**INSTITUTO POLITÉCNICO NACIONAL**

**ESCUELA SUPERIOR DE CÓMPUTO**

**ZEPEDA FLORES ALEJANDRO DE JESÚS**

**NO. BOLETA 2016601853**

**TEORÍA COMPUTACIONAL**

**PROF. LUZ MARÍA SANCHÉZ GARCÍA**

**24 DE MAYO DE 2018**

**INTRODUCCIÓN**

Un autómata de pila es un modelo matemático de un sistema que recibe una cadena constituida por símbolos de un alfabeto y determina si esa cadena pertenece al lenguaje que el autómata reconoce.

El lenguaje que reconoce un autómata con pila pertenece al grupo de lenguajes libros de contexto en la clasificación de la jerarquía de Chomsky

**Autómata de Pila:**

* Autómatas finitos modelan adecuadamente mecanismos que requieren una memoria pequeña.
* Autómatas finitos modelan adecuadamente mecanismos que requieren memoria infinita que sólo puede ser replicada con una pila LIFO (Last In First Out).

**Autómata de pila reconocedor determinístico (Q, ∑, P, δ, Q0, Z0, F)**

* Q: Conjunto finito de estados
* ∑: Alfabeto conjunto finito de símbolos de entrada
* P: Alfabeto o conjunto finito de símbolos
* δ: Función de transición de estados
* Q0: Estado inicial
* Z0: Símbolo distinguido del fondo de la pila
* F: Conjunto de estados finales

**PLANTEAMIENTO DEL PROBLEMA**

Se programará el autómata de pila con sus transiciones. Usará una pila como memoria auxiliar para hacer los recuentos y comparaciones sobre las palabras de entrada. Validará cadenas que pertenezcan al lenguaje del autómata de pila. Mostrar el contenido de la pila después de haber leído una cadena de entrada si esta es aceptada o rechazada.

**DIAGRAMA**

Establecer lineamientos a seguir para el óptimo funcionamiento.

Conocer los estados y las transiciones para generarlo.

Seleccionar un tipo de lenguaje para programar el autómata de pila que lo genere.

Existen errores

Verificar que el lenguaje al final de la ejecución sea aceptado o rechazado.

FIN

Elaborar reporte.

NO

SI

Corregir errores y realizar pruebas.

Realizar las pruebas necesarias para checar la implementación.

Utilizar la estructura de datos “PILA” junto con su conjunto de operaciones.

INICIO

**IMPLEMENTACIÓN DE LA SOLUCIÓN**

Para la implementación de este programa, vamos a leer la cadena de entrada desde un archivo de texto plano. Posteriormente asignamos el lenguaje de pila para insertar en caso de encontrar un carácter de apertura.

int autopila(char \* namefile,int cantidad){

FILE \* file; char caracter;

struct nodo \* stack;

struct informacion dato;

char \* cadena = automata(namefile,cantidad);

int apertura = 0, clausura = 0;

iniciar(&stack); fflush(stdin);

file = fopen(namefile,"r");

caracter = fgetc(file);

while(caracter != EOF){

switch(caracter){

case '{': dato.llave = 'A';push(&stack,dato);apertura = apertura + 1;break;

case '[': dato.llave = 'B';push(&stack,dato);apertura = apertura + 1;break;

case '(': dato.llave = 'C';push(&stack,dato);apertura = apertura + 1;break;

break;

case '}': case ']': case ')':

clausura = clausura + 1;

if(!empty(stack)){

dato = top(stack);

if((dato.llave=='A' && caracter=='}')||(dato.llave=='B' && caracter==']')||(dato.llave=='C' && caracter==')')){

pop(&stack);

}

}

break;

}

caracter = fgetc(file);

}

if(apertura == clausura){

if(empty(stack))

printf("Cadena %s aceptada - PILA VACIA\n\n",cadena);

else{

printf("Cadena %s rechazada - PILA NO VACIA\n\n",cadena);

while(!empty(stack)){

printf("\*\*\*\*\*\n");

if(top(stack).llave == 'A') printf("\* %c \* Lenguaje para llave\n", top(stack).llave);

if(top(stack).llave == 'B') printf("\* %c \* Lenguaje para corchete\n", top(stack).llave);

if(top(stack).llave == 'C') printf("\* %c \* Lenguaje para parentesis\n", top(stack).llave);

pop(&stack);

