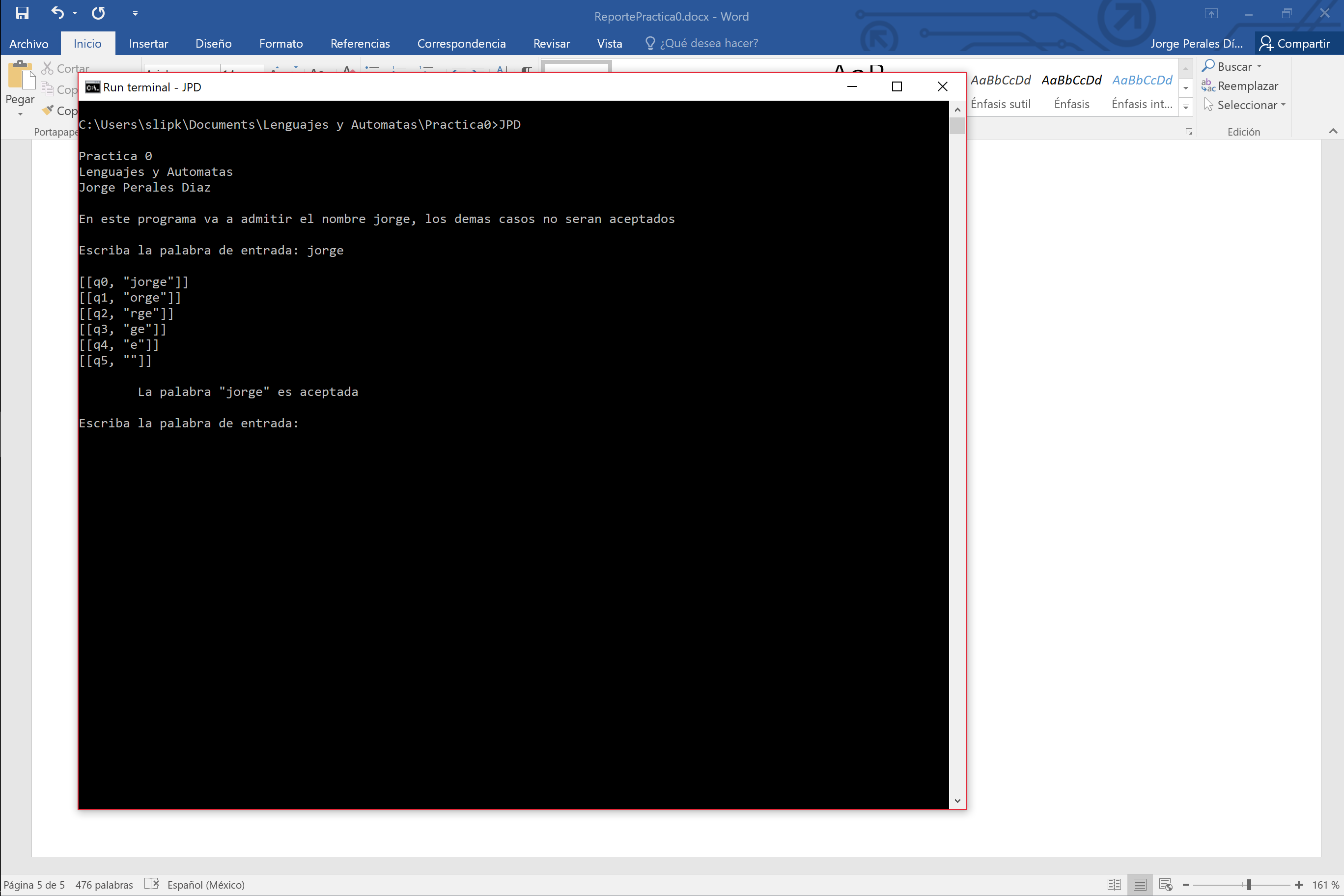
**Desarrollo**

Esta práctica se realizó en lenguaje C++, primero se importaron las librerías correspondientes para la ejecución del programa, luego se procedió a hacer el método principal en el cual se definió la quíntupla del AFD; los estados, el estado inicial, el estado final y la matriz de transiciones. Después se imprimió una leyenda con una breve descripción de la ejecución del programa, a continuación se pide la entrada de una palabra a evaluar, si esta no está dentro del alfabeto ya definido, el programa preguntara otra palabra, cuando se cumpla que la palabra ingresada este en el alfabeto entonces se procederá a realizar las transiciones necesarias, cuando se haya terminado este paso, se evaluara si el estado final corresponde con el ultimo estado de las transiciones, de ser así significa que la palabra es aceptada y se imprime en pantalla, caso contrario se imprimirá que la palabra no es aceptada dentro de nuestro autómata.

**Resultados**







**Conclusiones**

Como se vio en esta práctica una de las aplicaciones más importantes de los autómatas finitos deterministas, es el reconocimiento de palabras y lenguajes.

El proceso de reconocimiento es el siguiente:

1. El autómata comienza en el estado inicial cuando recibe la primera letra de la palabra y transita al edo. siguiente primera letra de la palabra y transita al estado siguiente.
2. A partir del edo. actual, se vuelve a procesar la siguiente letra para pasar de forma iterativa a los siguientes estados.
3. Al finalizar de procesar la palabra, si se llega a un estado final, la palabra es aceptada por el autómata. En caso contrario, se considera rechazada.

**Repositorio**

https://github.com/jpd21122012/LyAPractica0