Ejercicio 1.

Dadas las siguientes tareas indique:

- Qué tipo de módulos necesita para resolverlas (procedimientos o funciones)

- Escriba solo el encabezamiento de los mismos, definiendo los parámetros que se necesitan

mencione si se trata de parámetros por valor o referencia.

- Indique con qué estructuras de datos podría resolver las tareas

a) Obtener la cantidad de vocales de un string:

- Funciones - def cant\_vocales(palabra) - "palabra" es de tipo string y es un parámetro por valor, y devuelve un valor numérico entero (int) que hace referencia a la cantidad de vocales en "palabra"-No usar estructura de datos

b) Convertir un valor numérico en letras. (Ej. para el número 15 debe convertir a “quince”)

-Funciones- def conv\_a\_letra(numero)- "numero" es de tipo int(entero) y es parámetro por valor, y devuelve el valor de tipo string que hace referencia al número convertido en una palabra - No usa estructura de datos

c) Carga de un vector y cálculo del promedio de sus números.

Cargar un vector: -Procedimiento- def cargar\_vector(X) - "X" es una estructura de tipo Array y es un parametro por referencia. Esta funcion devuelve un vector cargado

Calculo del promedio de sus números: -Funciones - def promedio(X)- "X" es un vector y es un parámetro por valor. Esta función devuelve un valor de tipo Int, que es el promedio de todos los números cargados en "X"

d) Calcular el perímetro de un cuadrado.

-Funciones- def perimetro(longitud) - "longitud" es de tipo string y es un parametro por valor. Esta funcion devuelve un valor numerico de tipo float que es la multiplicación de longitud por 4 - No usa estructura de datos

e) Almacenar los diferentes precios y el stock de artículos de un negocio

-Procedimiento - def almacenar\_articulos(X) - "X" es una estructura de tipo Array y un parámetro por referencia. Esta devuelve una matriz cargada

f) Ordenar los elementos de un vector, en forma ascendente o descendente.

-Funciones - def ordenar(vector) - "vector" es una estructura de tipo Array y parámetro por referencia. Este devuelve un vector ordenado

g) Calcular el promedio de notas de alumnos almacenadas en un archivo

Buscar notas: -Funciones- def buscar\_notas(vector)-"vector" es una estructura de tipo Array y parámetro de valor. Esta devuelve las notas de un alumno donde cada valor es de tipo int