}

printf("\*\*\*\*\*\n\n");

}

}

else{

if(apertura<clausura)

printf("Cadena %s rechazada - PILA VACIA\n\n",cadena);

else{

printf("Cadena %s rechazada - PILA NO VACIA\n\n",cadena);

while(!empty(stack)){

printf("\*\*\*\*\*\n");

if(top(stack).llave == 'A') printf("\* %c \* Lenguaje para llave\n", top(stack).llave);

if(top(stack).llave == 'B') printf("\* %c \* Lenguaje para corchete\n", top(stack).llave);

if(top(stack).llave == 'C') printf("\* %c \* Lenguaje para parentesis\n", top(stack).llave);

pop(&stack);

}

printf("\*\*\*\*\*\n\n");

}

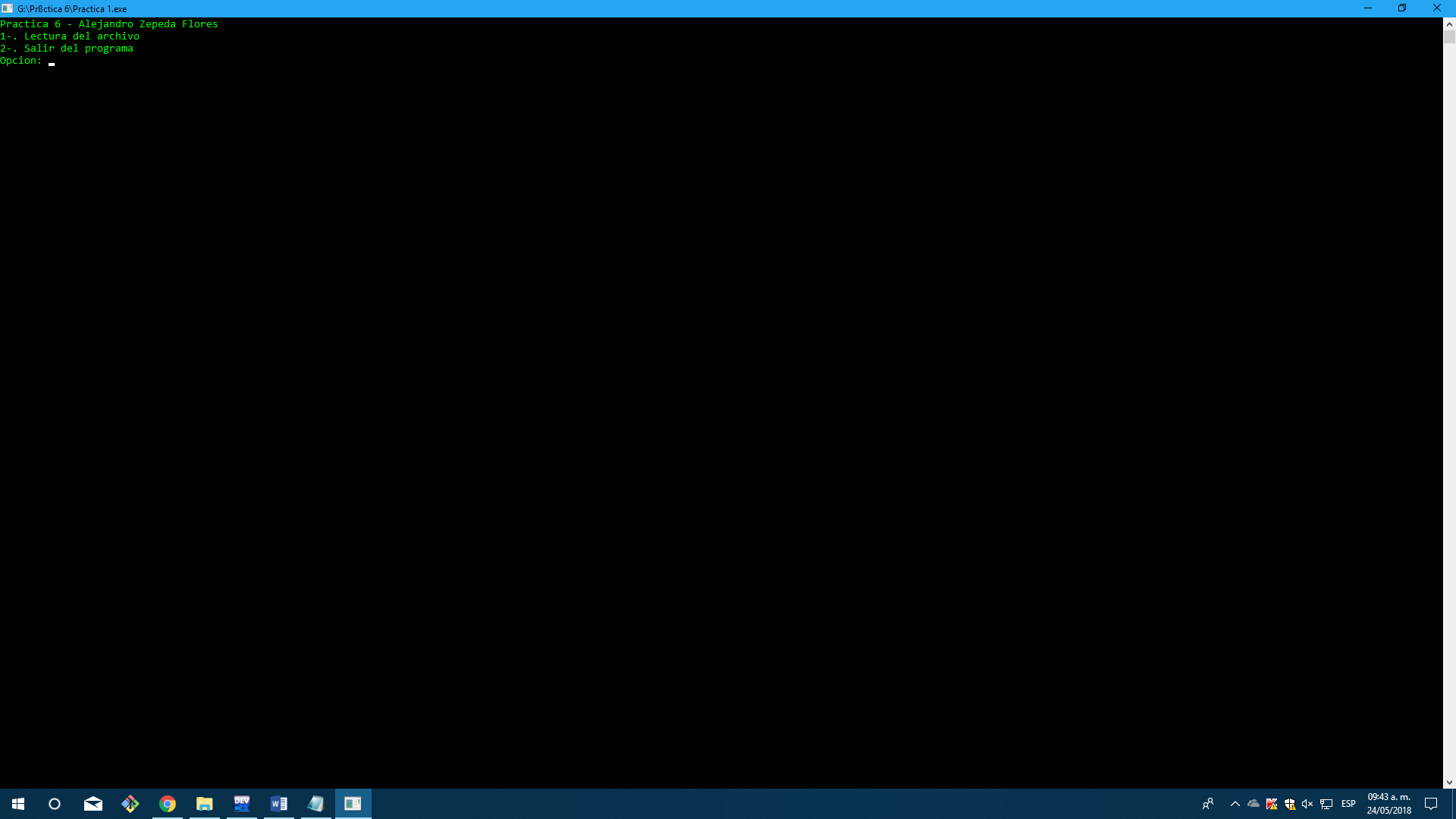
}

fclose(file);

