

**INSTITUTO TECNOLOGICO DE IZTAPALAPA.**

**INGENIERIA EN SISTEMAS COMPUTACIONALES**

**ACTIVIDAD:**

PRACTICAS Y APUNTES.

**MATERIA:**

LEGUAJES Y AUTOMATAS I .

**PROFESOR:**

ING. ABIEL TOMAS PARRA HERNANDEZ.

**ALUMNO:**

REYES GOMEZ JAVIER ROBERTO 181080144.

**GRUPO:**

6BM.

**FECHA DE ENTREGA:**

CDMX A 27 DE JUNIO DEL 2021.

.

**SESION I.**

TEORIA DE LENGUAJES FORMALES

Sabemos que los alfabetos son un sistema finito y no vacío que cuyos elementos, que estos se denominan símbolos. Estos se designan normal mente con letras: ∑

Algunos ejemplos que vimos en clase fueron:

{0,1}

{a,b,c….,x,y,z}

{0,1,2,3,4,5,6,7,8,9}

{a,b}

Recordemos que el autómata finito aceptará una cadena en este caso será W, esta cadena comenzará desde el estado de inicio hasta un estado de aceptación.

En lo mencionado en clase la teoría de autónomas tiene cuatro niveles que se mencionaron:

* Sistema combinacional
* Autómata finito
* Autómata con pila
* Máquina de Turing

En base a la estructura de un traductor, observamos que tiene una sucesión de pasos en base a eso lo primero es generar un código intermedio, después optimizar el código intermedio y posterior mente generar código objeto (debemos gestionar los errores, tabla de símbolos y enfocarnos al objetivo del texto.

Las Faces de un compilador se basan en programas de computadora que traducen de un lenguaje a otro.

El compilador toma como su entrada un programa escrito en lenguaje fuente y produce un programa equivalente escrito en lenguaje objeto.  
Un compilador se compone internamente de varias etapas, o fases, que realizan operaciones lógicas.

**SESION II**

DEFINICION FORMAL DE UNA ER

Lo que se comentó en clase de la definición formal de expresión ER, es que es un equivalente algebraico para un autómata debido a que son patrones utilizados para determinar combinaciones de caracteres dentro de una cadena. La cual podemos expresar de dos maneras muy simples:

1. Representación literal ( var re = /ab + c/; ) patron cerrado entre diagonales.
2. Función constructora var re = new RegExp('ab + c') recopilación en tiempo de ejecución de la expresión.

DISEÑO DE ER.

El diseño de ER se basa wen un doiseño que se miro en clase , el ejemplo de ellos es;

Dado el alfabeto Σ = { a,b,c } , (a U b\*)a\*(bc)\* según lo comentado en clase esto es una expresión regular que esto va a representar al lenguaje .

({a} U {b}\*) {a}\*  {bc}\*

APLICACIONES EN PROBLEMAS REALES.

Las expresiones regulares facilitan la construcción de un compilador. A menudo se utiliza una expresión regular larga y compleja para validar la sintaxis de un programa, si el código del programa no concuerda con la expresión regular, sabremos que hay un error de sintaxis dentro del código.

**SESION III.**

**AUTOMATAS FINITOS**.

Lo que se comentó en clase es que los autómatas finitos es un modelo computacional que realiza cómputos en una forma automática sobre una entrada para así producir una salida. A si mismo se conforma por medio de un alfabeto, se podría decir por un conjunto de estados finitos, es como una función de transición, un estado inicial y también un conjunto de estados finales.

Estos autónomos deterministas se definen como una variable (a) estaría formado, también se define como una quíntupla que se forma mediante 5 elementos:

Q será el número de estados que se encuentran en el autómata.

∑ se refiere al alfabeto

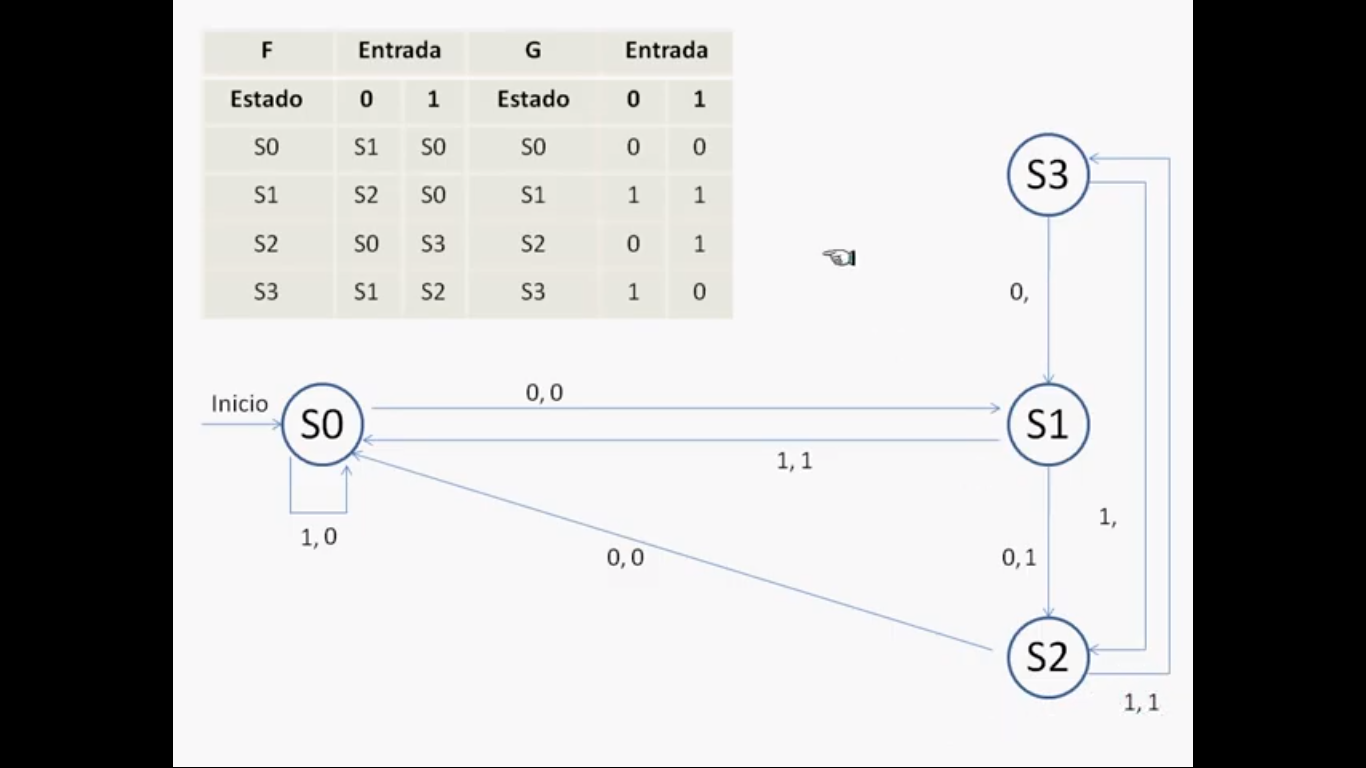
Ᵹ esta es la función de transición

q0 estado inicial

F seria conjunto de estados finales.

Deterministas: Cada combinación (estado, símbolo de entrada) produce un solo estado.

No Deterministas: Cada combinación (estado, símbolo de entrada) produce varios estados y además son posibles las transiciones con λ.

En clase también se mostró como se representa un ER a AFND, la tabla nos indica los resultados de los estados que obtenemos mediante el diagrama.

**SESION IV**

FUNCION ANALISIS LEXICO

A la conclusión que llegamos en clase es que un análisis léxico es una herramienta la cual nos ayuda a leer los caracteres de entrada de un código y así poder identificarlos, para poder realizar la función de leer la secuencia de caracteres del programa, carácter a carácter y así producir la salida compuesta de tokens.

COMPONENTES LEXICOS.

Un analizador léxico, inicialmente lee lexemas y le asigna un significado propio.

Componente Léxico: Secuencia lógica y coherente de caracteres relativos a una categoría (Identificador, palabras reservadas, espacios en blanco, paréntesis).

Patrón: Es la secuencia que genera la secuencia de caracteres que puede representar una expresión regular.

Lexema: Cadena de caracteres que concuerden con un patrón que describa el valor de la cadena.

Atributos: Proporciona información adicional sobre los tokens en sus atributos asociados.

CREACION DE TABLA DE TOKENS.

La tabla de Tokens es una serie de renglones cada uno de los cuales contiene una lista de valores de atributos que sean asociados con una variable en particular.

Símbolos de mayor importancia para poder proveer la información:

* Da un identificador.
* Que información es asociada con un nombre.
* Como se asocia es información con el nombre.
* Como acceder a la información

Funciones que realiza la tabla de tokens:

Validad que la semántica sea correcta.

Ayudar en la generación apropiada del código.

ERRORES LEXICOS.

Los errores léxicos los detectamos cuando el proceso de reconocimiento de componentes léxicos y la cadena de caracteres de la entrada no encajan con ningún patrón.

Algunos errores son:

* Nombres incorrectos de los identificadores: Se debe a que utilizamos caracteres inválidos (paréntesis, empezamos por un numero).
* Numero incorrectos: Los colocamos con caracteres inválidos (puntos en lugar de comas o no esta escrito correctamente).
* Palabras reservadas escritas incorrectamente: Problemas de ortografía.
* Caracteres que no pertenecen al lenguaje: @, ¿?

GENERADORES DE ANALIZADORES LEXICOS.

El generador léxico nos permite leer caracteres del archivo de entrada, donde encontraremos la cadena a analizar y poder reconocer las subcadenas que corresponden a los símbolos del lenguaje y retornar los tockens.

Existen tipos de generadores léxicos:

* Generador Lex: Este generador nos ayuda a generar análisis sintáctico, genera código fuente en c, a partir de una serie de especificaciones escritas en el lenguaje.
* Generador Flex: Este generador nos brinda la realización para patrones lexicos en texto.
* Generador JTLex: Este generador nos brinda expresar sintaxis y semántica en los esquemas de traducción.

**SESION V.**

ANÁLISIS SINTÁCTICO

Mediante el transcurso de la clase vivos gramática la cual tipo 3 son regulares de acuerdo al formato de producción puede ser:

Lineales a la derecha en ejemplo de ello que visualizamos en la sesión sincrónica

A aB

A a

Y las lineales a la izquierda

A Ba

A a

El tipo 2: se refiere al libre contexto como, por ejemplo:

.A B es siempre indispensable del contexto.

La gramática libre de contexto es formal en la que cada una de las reglas de producción es de la forma V W donde se puede decir que V será el símbolo no terminal y W será siempre una cadena de terminales y/o no terminales. Con esto nos referimos que al hecho de que el no terminal V puede siempre ser sustituto por W sin haber tenido en cuenta el contexto en el que ocurra.

Recordemos que los arboles de derivación es un lenguaje que se genera por una gramatica libre de contexto se representa gráficamente mediante un árbol con raíz ordenado, a eso se le conoció como árbol de derivación. Damos a entender que la raíz del árbol presenta un símbolo inicial. Los nodos internos del árbol representan los símbolos no terminales, que parecen en la derivación. Las hojas de los arboles esas representan los símbolos terminales.

En las formas normales de Chomsky: remplazaremos toda la producción de una forma como la que se muestra a continuación:

A C1,C2…..C3 por una cadena de producciones

A C1,V2 V1A C2V2….. Vn-2 Cn-1 Cn

Donde los V1…..Vn-2 son nuevas variables, intermedias.

Lo visto del diagrama de sintaxis es otra forma al igual que los arboles de derivación de poder especificar gramáticas de cualquier tipo en especial de tipo 2. Este esquema permite visualizar las derivaciones al instante de que ocurren. Es una forma visual de poder representar la gramática de un lenguaje.

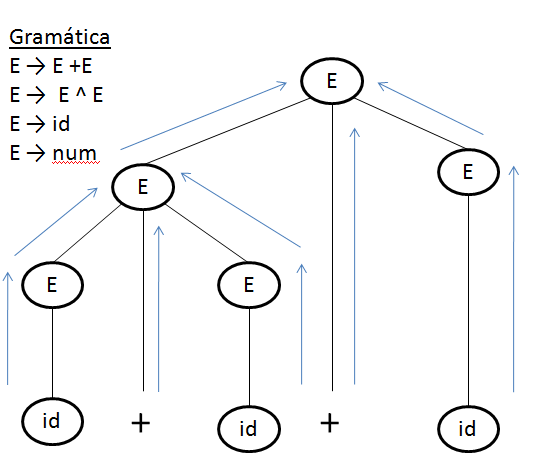
También vimos la ambigüedad que nos indica la eliminación de ella, para las construcciones de los lenguajes de programación comunes, en la mayoría de las gramáticas, damos entender que no se respeta la precedencia de operadores, también esa secuencia puede agruparse desde la izquierda y desde la derecha. Lo convencional es agrupar desde la derecha.

En el Análisis Sintáctico Ascendente se parte de las hojas y se intenta construir el árbol hacia arriba hasta llegar al símbolo inicial de la gramática.

En un análisis top-down un parser hacer corresponder cadenas de entrada con sus correspondientes derivaciones izquierdas.

     En un análisis bottom-up un parser hace corresponder cadenas de entrada con las inversas de las correspondientes derivaciones derechas.

EJEMPLO:



La generación de matriz predictiva nos menciona que los First y Follows son significados para verificar y construir pares predictivos, son un conjunto de tokens que se encuentran en el tope de la pila.

Pseudocódigo y algoritmo para calcular los First de una gramática:

1.-Si X es un terminal FIRST (X) = {X}2.-Si X -> e es una producción e (pertenece) FIRST(X)

3.-Si X -> Y1Y2....Y n es una producción.

a . FIRST (Y1) (es subconjunto de) FIRST(X)

b. Si e (pertenece) FIRST (YK) (para todo) k< iFIRST(Yi)(es subconjunto de) FIRST(X)

c.Si e (pertenece)

FIRST(YK) (para todo) I≤n e (pertenece a) FIRST(X) Nota (a,b y c pertenecen al paso 3 y solo se toma uno dependiendo el caso).

Los errores sistemáticos se suelen producir cuando la sintaxis del código es correcta, pero la semántica o significado no es el que se pretendía.

La construcción obedece las reglas del lenguaje, y por ello el compilador o intérprete no detectan los errores semánticos.

Los compiladores e intérpretes sólo se ocupan de la estructura del código que se escribe,

y no de su significado. Un error semántico puede hacer que el programa termine de forma anormal, con o sin un mensaje de error.

**SESION VI**

DEFINICION FORMAL MT.

En clase se determinó que en 1936 se definió la máquina de Turing por Alan Turing, la cual es un modelo matemático (computacional) el cual nos permite interpretar problemas matemáticos a través de un algoritmo, debido a que manipula símbolos mediante una tira de cinta con reglas (Tabla), debido a que es un dispositivo hipotético el cual representa una maquina de computación.

CONSTRUCCION MODULAR DE UNA MT.