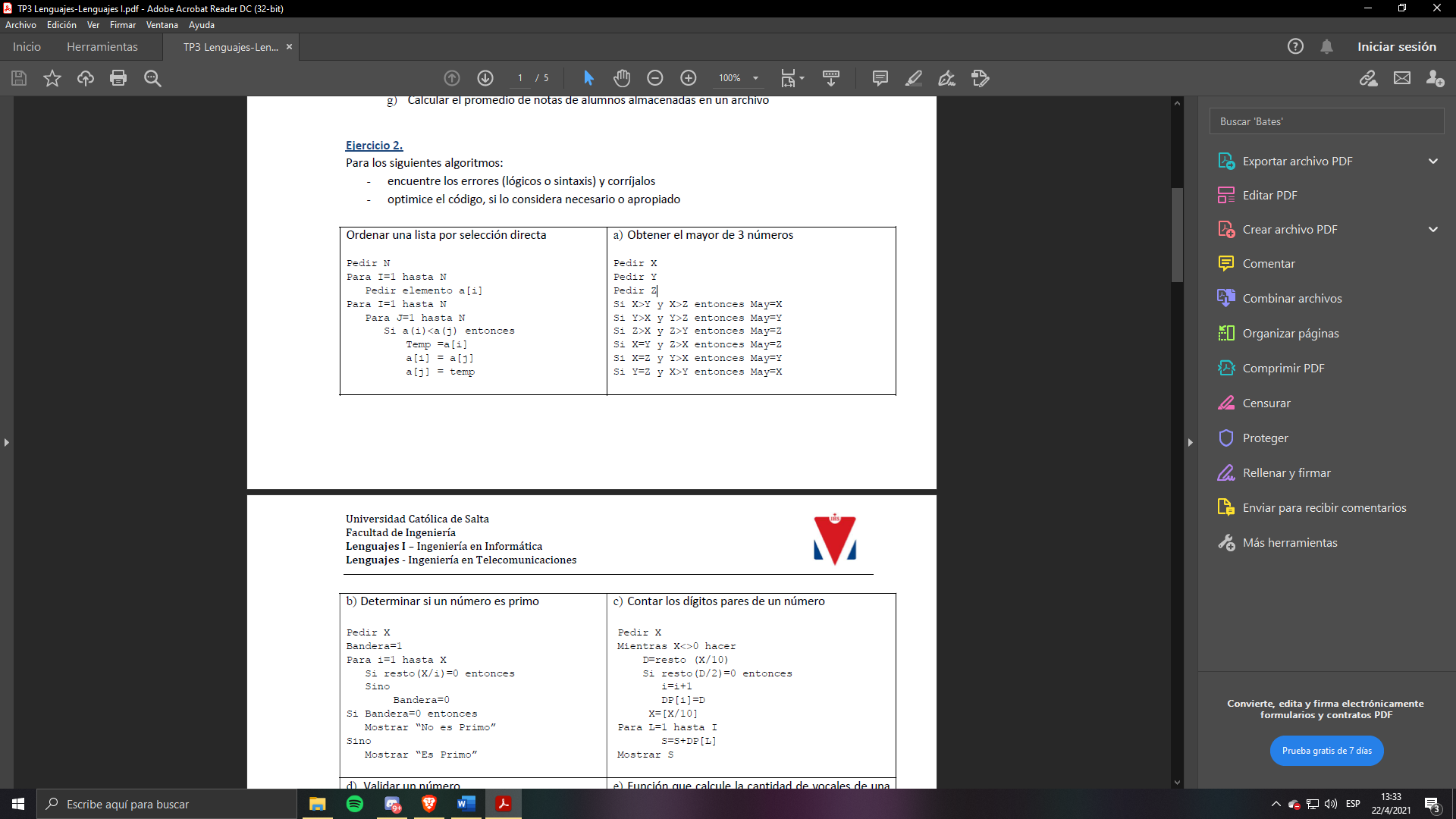
Calcular promedio de notas: -Funciones - def promedio(vector) - "vector" es una estructura de tipo Array y parámetro por valor. Esta devuelve un valor de tipo entero int

Ejercicio 2

Para los siguientes algoritmos:

- encuentre los errores (lógicos o sintaxis) y corríjalos

- optimice el código, si lo considera necesario o apropiado



a) Ordenar una lista por selección directa

-Error lógico línea 3. Pide a[i] en vez de a[I] (mayusc)

-Error sintaxis línea 2 y 4. Se define I dos veces

- Error lógico línea 6 (8 y 9). Ocurre el error 1 y el mismo pero en vez de i usan j