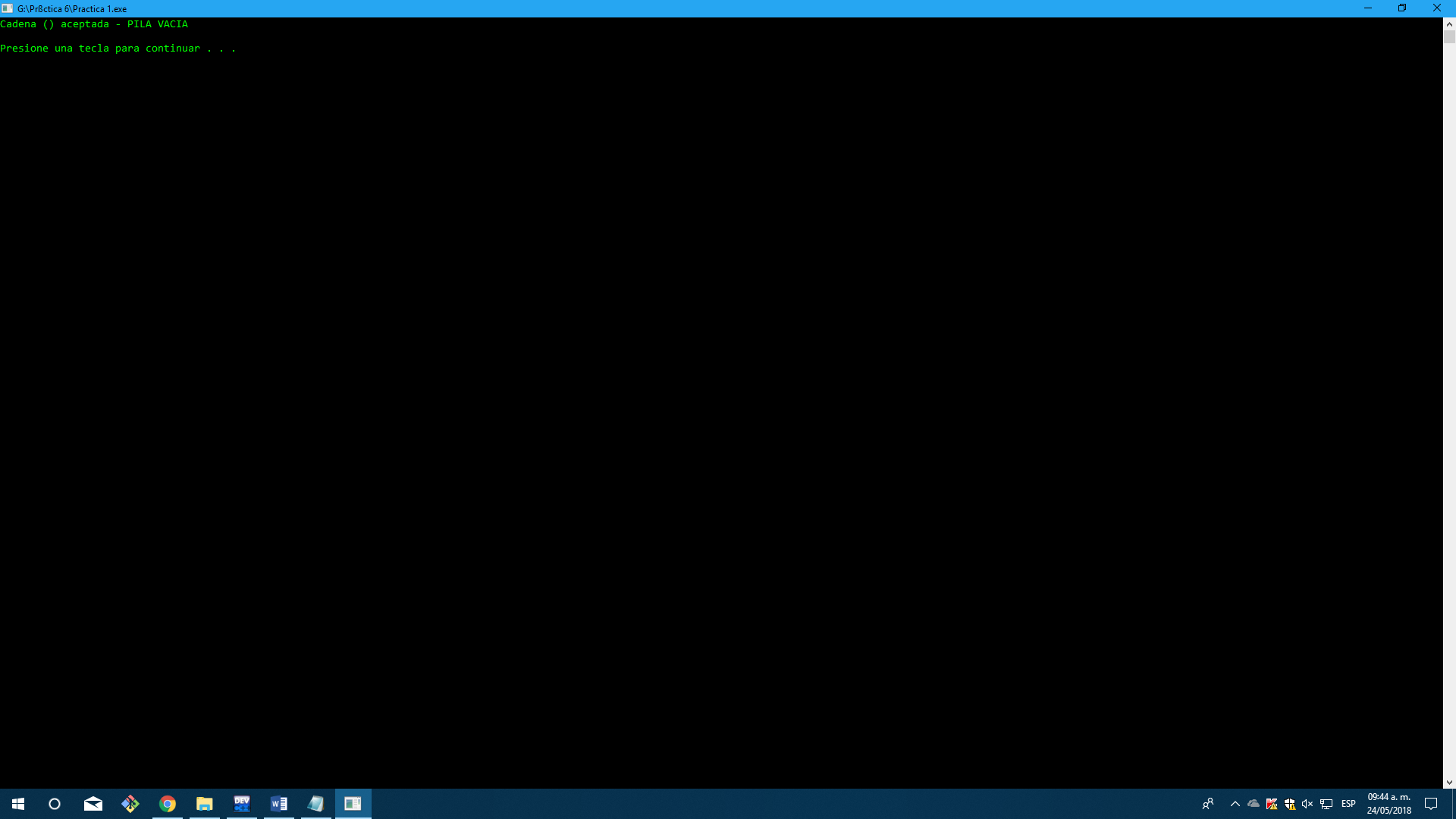