La construcción modular tiene como finalidad desarrollar maquinas complejas a partir de bloques elementales

(a partir de máquinas pequeñas), para la creación de máquinas de Turing se llevan a cabo diagramas de transición para poder combinarlos de manera parecida y así realizar la formación de la unión y concatenación de los autómatas finitos

LENGUAJES ACEPTADOS POR LA MT.

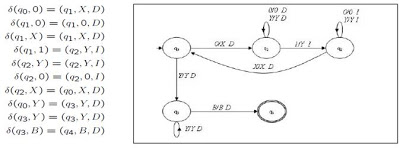
La máquina de Turing puede considerarse como un [autómata](https://es.wikipedia.org/wiki/Teor%C3%ADa_de_aut%C3%B3matas) capaz de reconocer [lenguajes formales](https://es.wikipedia.org/wiki/Lenguaje_formal), es capaz de reconocer los lenguajes recursivamente enumerarles ( lenguajes más generales, o tipo cero (0) ), de acuerdo a la [jerarquía de Chomsky](https://es.wikipedia.org/wiki/Jerarqu%C3%ADa_de_Chomsky) .

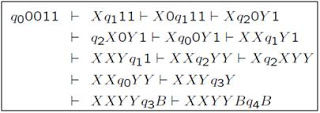
Su potencia es, , superior a otros tipos de autómatas, como el [autómata finito](https://es.wikipedia.org/wiki/Aut%C3%B3mata_finito), o el [autómata con pila](https://es.wikipedia.org/wiki/Aut%C3%B3mata_con_pila), o igual a otros modelos con la misma potencia computacional.

El siguiente ejemplo se visualizó en una de las sesiones:

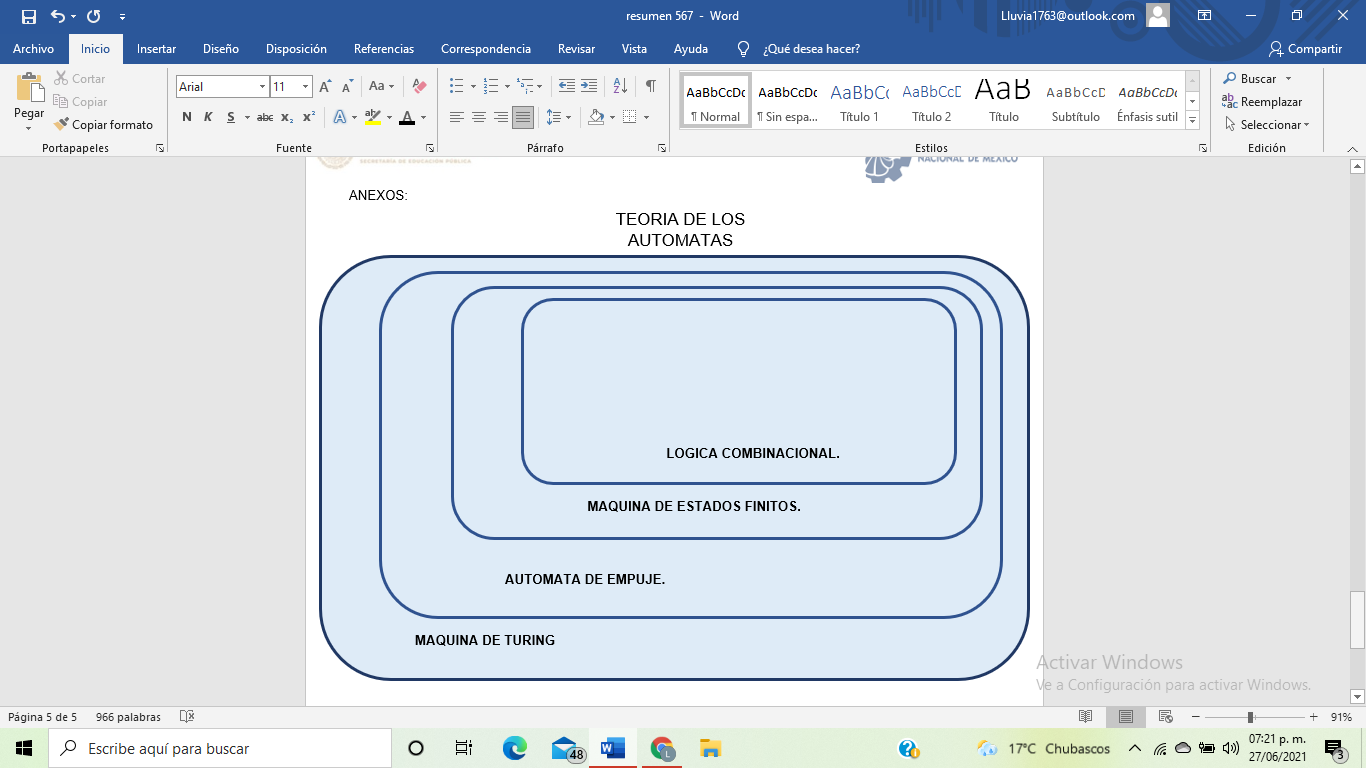
Una cadena ω∈A^\*, es aceptada por una MT, si comienza en el estado e0, con la cabeza de lectura/escritura en el símbolo más a la izquierda, luego de leer toda la cadena ω, llega a un estado e\_f∈F.

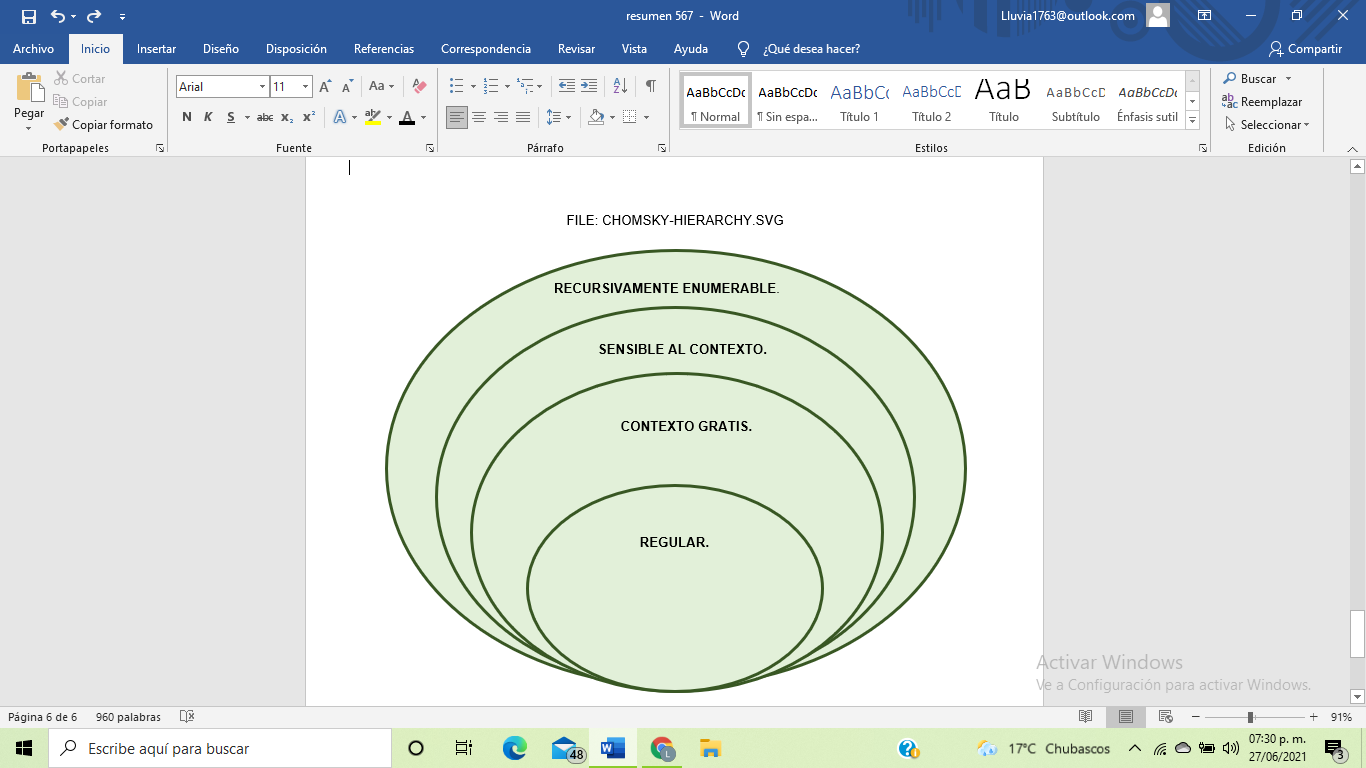
El lenguaje aceptado por MT, es el conjunto de todas las cadenas que son aceptadas por MT:  
L(MT)={ω / e\_0 ω ⊢\*α\_1 e\_f α\_2 y e\_f∈F y α\_1,α\_1 ∈C^\* y ω∈A^\* }  
Tenemos por ejemplo una MT que reconoce el lenguaje {0^n 1^n:n≥1}. Las transiciones de la máquina se representan como sigue:

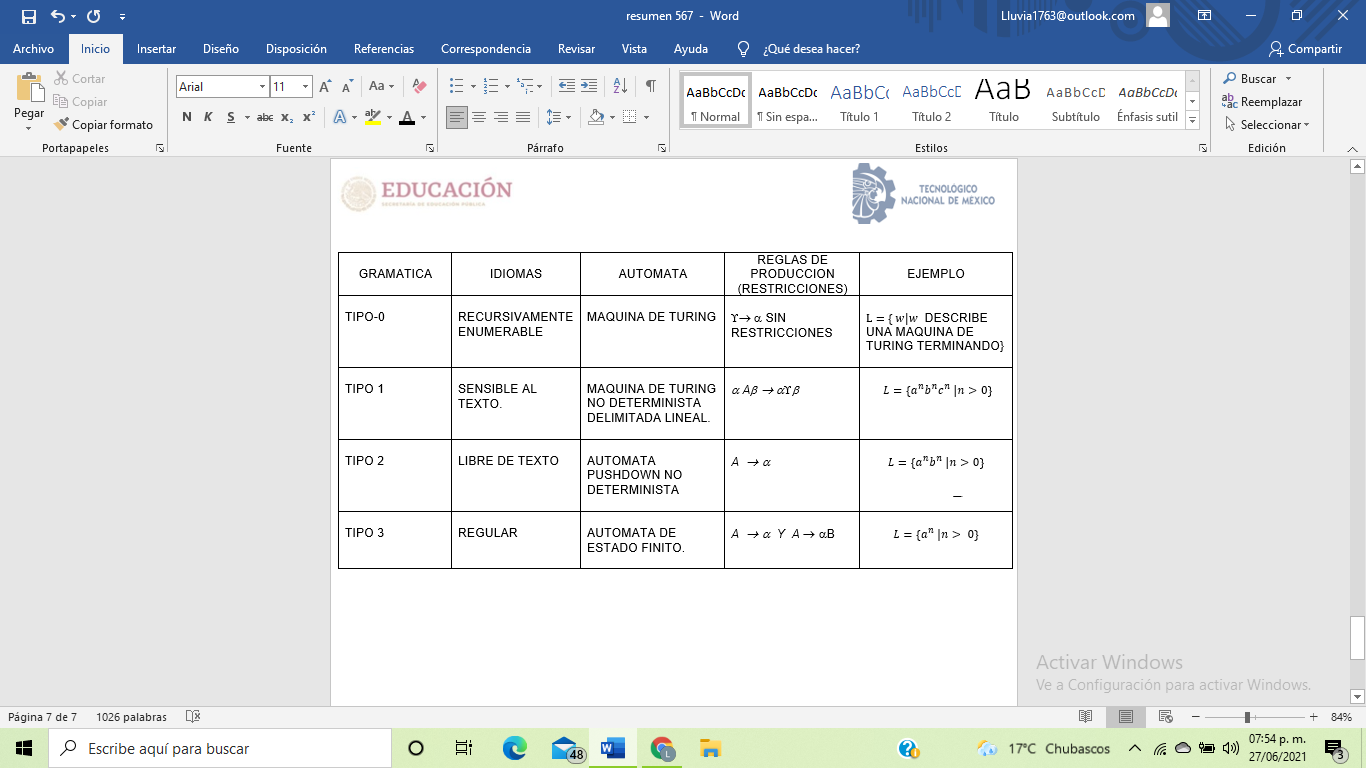
[](http://2.bp.blogspot.com/_iM4O7Io_Qd0/TFzqDeo6AVI/AAAAAAAAABs/qpnpufYMue0/s1600/6.1+Ejemplo+Lenguaje.JPG)  
Se evalúa la cadena w = 1100, arrojando el siguiente resultado:

[](http://3.bp.blogspot.com/_iM4O7Io_Qd0/TFzqvljmOhI/AAAAAAAAAB0/SJFcBluekXw/s1600/6.2+Ejemplo+Lenguaje.JPG)

Otras cadenas también aceptadas por esta MT son 11110000, 10, 11111110000000.

ANEXOS:





INSTITUTO TECNOLÓGICO DE IZTAPALAPA 1

LENGUAJES Y AUTOMATAS 1

**RESUMEN. DESENREDANDO LA HISTORIA DE ADA LOVELACE.**

INTEGRANTES:

ANGELICA BEATRIZ CAMACHO CAPULÍN 181080421

OSVALDO RUIZ BARCO 181080148

JAVIER ROBERTO REYES GÓMEZ 181080144

AVILEZ ARELLANO RAFAEL IVAN 181080381

ISC-6BM

**RESUMEN. DESENREDANDO LA HISTORIA DE ADA LOVELACE.**

[Ada Lovelace](https://www.wolframalpha.com/input/?i=ada+lovelace) nació hace 200 años hoy. Para algunos, es una gran heroína en la historia de la informática, nació en Londres el 10 de diciembre de 1815 de padres de la alta sociedad.  Ada aprendió historia, literatura, idiomas, geografía, música, química, costura, taquigrafía y matemáticas (enseñadas en parte a través de métodos experimentales) hasta el nivel de geometría elemental y álgebra. Cuando Ada tenía 11 años, fue con su madre y un séquito en una gira de un año por Europa. Cuando regresó, estaba haciendo cosas con entusiasmo, cómo estudiar lo que ella llamaba “flyología”, e imaginar cómo imitar el vuelo de las aves con máquinas de vapor. Su padre era un orfebre y banquero emprendedor y exitoso  fue a Cambridge para estudiar matemáticas, pero pronto se propuso modernizar la forma en que se realizaban allí, fundó la Sociedad Analítica para impulsar reformas cómo reemplazar la notación basada en puntos de Newton para el cálculo con la notación basada en funciones de Leibniz .A principios de 1822, Babbage, de 30 años, estaba ocupado estudiando diferentes tipos de maquinaria y produciendo planos y prototipos de lo que podría ser el Motor Diferencial. El encuentro de Ada con el motor de diferencias parece ser lo que encendió su interés por las matemáticas. Ella escribió: "Creo que poseo una combinación sumamente singular de cualidades que encajan exactamente para hacerme preeminentemente un descubridor de las realidades ocultas de la naturaleza". Ada había tenido problemas de salud intermitentes durante años, pero en 1841 aparentemente empeoraron y comenzó a tomar opiáceos de forma sistemática. Tenía muchas ganas de sobresalir en algo y comenzó a tener la idea de que tal vez debería ser música y literatura en lugar de matemáticas. Pero su esposo William parece haberla convencido de que no lo hiciera y, a fines de 1842, volvió a estudiar matemáticas.