-Error lógico línea 9. Se usa temp en vez de Temp

b) Ordenar el mayor de 3 números

-Se puede optimizar

Si X=>Y y X=>Z

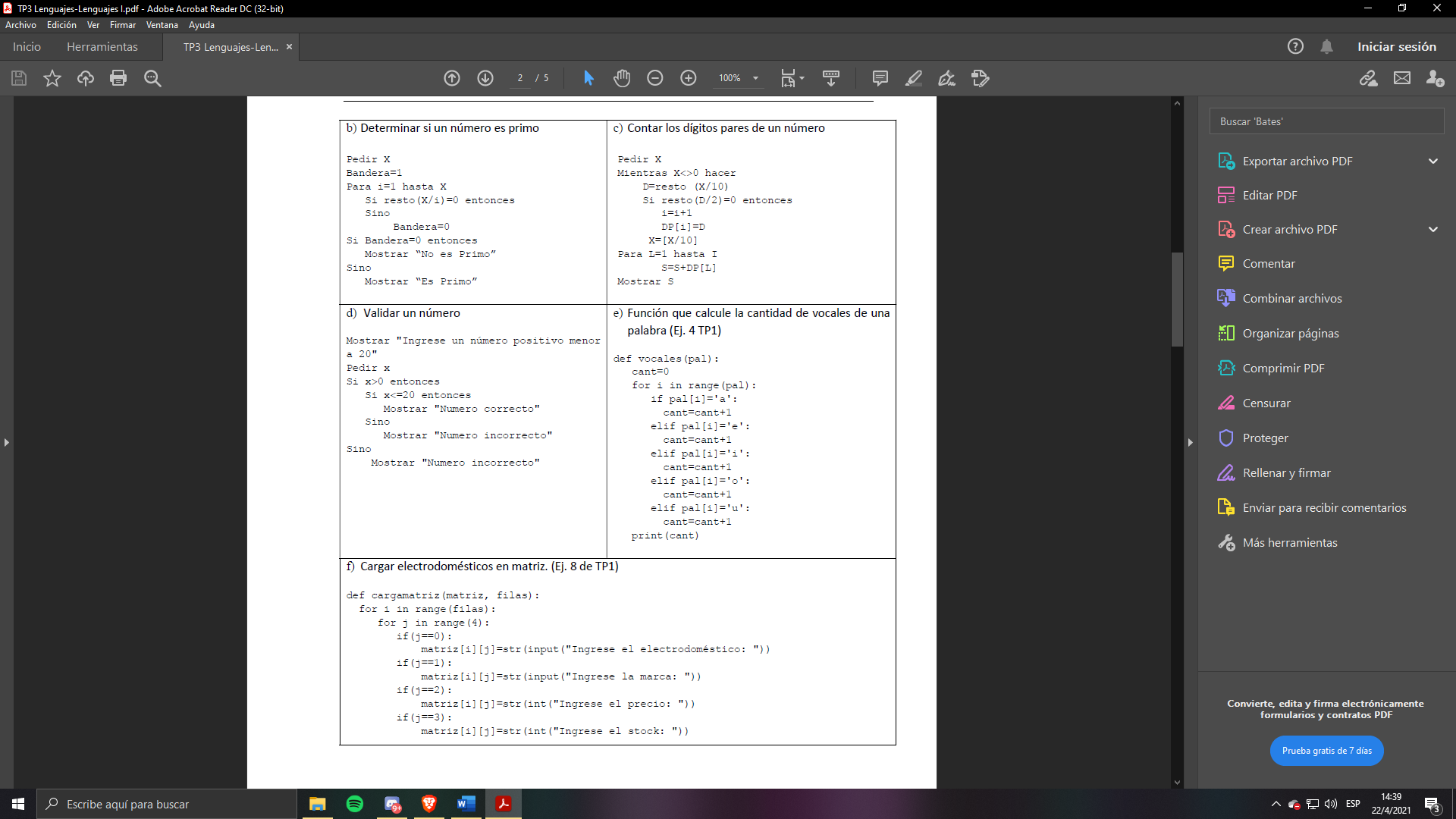
May=X

Si Y=>X y Y=>Z

May= Y

Si Z=>Z y Z=>Y

May=Z



b) Determinar si es un numero primo

-Error sintaxis línea 4 (X%2)==0 además del sino que esta de mas

c) Contar los dígitos pares de un numero

Falto inicia i=0

S=0

Error sintaxis línea 2 (X<>0)

S=S+D

d)Validar un numero

Se puede simplificar

if(x>0 and x<20)

mostrar "Numero correcto"

sino "Numero incorrecto"

e) Cantidad de vocales de una palabra

def vocales (pal):

cant=0

for i in range (pal):

if pal[i]=”a” or pal[i]=”e” or pal[i]=”i” or pal[i]=”o” or pal[i]=”u”:

cant= cant +1

print (cant)

f)Cargar electrodomésticos

def cargamatriz (matriz,filas):