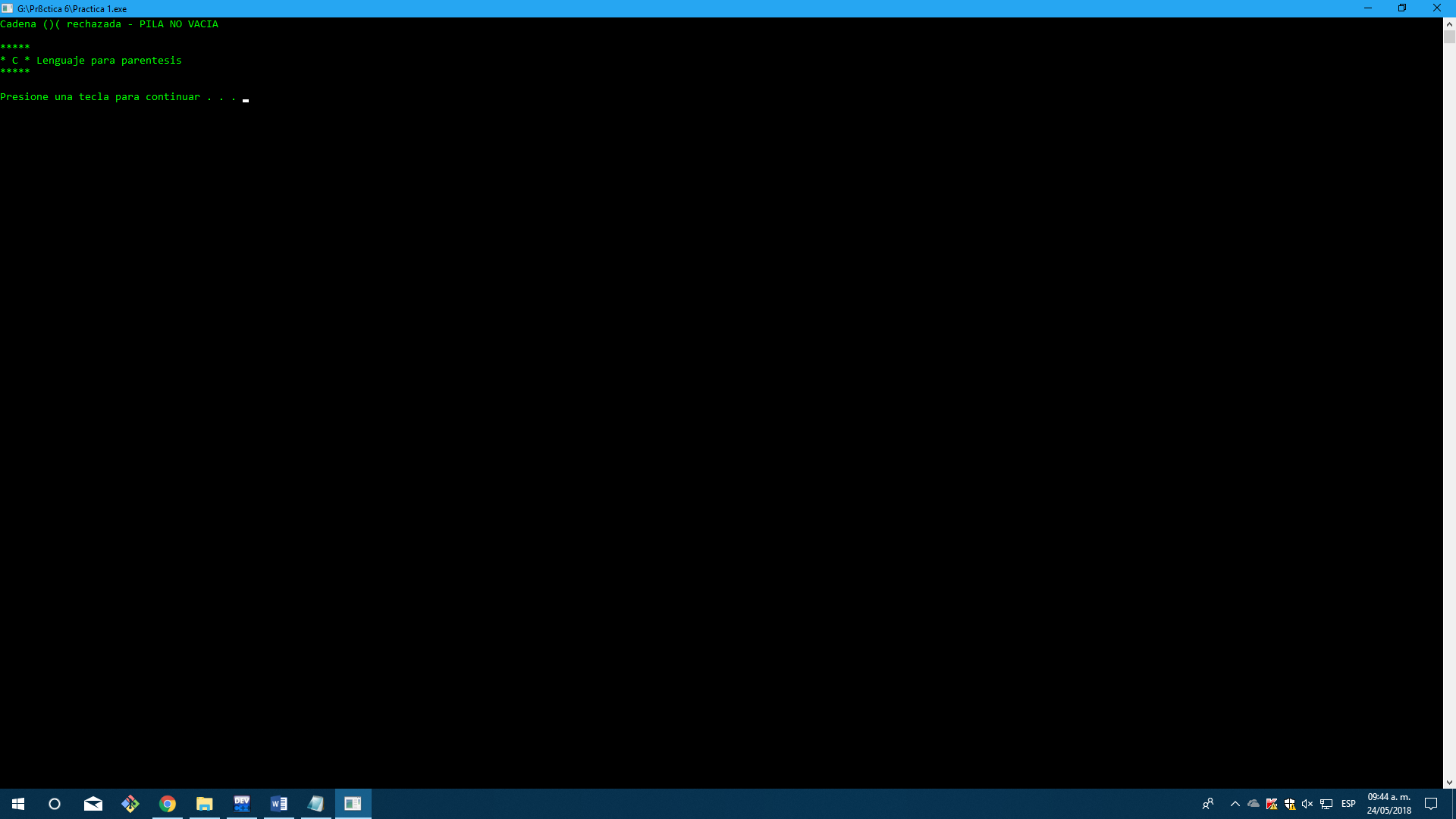
}