Había estado haciendo todo tipo de cosas, con distintos grados de éxito, después algunos intentos fueron nombrado profesor LUCASIANO DE MATEMÁTICAS en Cambridge, escribiendo un libro bastante influyente, “sobre la economía de la maquinaria y las manufacturas”, trataba de dividir las tareas en fábricas (un tema surgido en relación al cálculo humano en operaciones matemáticas). En 1837 intervino con un tema popular de la teología natural, agregando su noveno tratado de bridgewater. La pregunta central era si hay evidencia de una deidad a partir del diseño aparente visto en la naturaleza, su libro Babbage es difícil de leer,: "Las nociones que adquirimos sobre la invención y el diseño surgen al comparar nuestras observaciones sobre las obras de otros seres con las intenciones de las que somos conscientes en nuestras propias empresas”, alrededor de 150 años después, habla de la relación entre procesos mecánicos, las leyes naturales y el libre albedrío, realizando afirmaciones como "los cálculos de gran complejidad se pueden realizar por medios mecánicos", pero afirma con ejemplos que son bastante débiles que un motor mecánico puede producir secuencias de números que muestran cambios inesperados podría ocurrir con milagros. Después de tiempo Babbage probó suerte en la política, postulándose para el parlamento dos veces en una plataforma orientada a la fabricación, pero no pudo ser elegido debido a denuncias de uso indebido de fondos gubernamentales en Difference Engine, Babbage celebraba con fiestas de lujo en su casa grande y cada vez más desorganizada de Londres, atrayendo a luminarias como Charles Dickens, Charles Darwin, Florence Nightingale, Michael Faraday y el Duque de Wellington. a pesar de los grados y honores que enumeró su nombre tenía 6 líneas, estaba cada vez más amargado por su falta de reconocimiento. Babbage había contratado a un ingeniero de su época para que construyera el motor, pero en una década de trabajo, a pesar del desarrollo de máquinas, herramienta de precisión, el motor real no estaba terminado, poco después de conocer a Ada, Babbage había intentado frenar el proyecto, pero obtuvo como resultado el renunció del ingeniero a cargo e insistió en que podía conservar todos los planos de la máquina diferencial, incluso los propios de Babbage, dio un cambio por completo y decidió hacer una máquina que sólo calculara las diferencias, imaginó un "motor analítico". En 1826 había inventado la notación mecánica, que tenía la intención de proporcionar la representación simbólica del funcionamiento de la maquinaria, pero tuvo una decepción ya que la gente no apreciaba su invento. Después de un tiempo en Inglaterra, las ideas de Babbage adquirieron cierta popularidad en otros lugares, y en 1840 fue invitado a dar una conferencia sobre la Máquina Analítica en Turín, y recibió honores del gobierno italiano, nunca había publicado una descripción seria del motor de diferencias ni del motor analítico, pero se decidió por fin hablar sobre la máquina analítica en Turín. En octubre de 1842, Menabrea publicó un artículo en francés basado en sus notas. Cuando Ada vio el periódico, decidió traducirlo al inglés y enviarlo a una publicación británica, después de años Babbage afirmó que le sugirió a Ada que escribiera su propio relato de la máquina analítica, ella había respondido que no se le había ocurrido esa idea, en febrero de 1843, Ada había resuelto hacer la traducción, pero agregó extensas notas propias a finales de 1843 de julio Ada había terminado de escribir sus notas estaba orgullosa de ella pero para Babbage los elogiaba quería agregar prefacio anónimo explicando que el gobierno británico no lo había apoyado con el proyecto Ada se negó a ayudarlo Babbage insistía bastante tanto que llego a pensar que podrían retirar la nota de la publicación. Ada estaba entusiasmada con lo que había logrado lo compartió con familia y amigos su triunfo explicándoles que nadie puede estimar el problema y el trabajo interminable de tener que revisar la impresión de fórmulas matemáticas. En cuestión de días ya había chismes de la sociedad sobre la publicación de Ada, no le importaban las críticas y los chismes ya que ella estaba dedicada a convertirlo en algo real y no en un secreto, aunque no le importaba que fuera algo exagerado, sino que fuera sobrevalorado. Babbage le expresa su admiración por ella describiendola como una hechicera de los números. Al concluir sus notas decidió dedicarse por completo a su hogar y a su familia, sin embargo, tuvo problemas de salud cáncer de cuello uterino los dolores no los toleraba ya, pasaron unos meses y a Ada se le complicaba más sus dolores, durante su ausencia llegó a sentirse cerca de la muerte finalmente terminaron sus sufrimientos a su edad de 36 años el día 27/Nov/1852. El funeral de Ada fue pequeño su madre , ni Babbage asistieron ,a los 41 años William finalmente se volvió a casar su hijo mayor con quien tuvo muchas dificultades Ada se unió a la Marina varios años antes de que ella muriera , falleció a sus 26 años trabajando en un astillero en Inglaterra la hija de Ada se casó con un poeta un tanto salvaje pasó años en el Medio Oriente y se convirtió en el principal creador de caballos árabes ,el hijo menor de edad es el heredero titular de la familia y pasó la mayor parte en la propiedad, su madre de Ada murió en 1960 todos hablaban de ella pareciendo que estaban en contra de ella . Babbage vivió 18 años más después de la muerte de Ana muriendo en 1871 intento trabajar en el motor analítico de nuevo en 1856 pero no hizo grandes progresos escribiendo artículos sobre las estadísticas de la casa de luz en 1864 publicó su autobiografía pasajes de la vida de un filósofo un documento extraño bastante amargo el capítulo sobre el mentor analítico se abre como una cita de un poema de Byron nunca se volvió a casar después de la muerte de su esposa y sus últimos años parecen haber sido solitarios. aparentemente le gustaba decir que con mucho gusto renunciaría al resto de su vida si pudiera pasar 3 días 500 años en el futuro, su cerebro se conservó y todavía se muestra en exhibición, aunque nunca terminó Babbage su difference Engine, una empresa sueca lo hizo y pasaron sus proyectos a su hijo.

Las computadoras mecánicas continuaron desarrollándose, dando paso gradualmente para la programación que comenzó a entenderse donde en la década de 1940, se redescubrió el trabajo de Babbage y Ada's Notes y fue un tal Bertram Bowden, quien “redescubrió” a Ada. Al investigar su libro de 1953 Más rápido que el pensamiento.El conocimiento acerca de Ada Lovelace y Charles Babbage comenzó a aumentar, y comenzaron a aparecer biografías, aunque a veces con errores espeluznantes “problema de los tres cuerpos” en una carta de Babbage se refiere al problema de los tres cuerpos en la mecánica celeste.A medida que aumentaba el interés por Babbage y Ada, también lo hacía la curiosidad por saber si el motor de diferencia realmente habría funcionado por esa razón se montó un proyecto y en 2002, después de un esfuerzo heroico, se construyó un Motor Diferencial completo, con solo una corrección en los planos. Sorprendentemente, la máquina funcionó.

El contenido real de sus notas sobre el motor analítico explica que el motor de diferencia puede calcular valores de cualquier polinomio de sexto grado, pero el motor analítico es diferente porque puede realizar cualquier secuencia de operaciones. Ada analizo cómo funcionaría una secuencia de tipos específicos de cálculos en el motor analítico, con "Tarjetas de operación" que definen las ubicaciones de los valores. Ada habla de "ciclos", ahora conocidos como bucles y bucles anidados, dándoles una notación matemática. La idea de usar bucles para reducir la cantidad de tarjetas necesarias y el valor de reorganizar las operaciones para optimizar su ejecución en el motor analítico. Ada parece haber entendido con cierta claridad la visión tradicional de la programación: que diseñamos programas para hacer cosas que sabemos hacer. Lo que se ha convertido en la parte más famosa de lo que escribió Ada es el cálculo de los números de Bernoulli, en la Nota G.

Donde Ada tenía que especificar cómo calcular realmente esto en el motor analítico. El algoritmo que utilizó para el cálculo fue (en términos modernos):

En el motor analítico, la idea era tener una secuencia de operaciones (especificadas por “Tarjetas de operación”) realizadas por el “Molino”, con operandos provenientes de la “Tienda” (con direcciones especificadas por “Tarjetas variables”). Para calcular los números de Bernoulli de la forma en que Ada quería, se necesitan dos ciclos de operaciones anidados.

Los números de Bernoulli hace unos años, básicamente estábamos usando el mismo algoritmo que Ada, aunque ahora hay algoritmos un poco más rápidos que calculan efectivamente los numeradores de números de Bernoulli. El logro de Ada fue extraer de estos detalles una exposición clara del funcionamiento abstracto de la máquina, algo que Babbage nunca hizo. Es una forma bastante agradable de representar cómo funciona un sistema, similar en algunos aspectos a un diagrama de tiempos moderno, pero no exactamente igual. Y a lo largo de los años que Babbage trabajó en el motor analítico, sus notas. La "Notación Mecánica" es el volante que había impreso para la Gran Exposición de 1851, aparentemente una propuesta para la estandarización en dibujos de componentes mecánicos.

Charles Babbage era un hombre enérgico que tenía muchas ideas, algunas buenas. Ada Lovelace era una mujer inteligente que se hizo amiga de Babbage. Como un favor a Babbage, escribió una exposición de la máquina analítica y, al hacerlo, desarrolló una comprensión más abstracta de la misma que Babbage, y vislumbró la increíblemente poderosa idea de la computación universal. Y es esta idea de computación universal la que, por ejemplo, hace posible el software y la que lanzó toda la revolución informática en el siglo XX.

Gottfried Leibniz había ya tenía un concepto filosófico de algo así como volver cómputo universal en la década de 1600. Y el motor analítico de Babbage es el primer ejemplo explícito que conocemos de una máquina que habría sido capaz de realizar cálculos universales. Sin embargo, Babbage no lo pensó en estos términos. Pero en el esfuerzo por diseñar esto, terminó con una computadora universal.

Otra forma importante en que las computadoras modernas reducen el hardware es manejando números en binario en lugar de decimal.

Leibniz sabía de binario. Y si George Boole hubiera seguido su reunión con Babbage en la Gran Exposición, tal vez eso hubiera llevado a algo. La concepción principal de Babbage del motor analítico era como una máquina para producir automáticamente tablas matemáticas, ya sea imprimiéndolas por composición tipográfica o dándolas como tramas dibujándolas en una plancha. Imaginó que los humanos serían los principales usuarios de estas tablas, aunque pensó en la idea de tener bibliotecas de tarjetas precalculadas que proporcionarían versiones legibles por máquina.

Pero en la época de Babbage, con la idea de un motor analítico masivo, esta forma de hacer las cosas habría sido impensable. Entonces, ¿qué pasa con Ada? Por todo lo que puedo decir, ella hablaba con claridad y pensaba con claridad. Ella venía de las clases altas, pero no usaba ropa especialmente a la moda y se comportaba mucho menos como una condesa estereotipada que como una intelectual. De adulta, era emocionalmente bastante madura, probablemente más que Babbage, y parece haber tenido una buena comprensión práctica de las personas y el mundo.

Al igual que Babbage, era rica de forma independiente y no tenía necesidad de trabajar para ganarse la vida. Pero era ambiciosa y quería hacer algo por sí misma. Lo que dice suele ir vestido con agradables detalles victorianos, pero en el fondo, las ideas son claras y, a menudo, bastante contundentes.



**INSTITUTO TECNOLOGICO DE IZTAPALAPA.**

**INGENIERIA EN SISTEMAS COMPUTACIONALES**

**INFORME TECNICO DE RESIDENCIA**

**PROFESINAL.**

**TEMA:**

AUTOMATA CELULAR.

**LUGAR DE REALIZACION:**

INSTITUTO TECNOLOGICO DE IZTAPALAPA.

**PROFESOR:**

ING. ABIEL TOMAS PARRA HERNANDEZ.

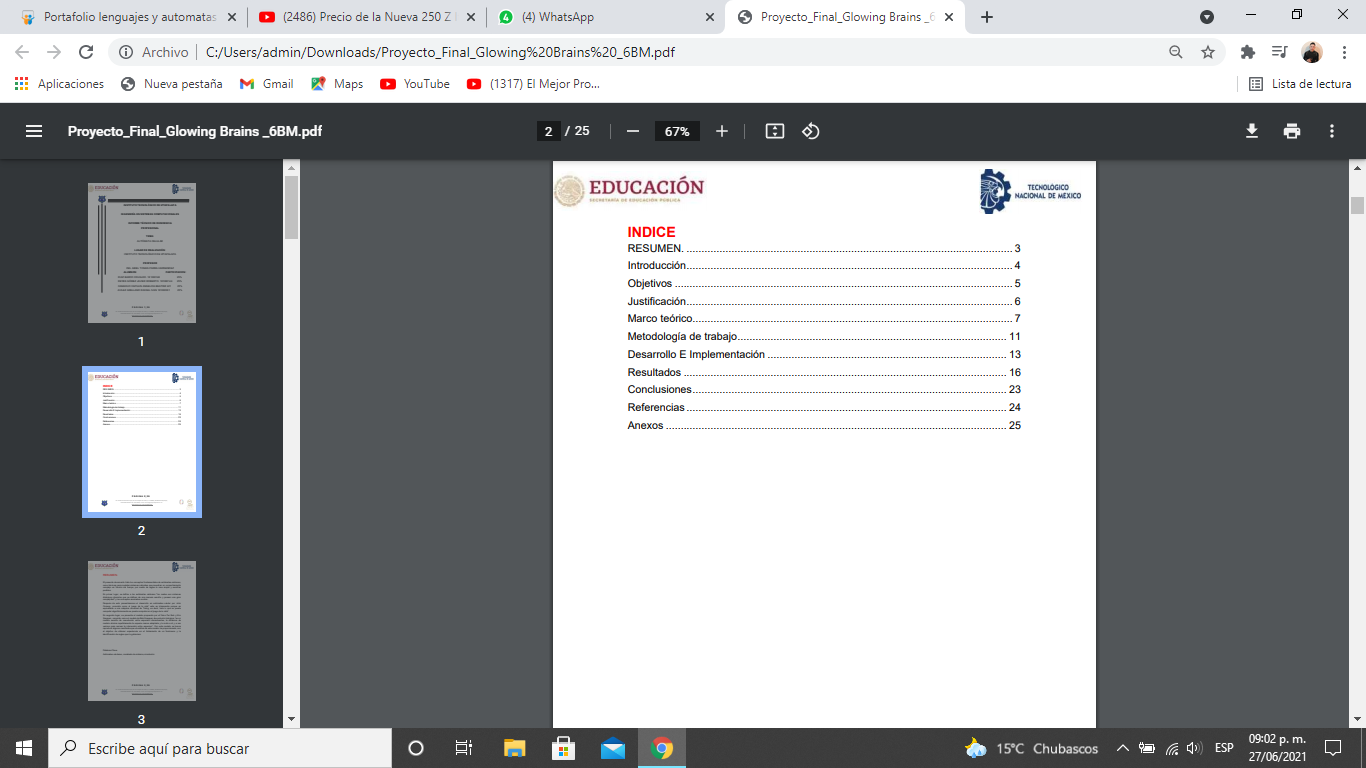
ALUMNOS: PARTICIPACION:

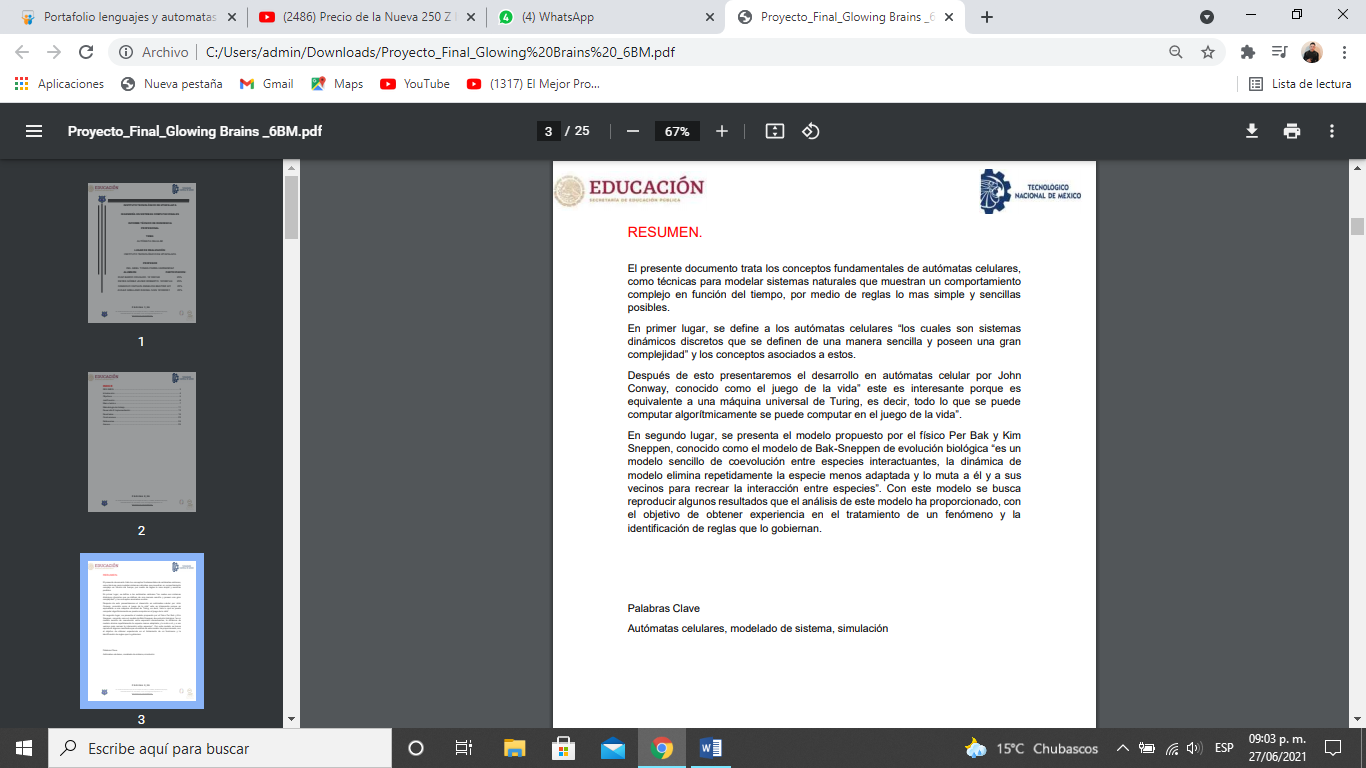
RUIZ BARCO OSVALDO. 181080148 25%

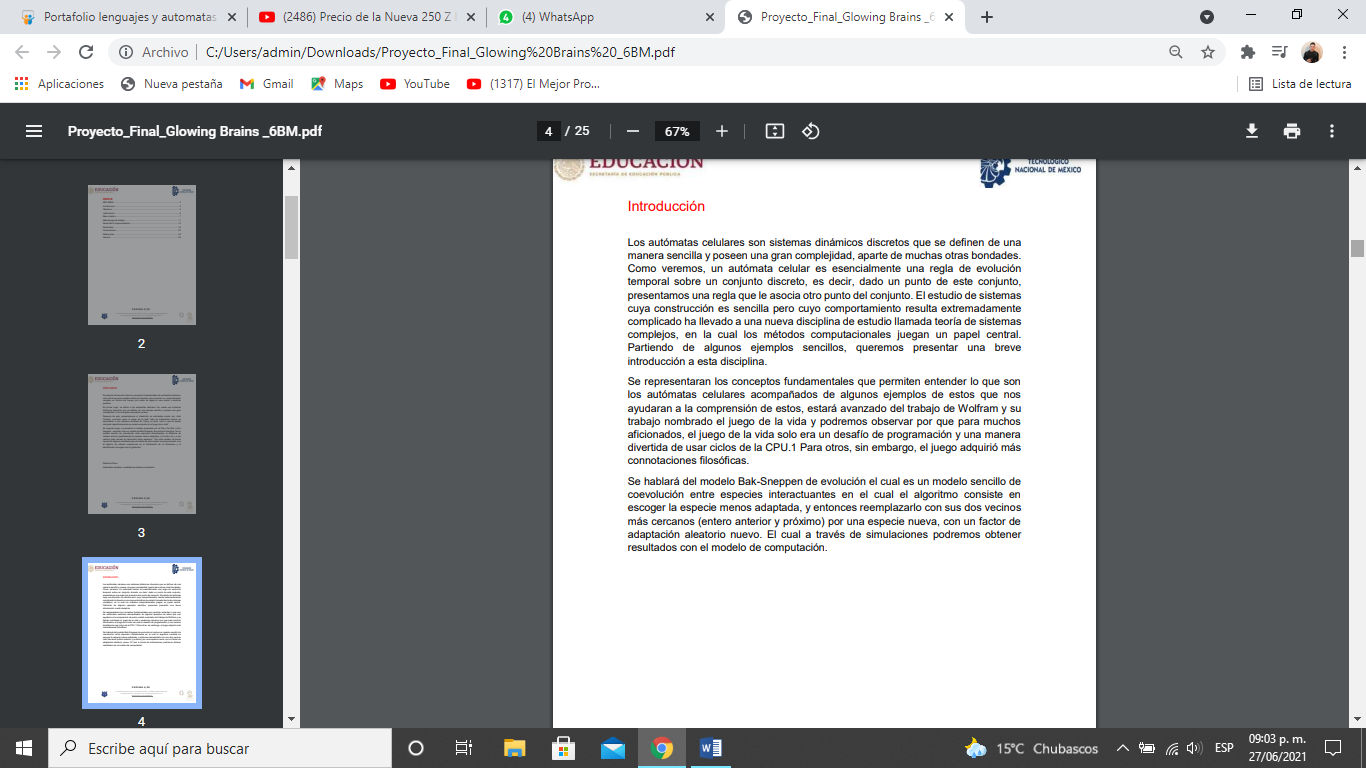
REYES GÓMEZ JAVIER ROBERTO. 181080144 25%

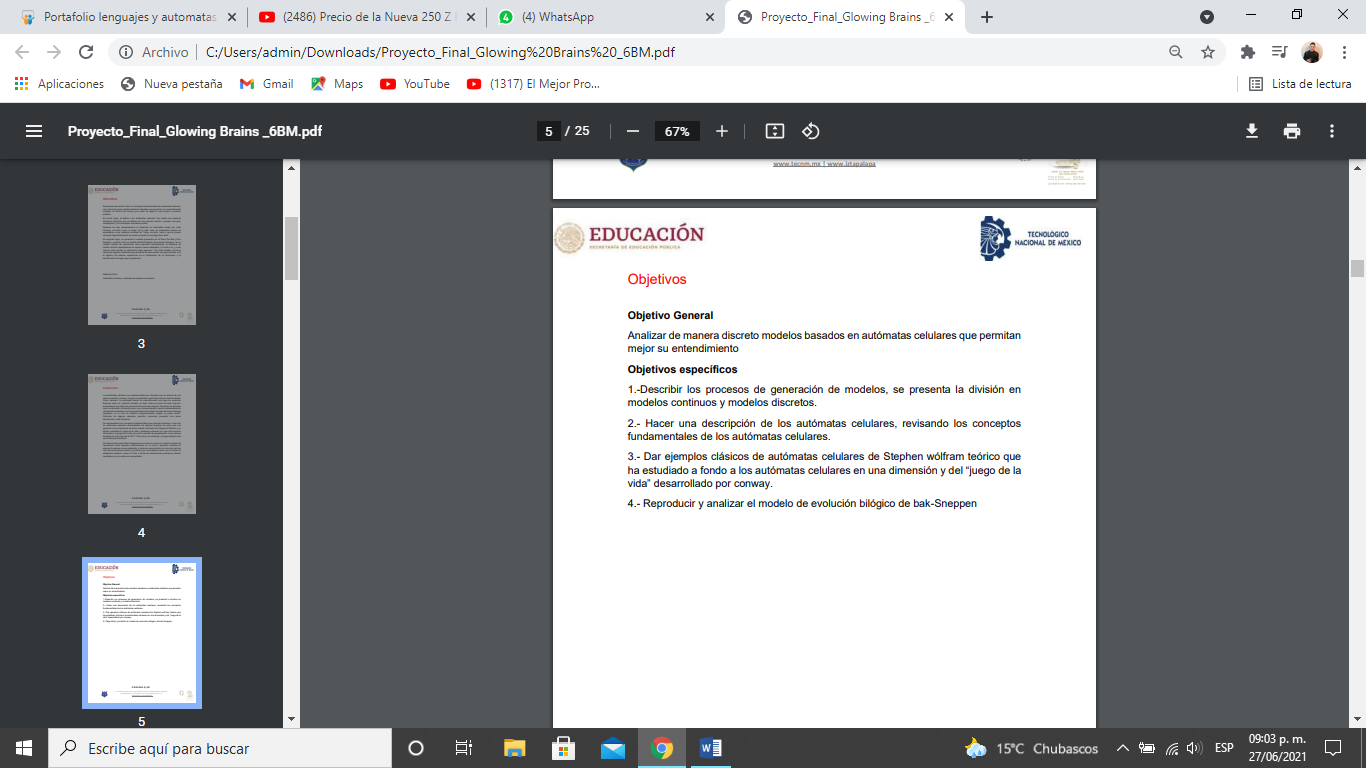
CAMACHO CAPULIN ANGELICA BEATRIZ 181080421 25%

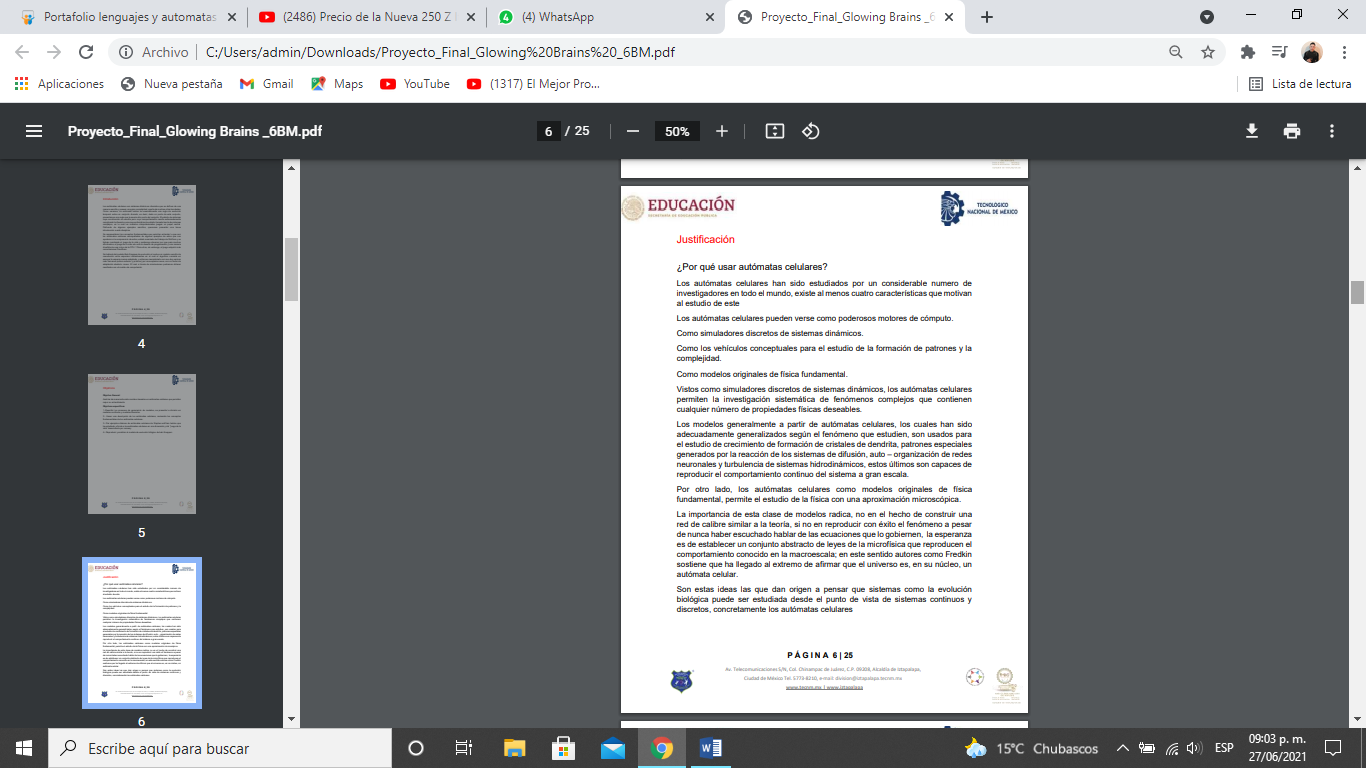
AVILEZ ARELLANO RAFAEL IVAN 181080381 25%