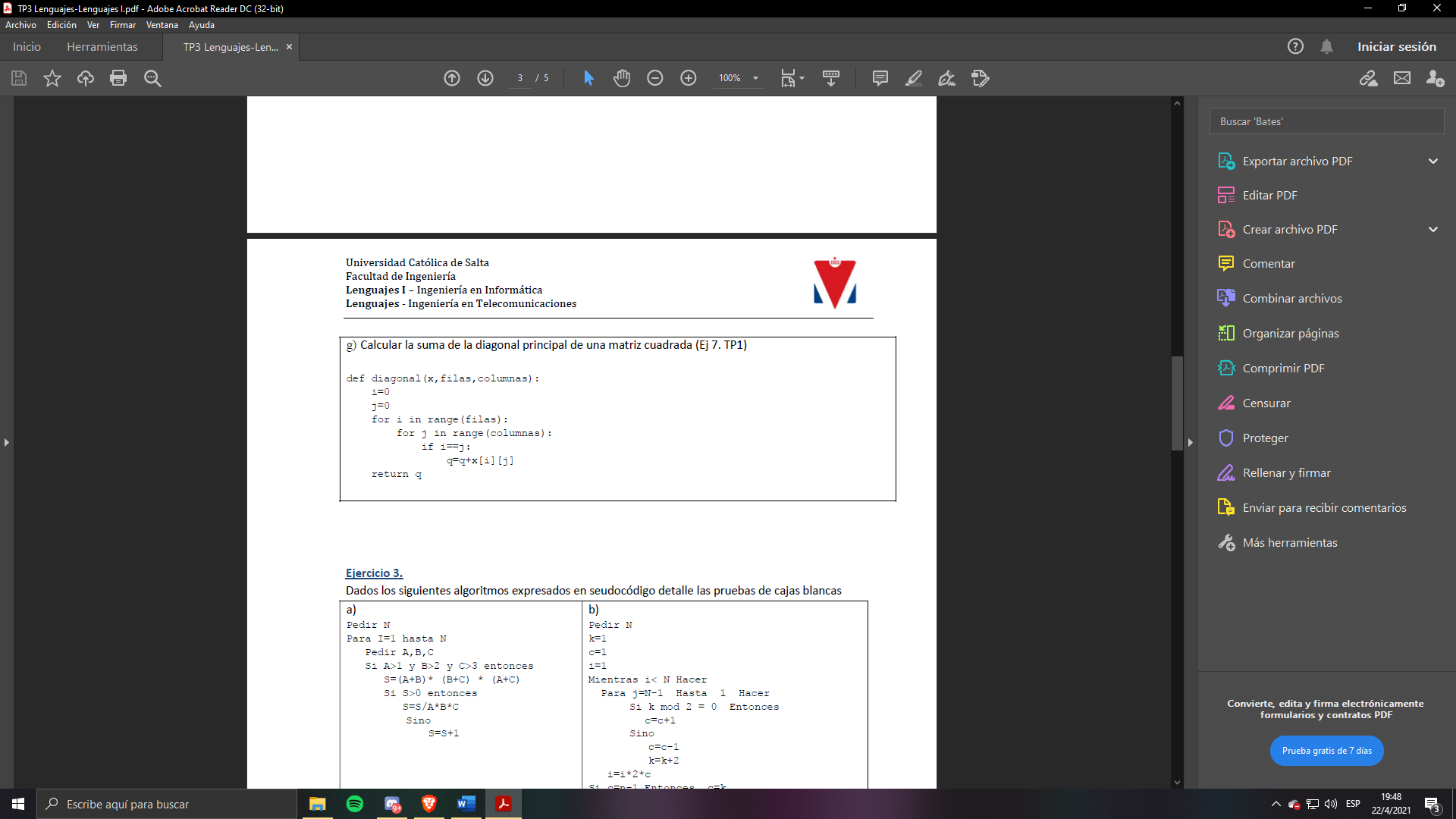
for i in range filas:

matriz[i][0]=input (“Ingrese el electrodoméstico: ”)

matriz[i][1]=input (“Ingrese la marca: ”)

matriz[i][2]=input (“Ingrese el precio: ”)

matriz[i][3]=input (“Ingrese el stock: ”)



g) Calcular la suma de la diagonal principal de una matriz cuadrada

Def diagonal(x):