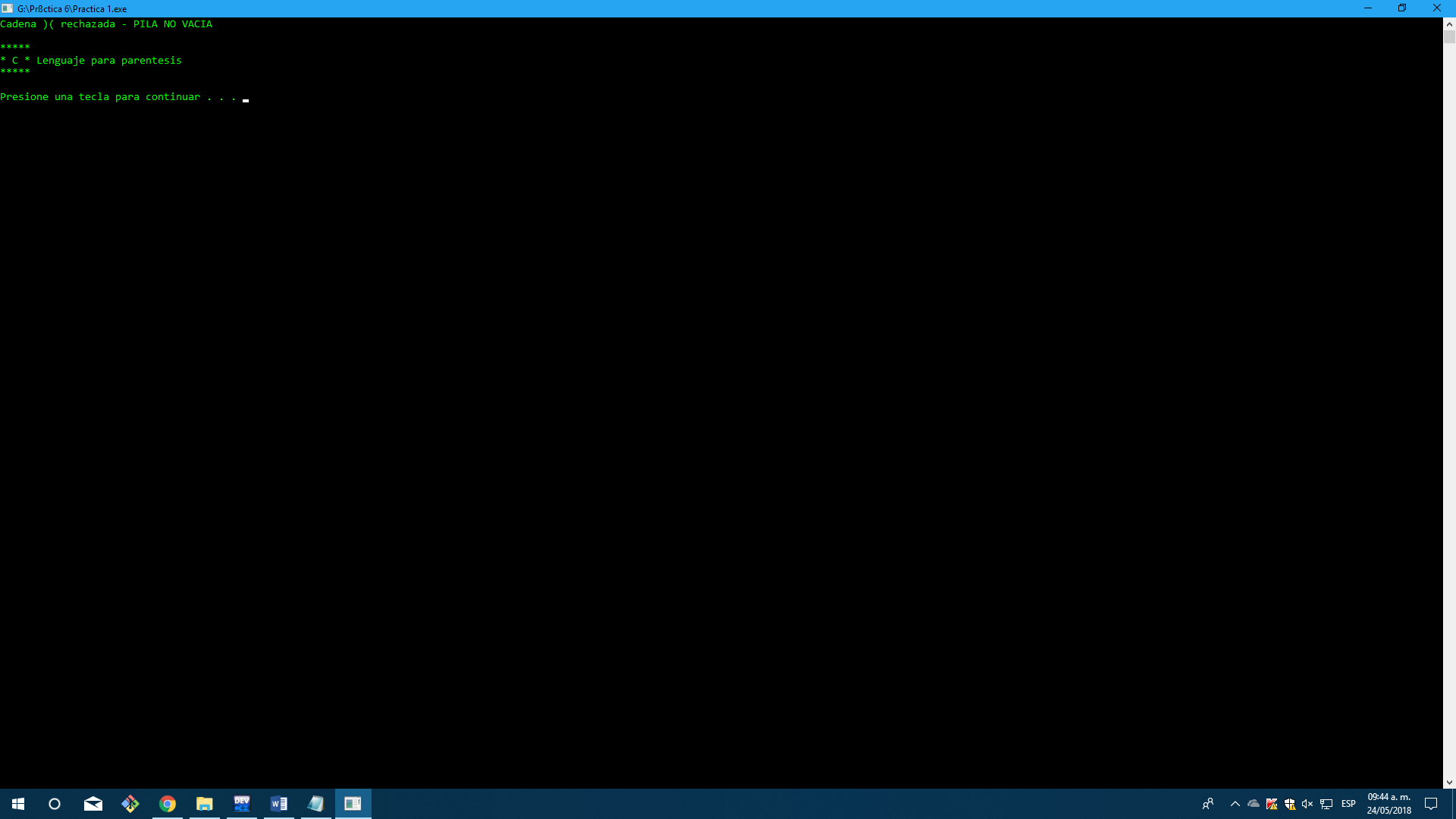
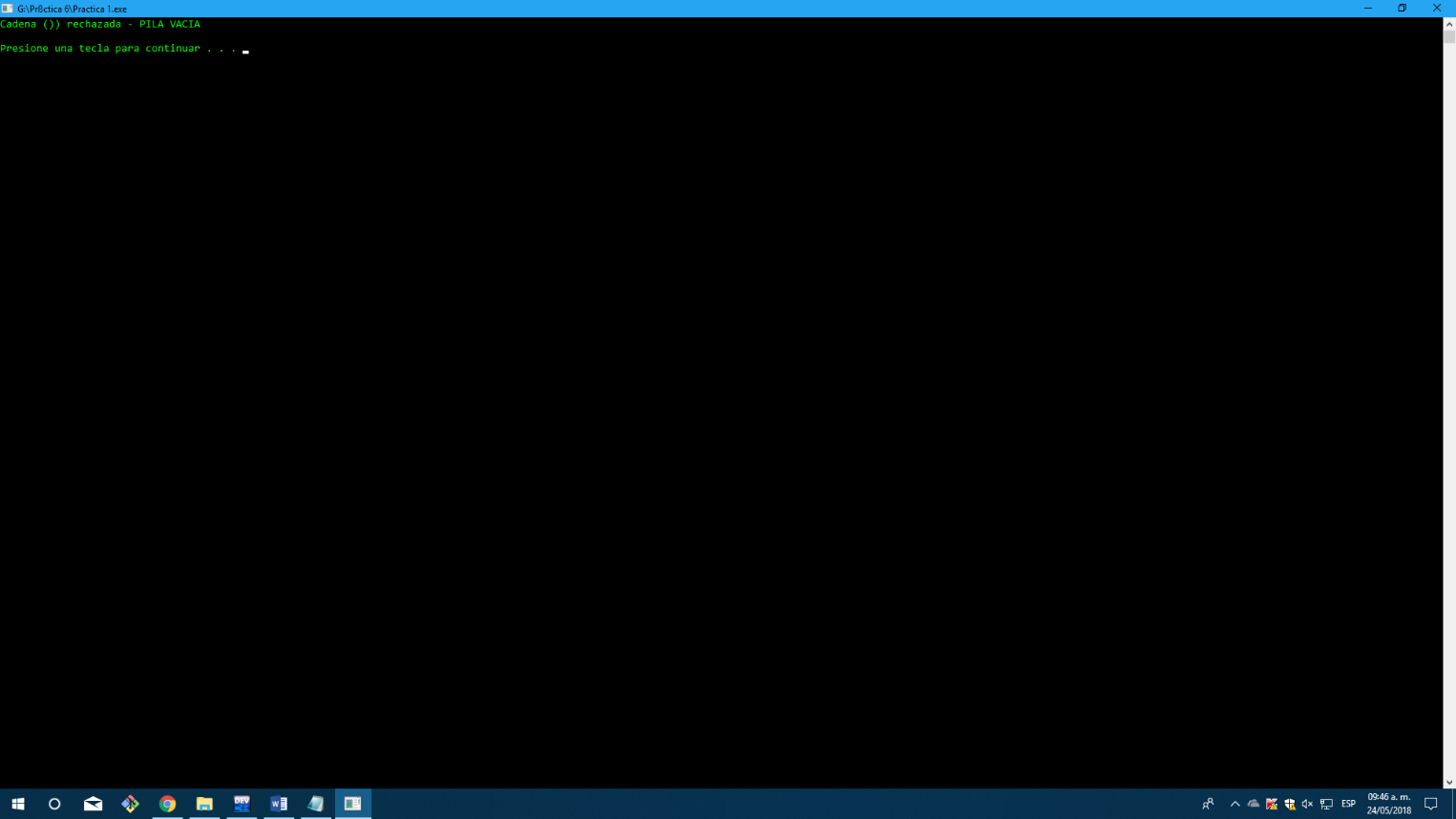
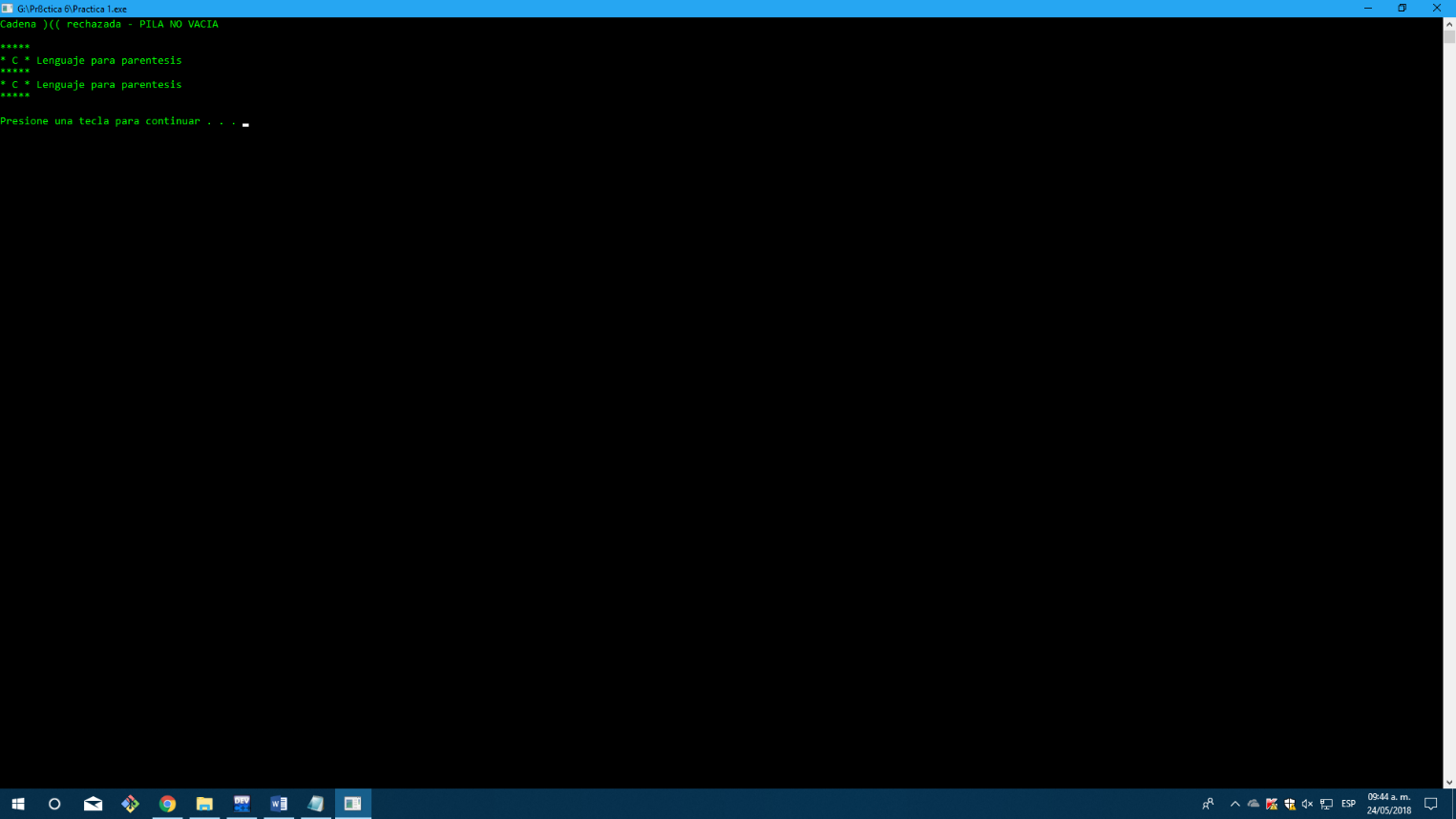
Seguimos con la lectura y al terminar de leer evaluamos la pila. Si está vacía significa que la cadena fue aceptada, caso contrario puede ser que falten paréntesis de apertura o cierre o que no coincidan.