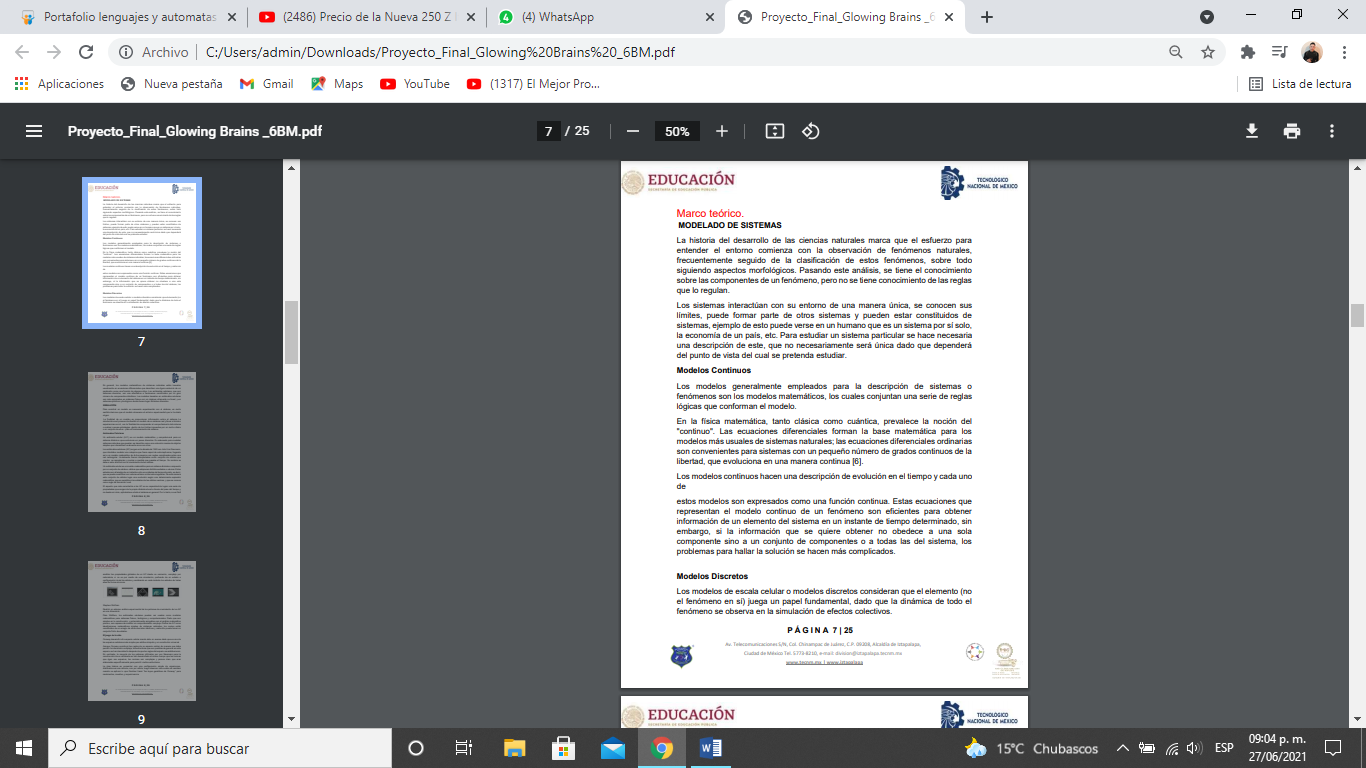


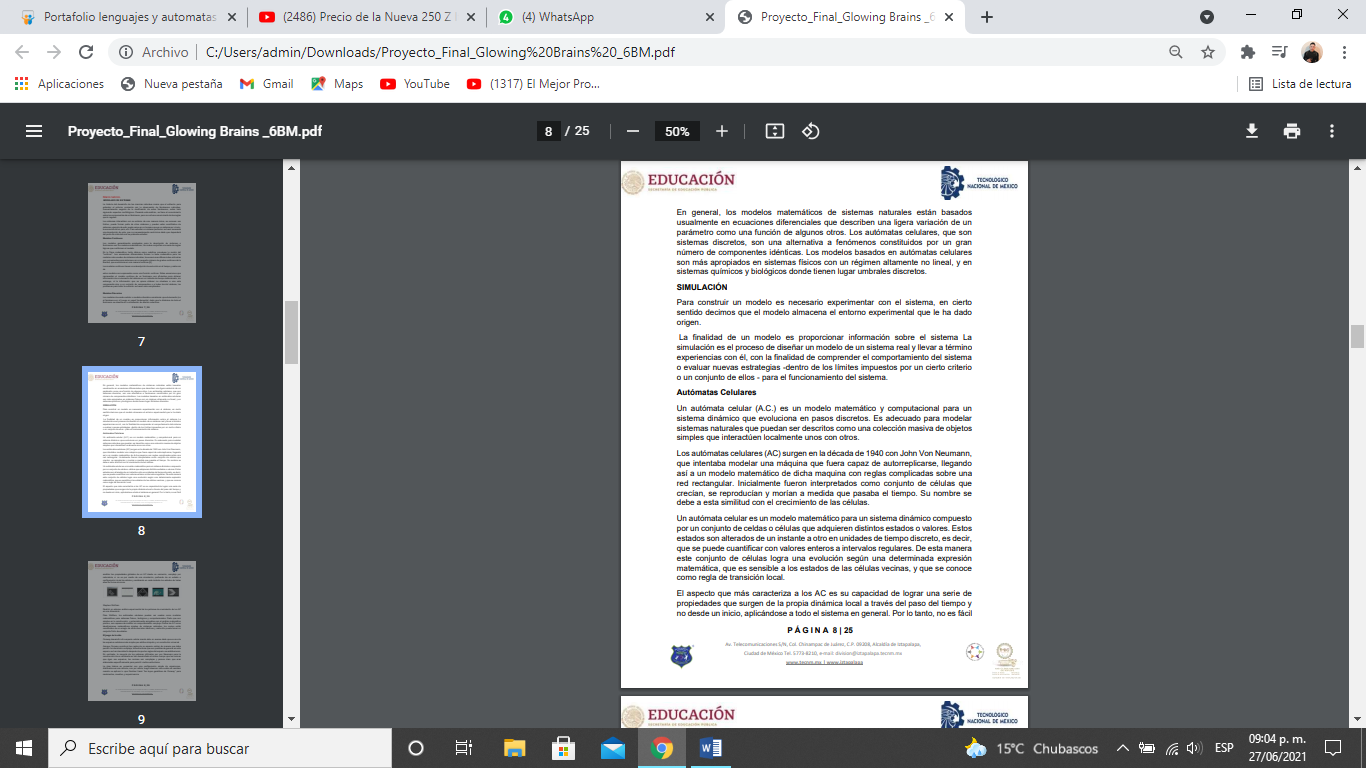


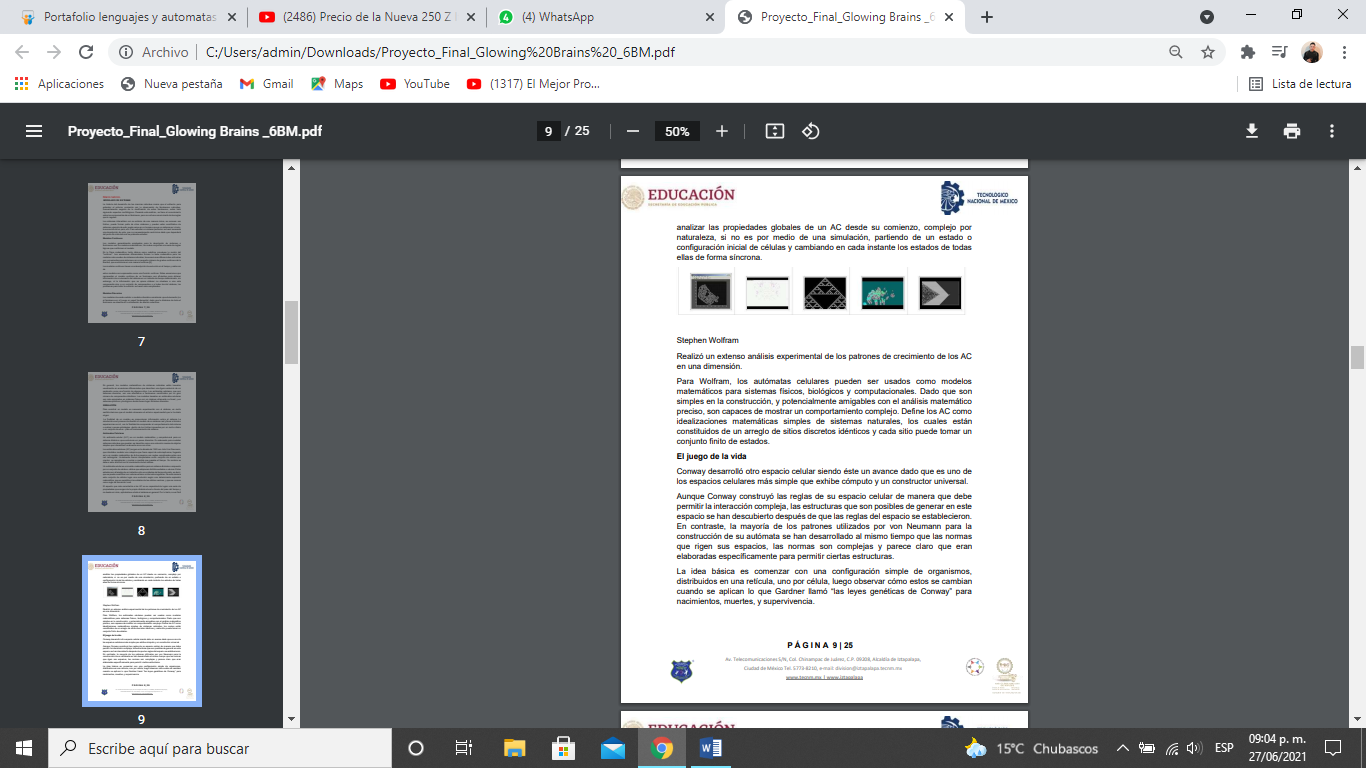


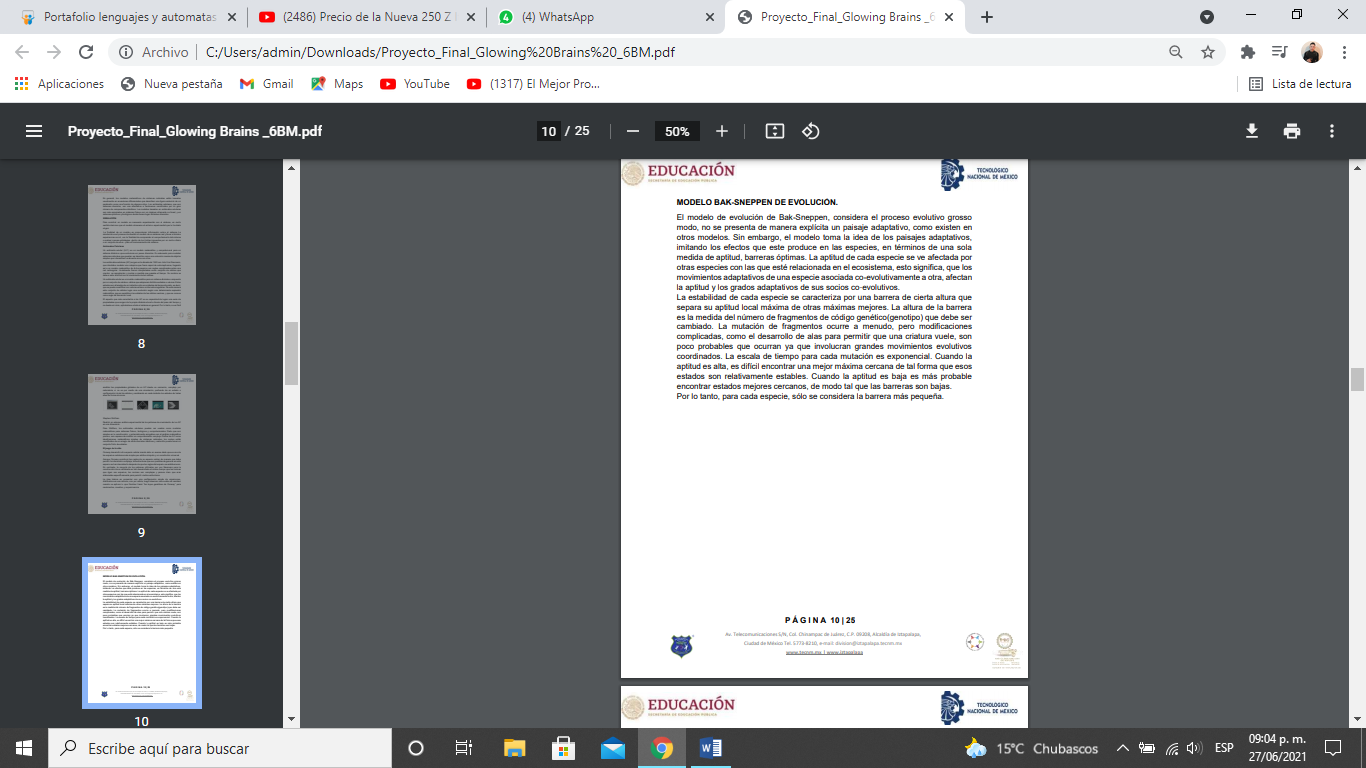




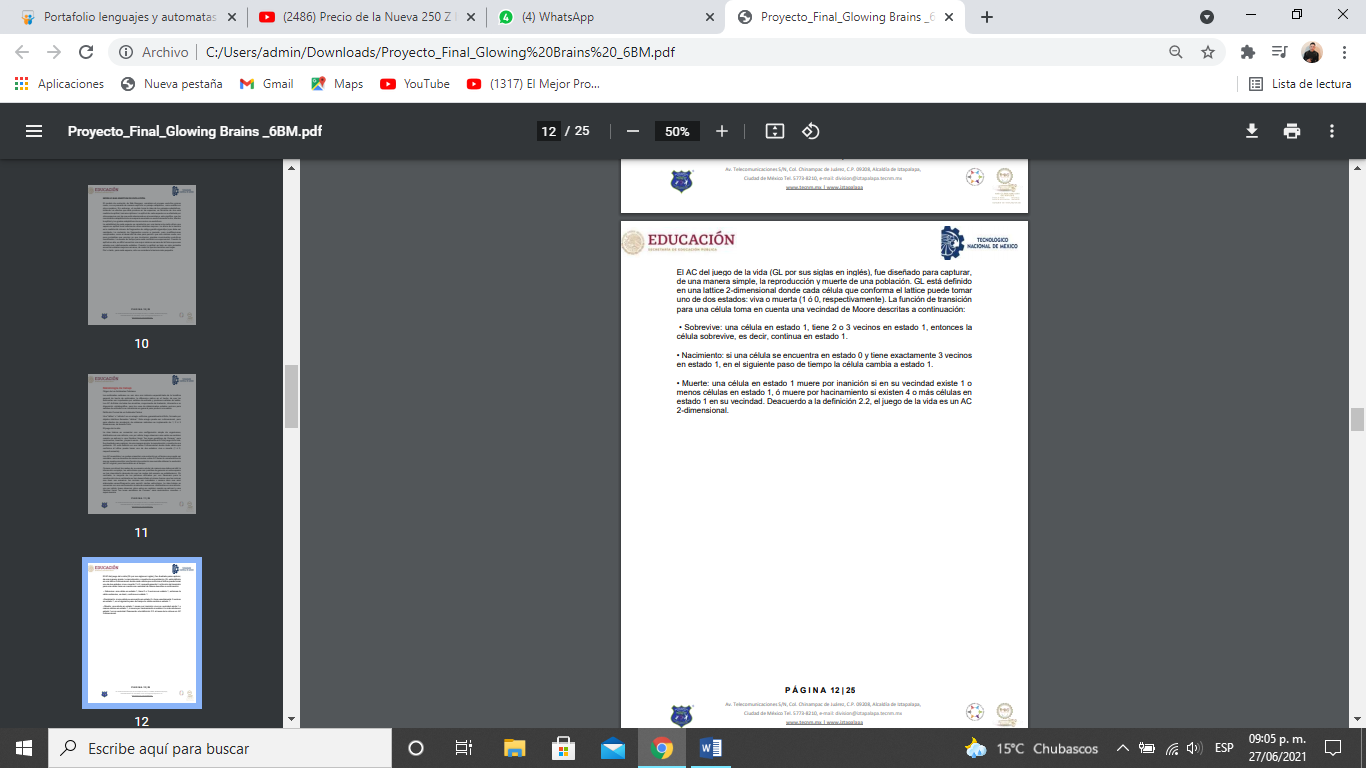


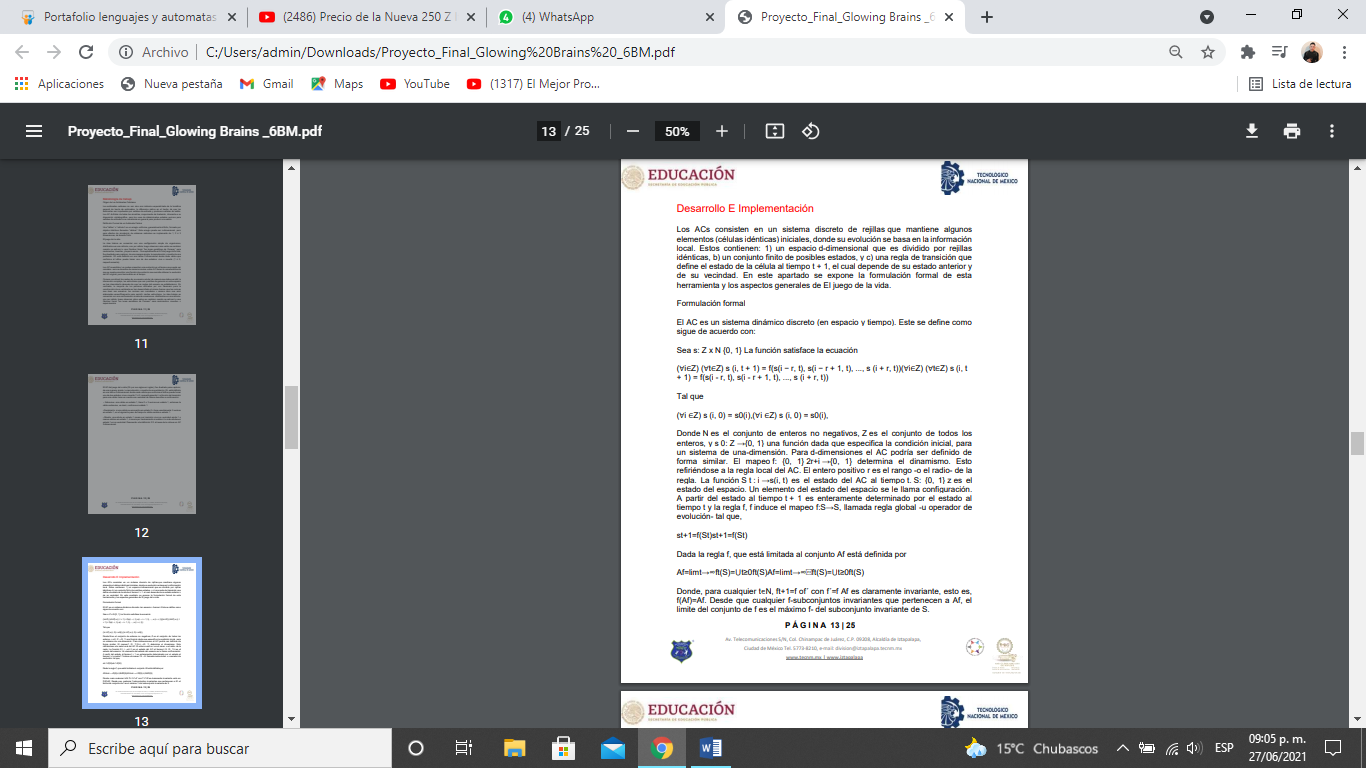