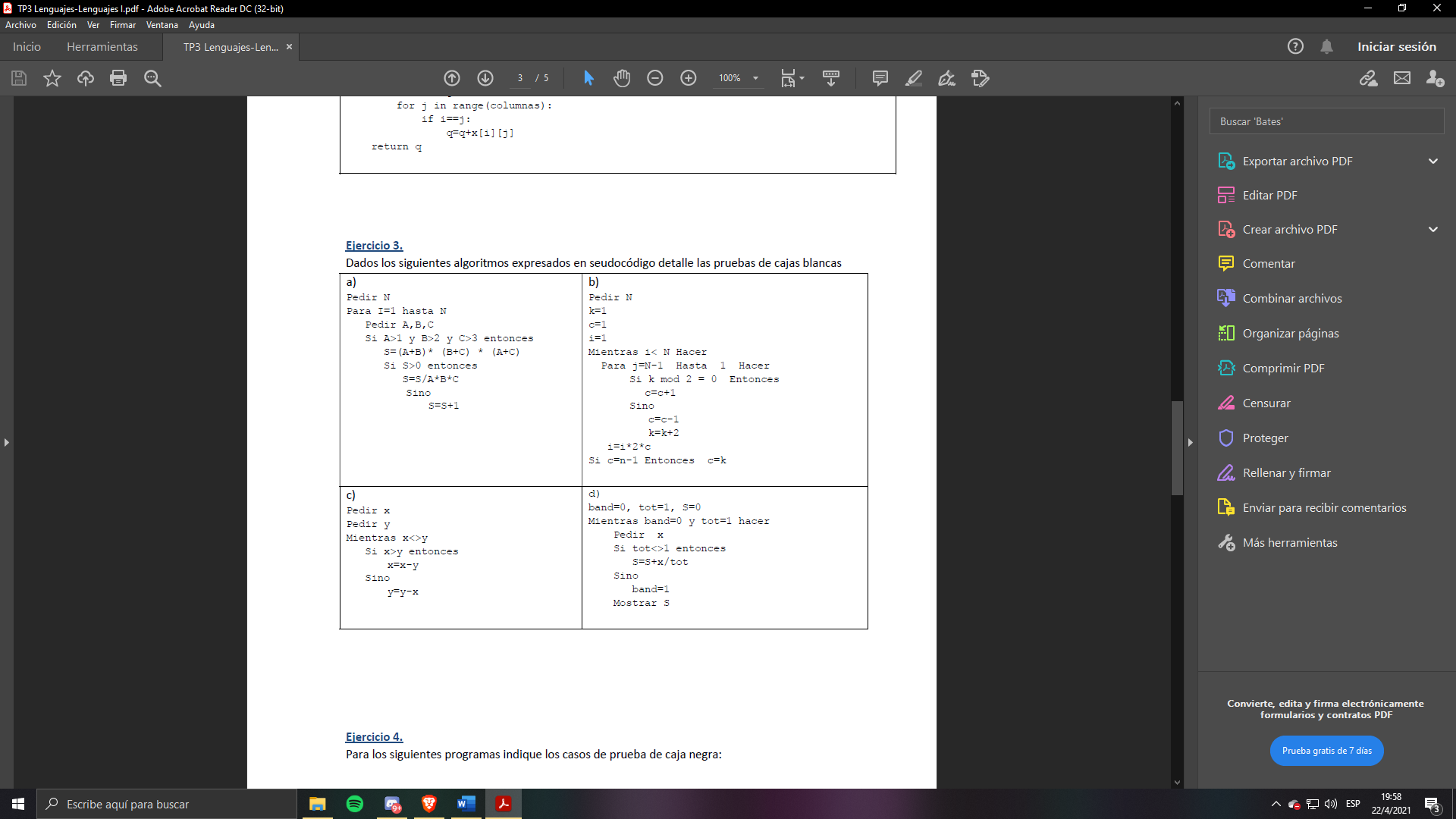
q=0

for i in range (len(x)):

q=q+x[i][i]

return q

Ejercicio 3: Dados los siguientes algoritmos expresados en seudocódigo detalle las pruebas de cajas blancas



a)