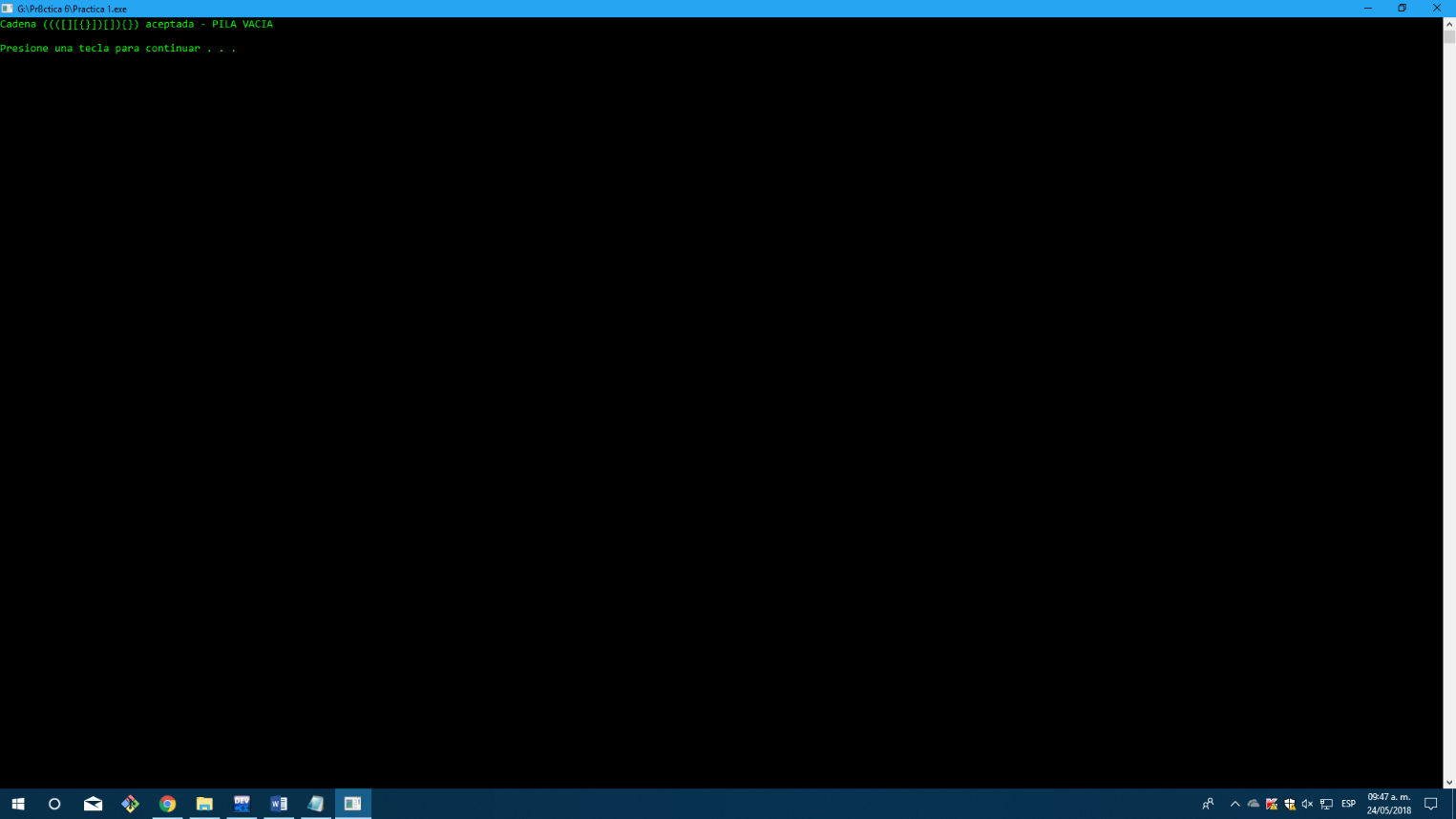
**FUNCIONAMIENTO**



Interfaz de usuario







**CONCLUSIÓN**

Esta es una buena práctica para poner a prueba los conocimientos adquiridos en la asignatura de estructuras de datos. De hecho, reutilicé el código de la pila implementada en dicha materia y sólo tuve que modificarlo para que se convirtiera en un autómata de pila. Con esto comprobamos que hay más formas de generar lenguajes o cadenas y que sólo es el comienzo para la máquina de Turing.

**BIBLIOGRAFÍA**

* Franco Martínez Edgardo Adrián. (2013). Autómatas de pila. mayo 24, 2018, de Instituto Politécnico Nacional Sitio web: http://www.eafranco.com/docencia/teoriacomputacional/

files/17/Clase\_17.pdf