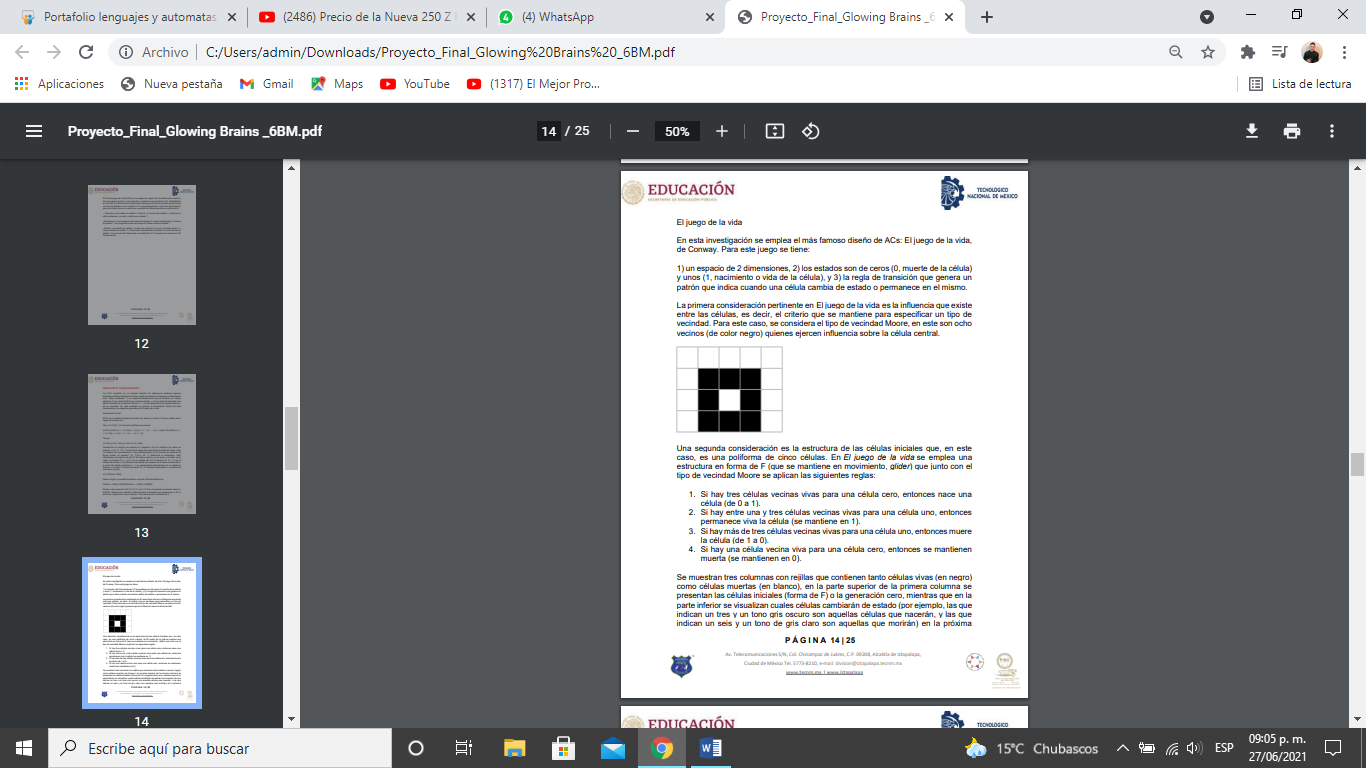


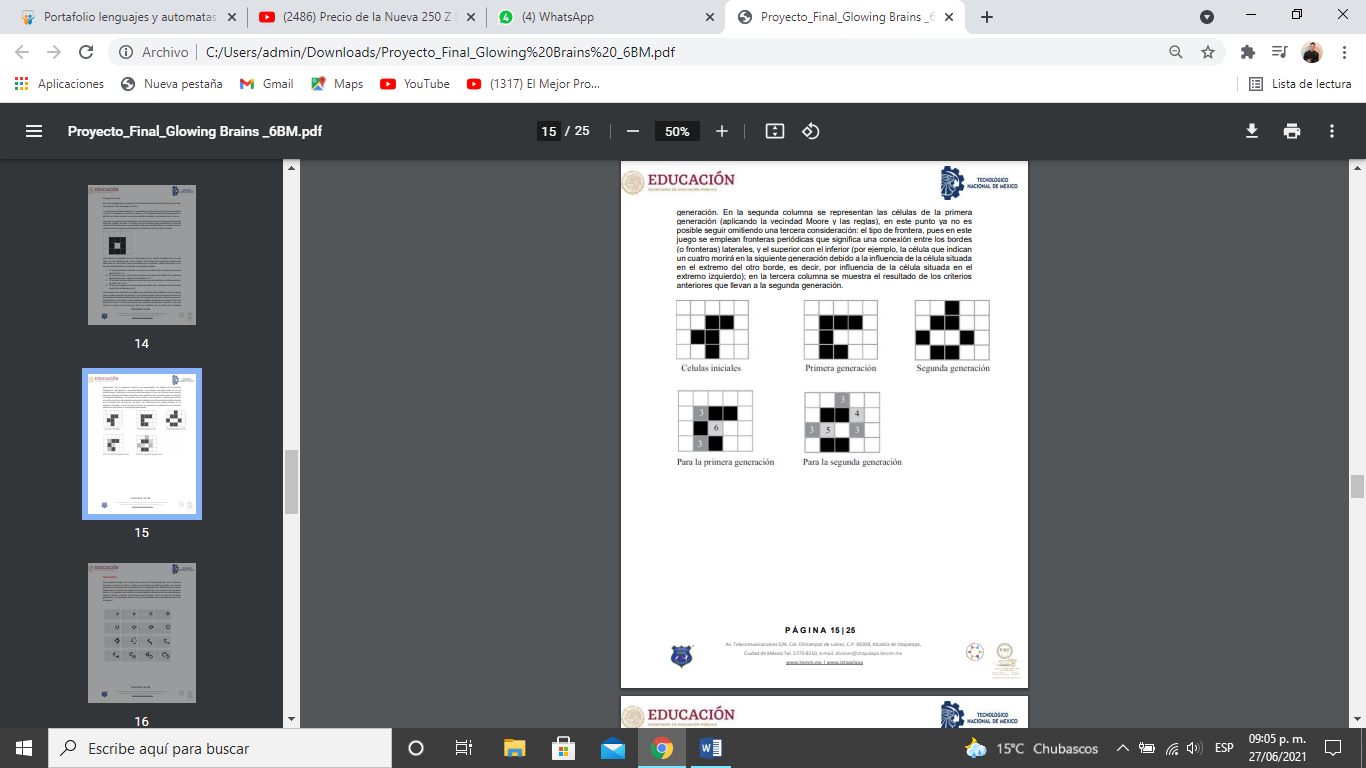


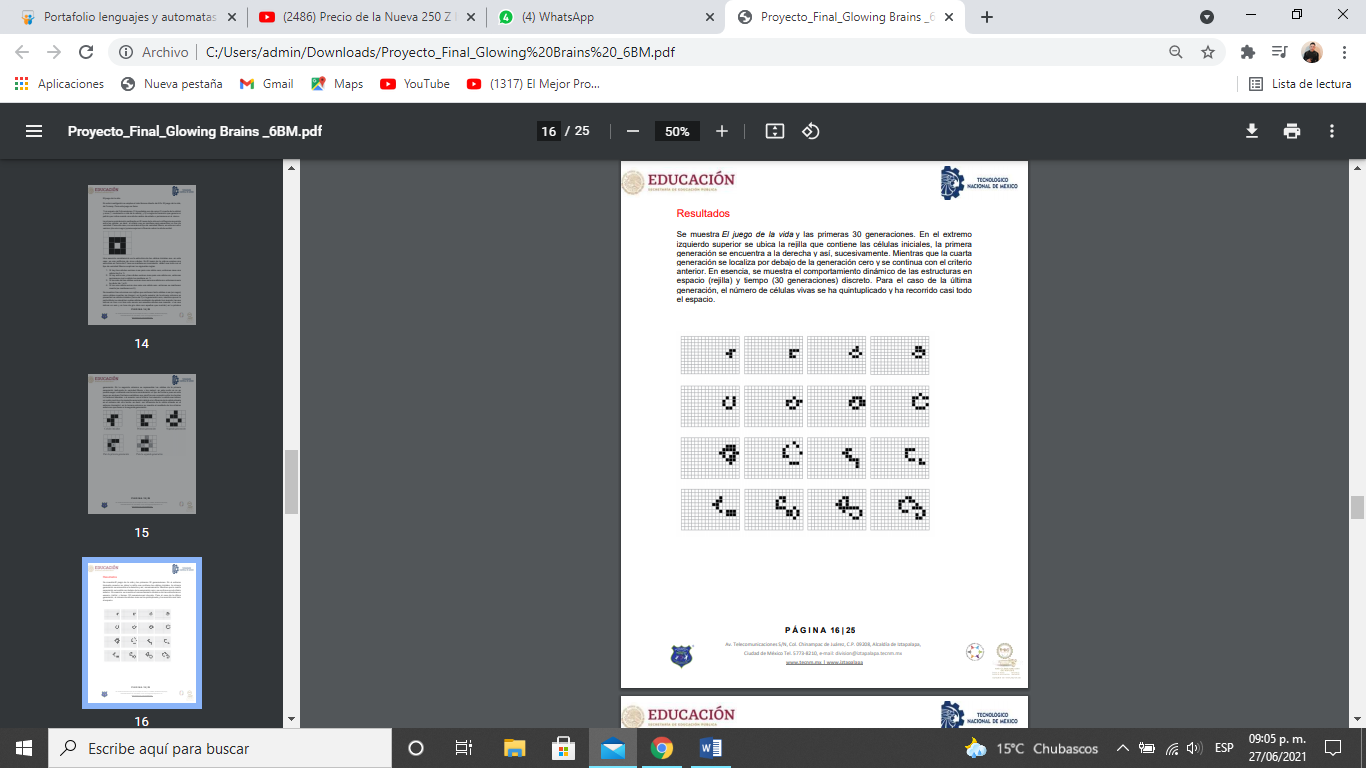


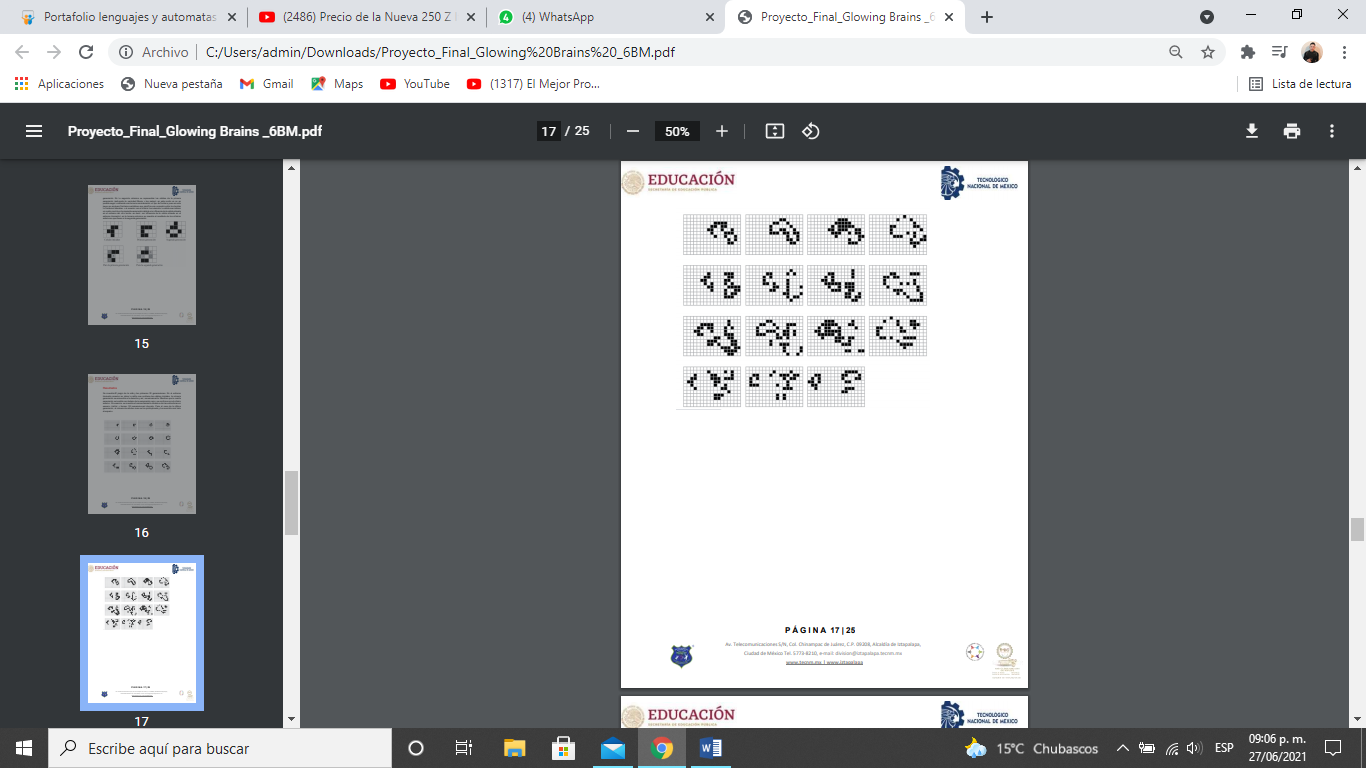


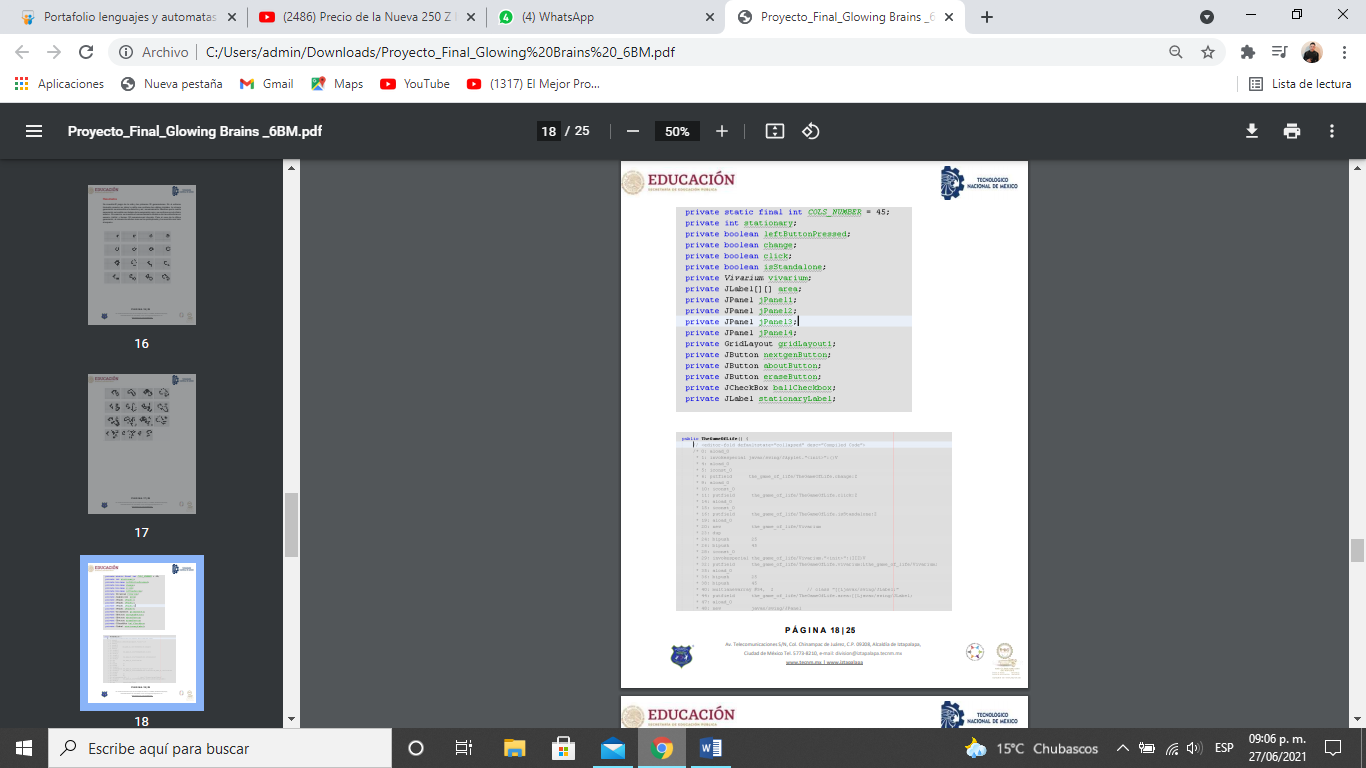