Pedir N

Para I=1 hasta N

Pedir A,B,C

Si A>1 y B>2 y C>3 entonces

S=(A+B)\* (B+C) \* (A+C)

Si S>0 entonces 🡨Se usa siempre, no hay un caso falso nunca

S=S/A\*B\*C

Sino 🡨No hay una condición “Si” anterior

S=S+1

b)

Pedir N

k=1

c=1

i=1

Mientras i< N Hacer 🡨 No cumple ninguna función

Para j=N-1 Hasta 1 Hacer 🡨 siempre va a tomar el valor N-1 todas las veces del ciclo i<N

Si k mod 2 = 0 Entonces 🡨 Se va a ejecutar siempre ya que k no puede cambiar

c=c+1

Sino 🡨 no se ejecuta nunca

c=c-1

k=k+2

i=i\*2\*c

Si c=n-1 Entonces c=k 🡨 “n” no existe, no se va a ejecutar nunca

c)

Pedir x

Pedir y

Mientras x<>y 🡨Funciona solo en números mayores a 0

Si x>y entonces

x=x-y

Sino

y=y-x

d)

band=0, tot=1, S=0

Mientras band=0 y tot=1 hacer 🡨”tot” no es necesario ya que no cambia

Pedir x

Si tot<>1 entonces 🡨 No entra nunca, “tot” no cambia

S=S+x/tot

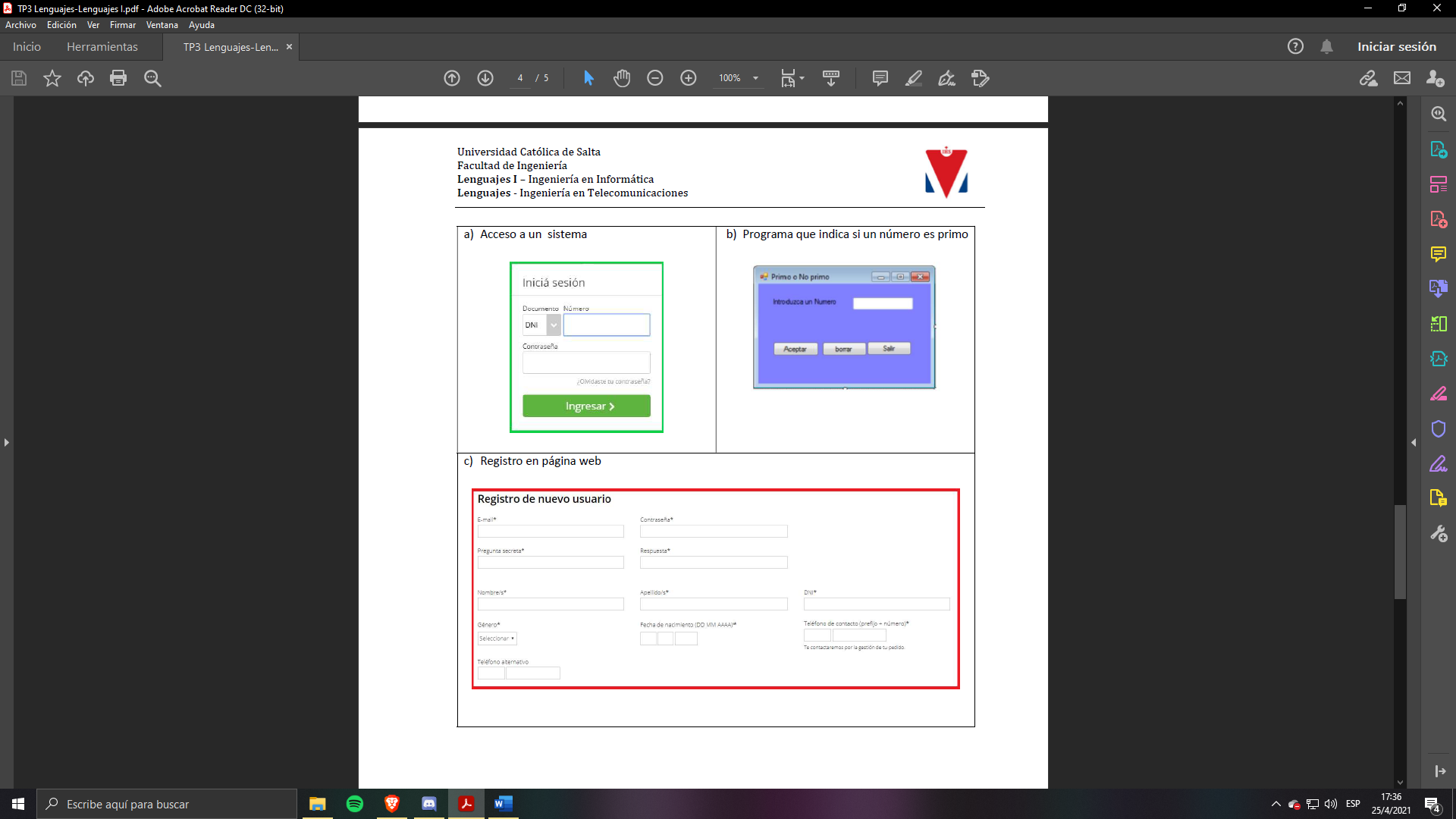
Sino

band=1

Mostrar S

Ejercicio 4.

Para los siguientes programas indique los casos de prueba de caja negra:



a)

DNI:

Valores típicos

Valores fuera de rango (muy grandes o muy chicos)

Letras y símbolos

Contraseña:

Cantidad máxima de dígitos

Cantidad mínima de dígitos

Símbolos y espacios

b)

Valores típicos

Valores muy grandes o muy chicos (negativos)

Letras y símbolos

c)

E-mail:

Valores típicos

Símbolos (Excluyendo @) y números

Direcciones de correo validas (Hotmail, Gmail, Yahoo)