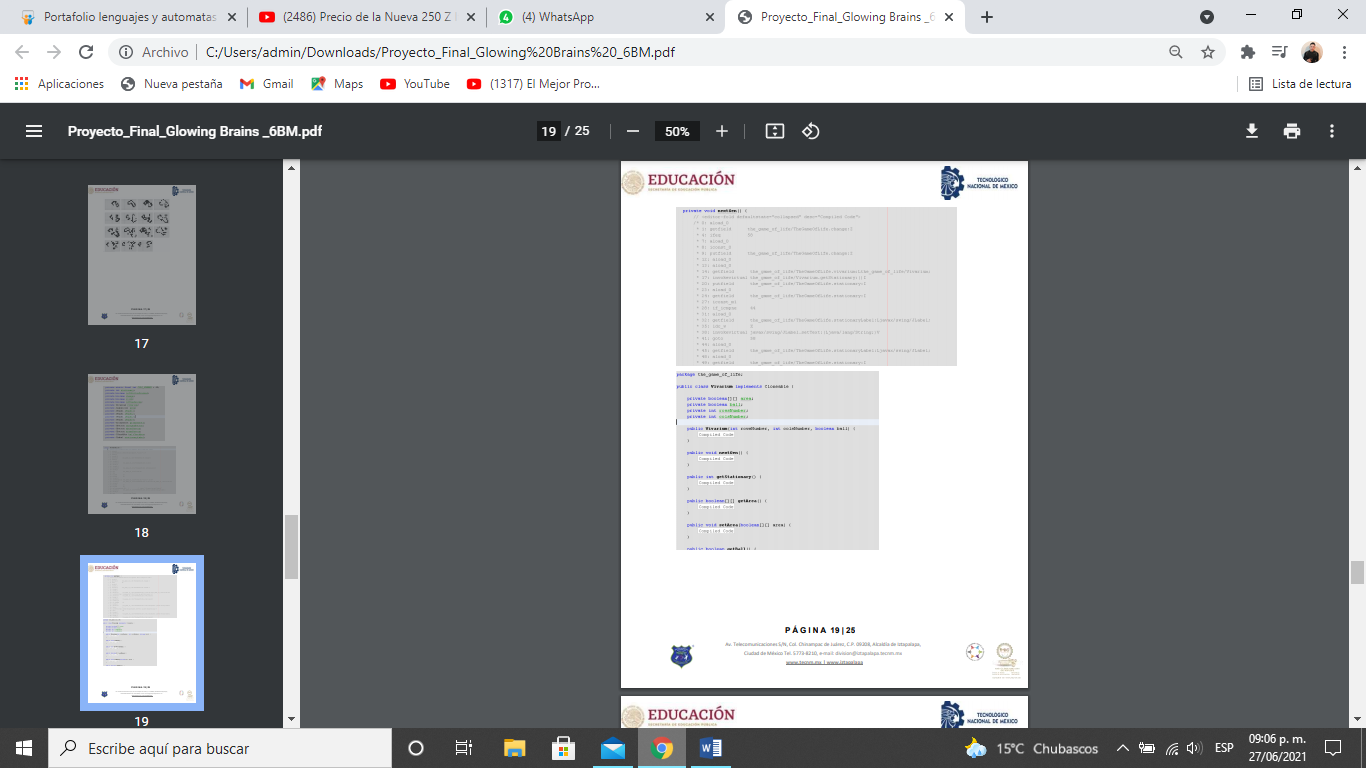


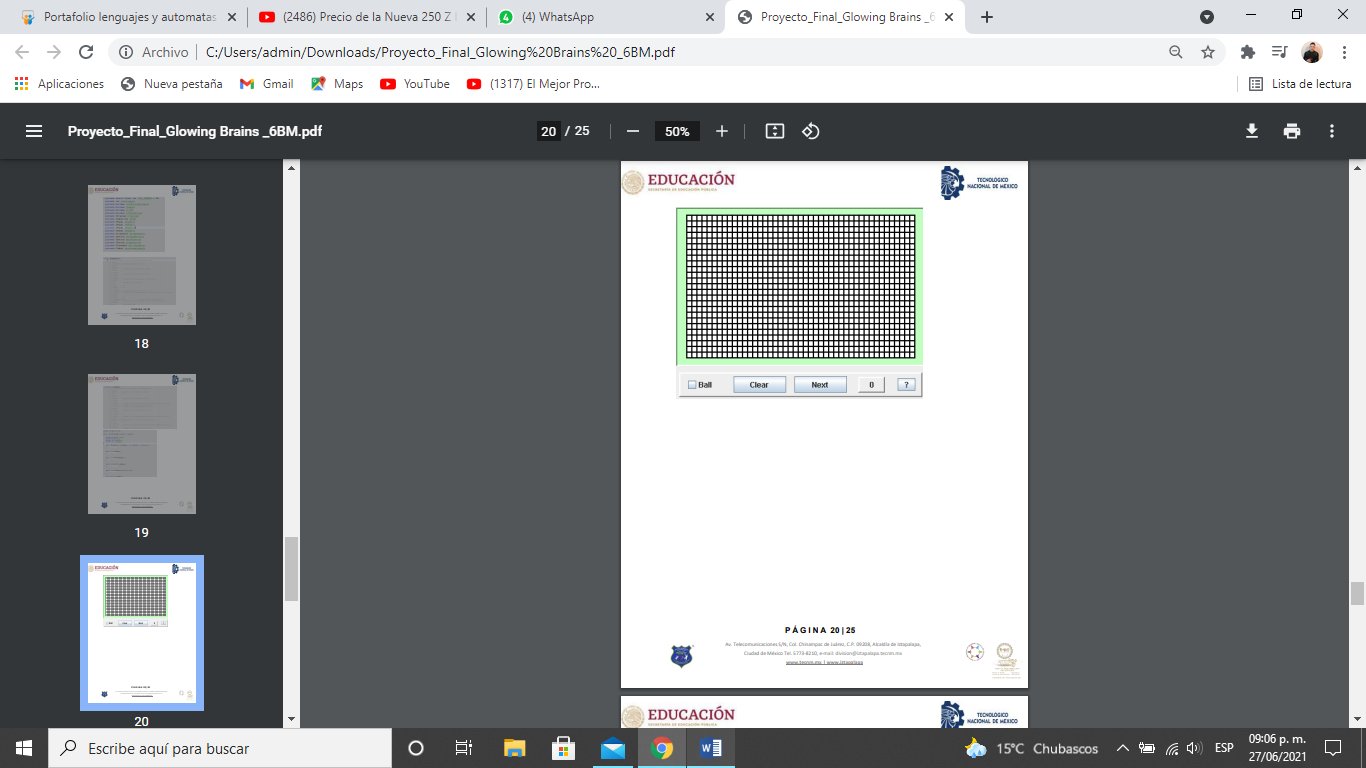


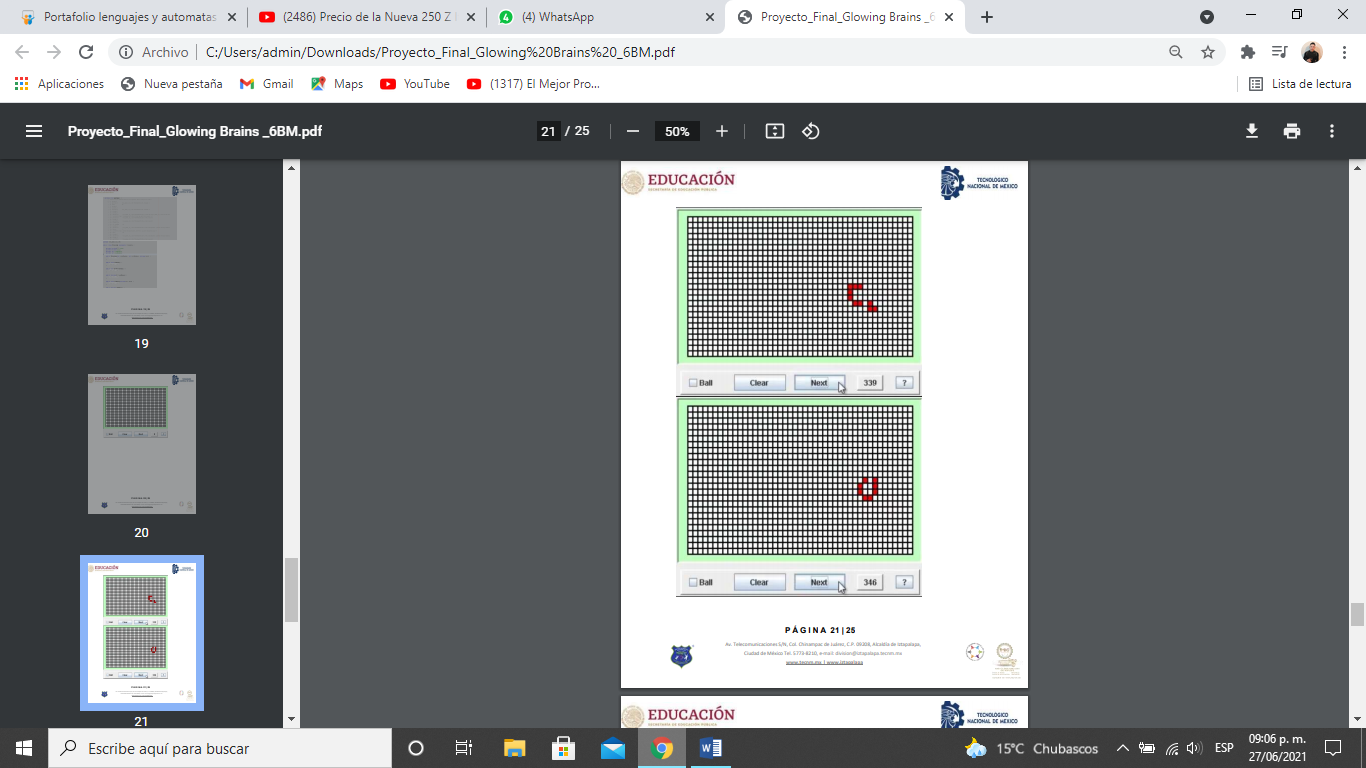


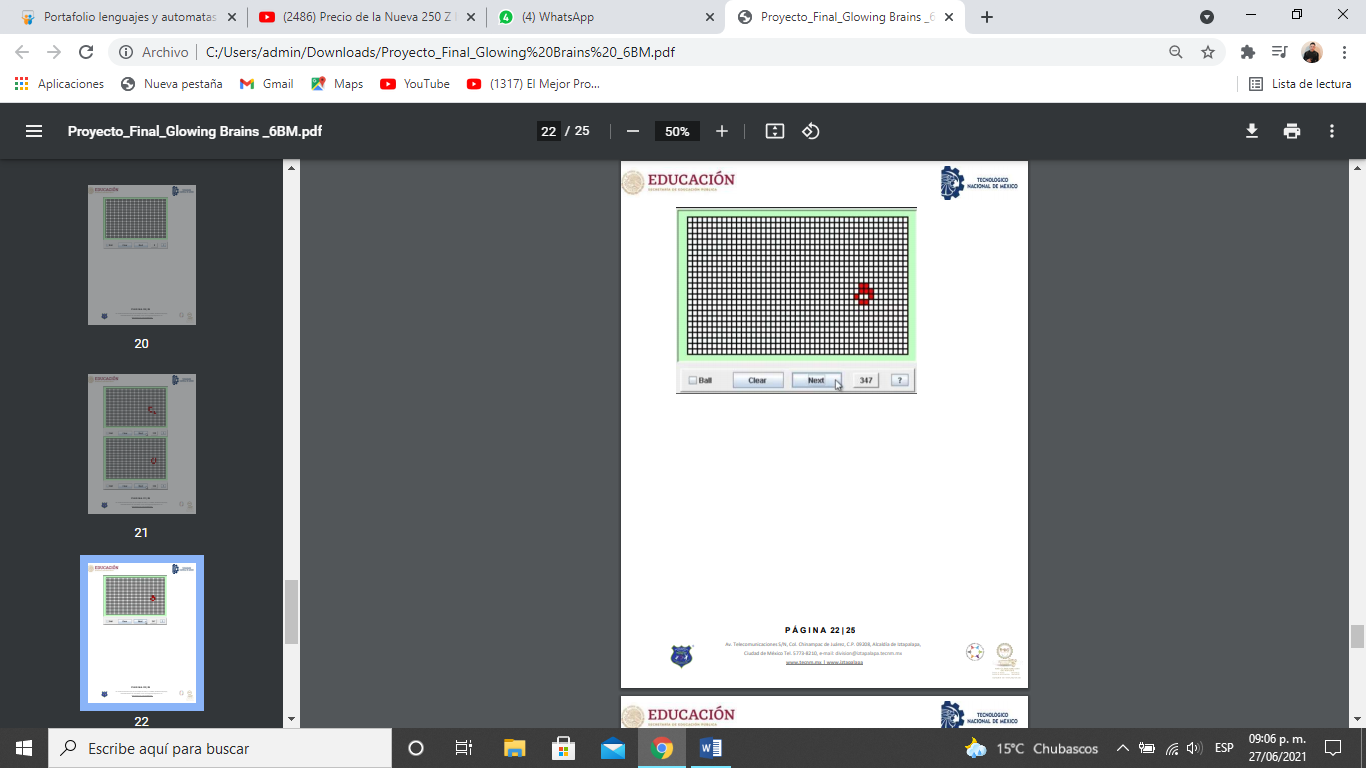


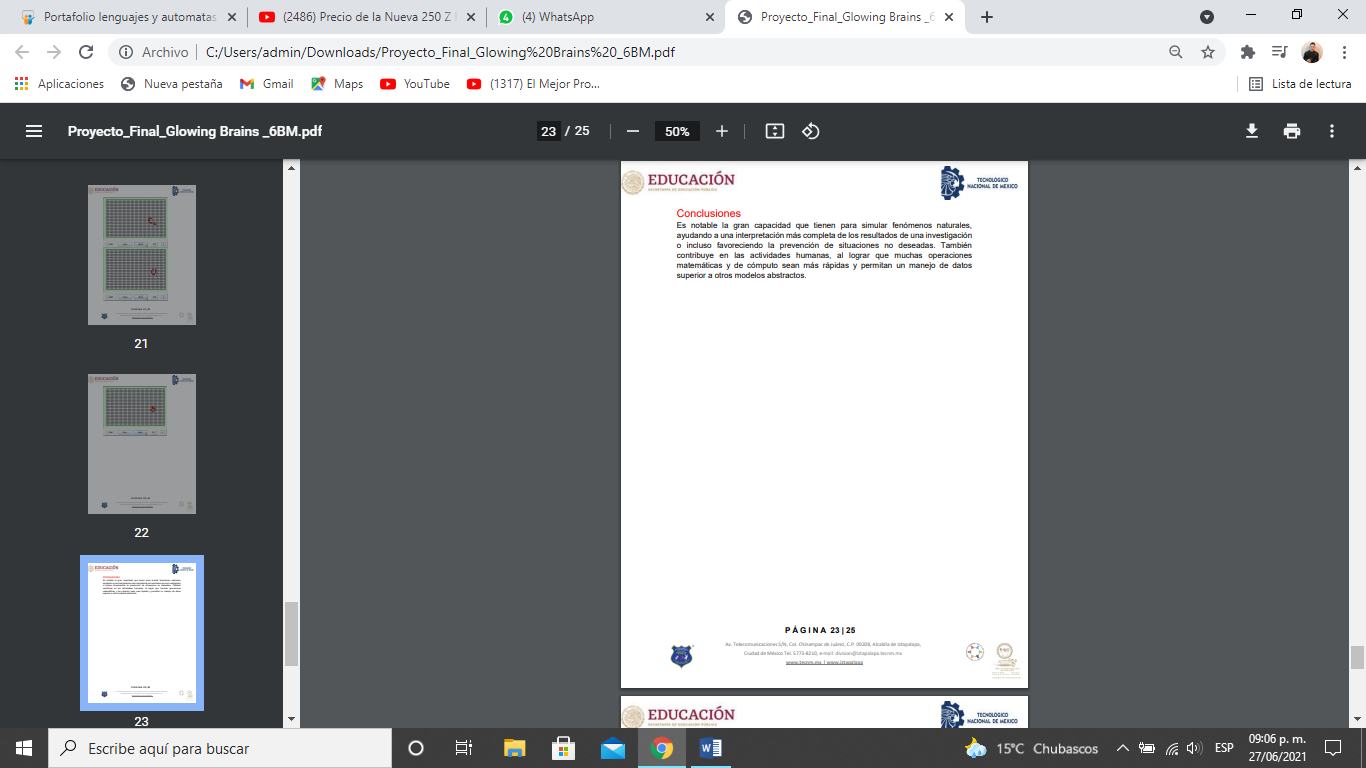


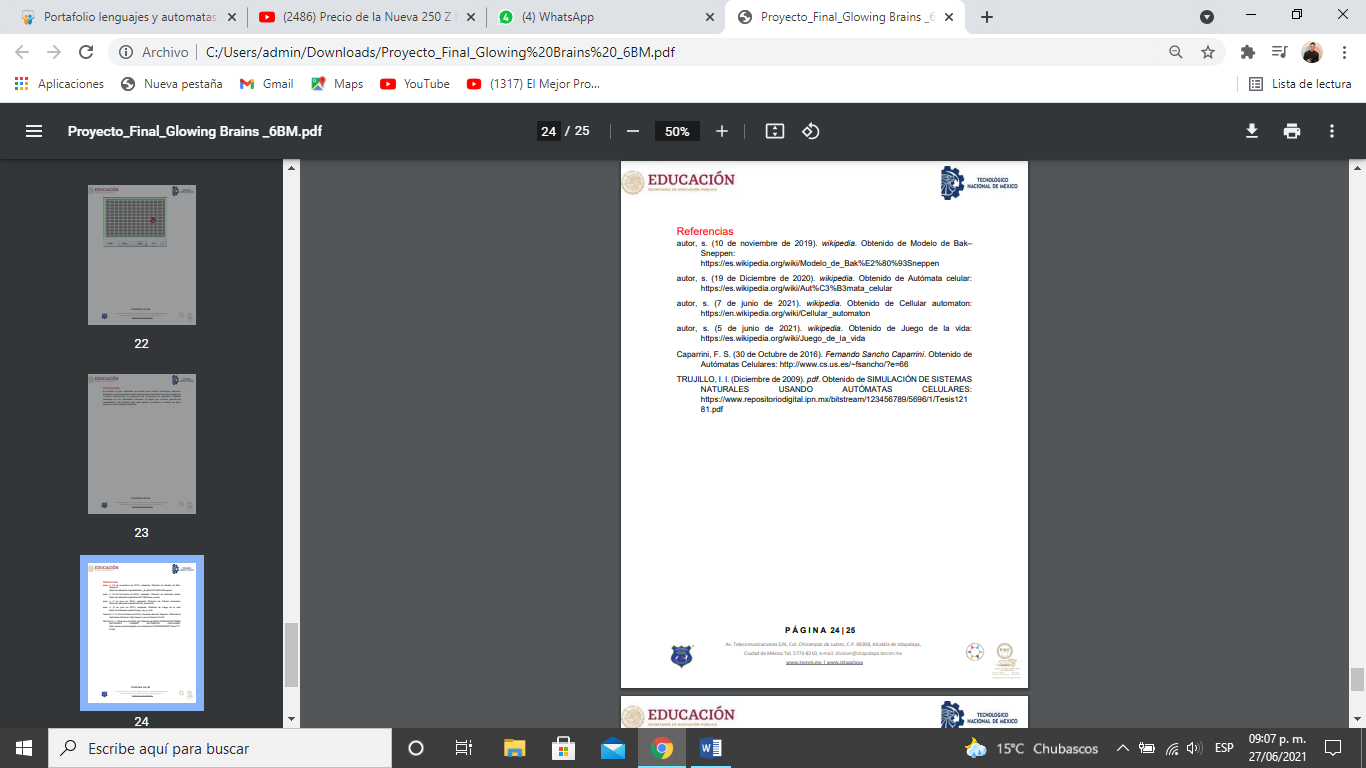


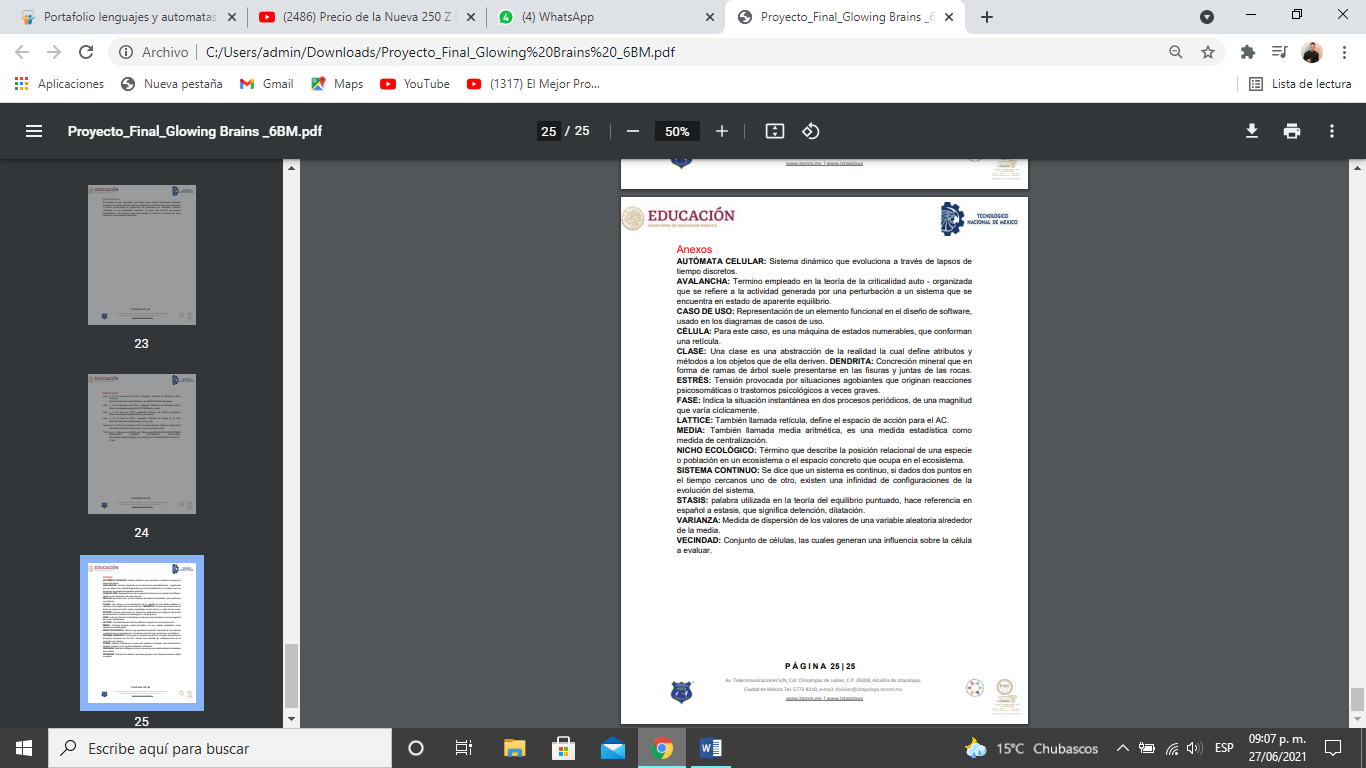












**EXPOSICIONES (LIGAS).**

**(RESUMEN)** [**https://youtu.be/eGW8VACOPmk**](https://youtu.be/eGW8VACOPmk)

**(MODELOS)** [**https://youtu.be/lvTfkxsJnm4**](https://youtu.be/lvTfkxsJnm4)