Máximo número de dígitos y mínimo

Espacios

Contraseña:

Cantidad máxima de dígitos

Espacios

Cantidad mínima de dígitos

Símbolos

Pregunta secreta:

Si la pregunta es coherente

Símbolos y Números

Cantidad máxima y mínima de dígitos

Respuesta:

Símbolos y Números

Cantidad máxima y mínima de dígitos

Nombre y Apellido:

Valores típicos

Símbolos y Números

Cantidad máxima y mínima de dígitos

DNI:

Valores típicos

Valores fuera de rango (muy grandes o muy chicos)

Letras y símbolos

Espacios

Fecha de nacimiento:

Letras y símbolos

Cantidad máxima y mínima de dígitos

Espacios

Número de teléfono:

Valores típicos

Espacios

Letras y Símbolos

Cantidad máxima y mínima de dígitos

Que el numero sea existente

Ejercicio 5

Obtenga los tiempos de ejecución del peor caso para cada uno de los procedimientos/programas

siguientes escritos en pseudocódigo:

a) Suma de matrices

Proc Suma\_Mat(A, B, n, m, C)

Para i=1 hasta n 🡨 O(m)\*n 🡨 O(m\*n)

Para j=1 hasta m 🡨 O(1)\*m 🡨 O(m)

C[i,j]= A[i,j]\*B[i,j] 🡨 O(1)

b) Proc Prod\_Mat(n, A,B, C)

Para i=1 hasta n 🡨 O(n2) \*n 🡨 O(n3)

Para j=1 hasta n 🡨 O(1) \*O(n) \*O(1) \*O(n) 🡨 O(n2)

suma=0 🡨 O(1)

Para k=1 hasta n 🡨 O(1)\*n 🡨 O(n)

suma= suma+A[i,k]\*B[k,j] 🡨O(1)

C[i, j]= suma 🡨 O(1)

c) Proc Misterio (n)

Para i=1 to n 🡨 O(n)\*n 🡨 O(n2)

si i mod 2=0 entonces 🡨 O(n)

Para j=i hasta n 🡨 O(1)\*n 🡨 O(n)

x=x+1 🡨 O(1)

Para j=1 hasta n 🡨 O(1)\*n 🡨 O(n)

y= y+1 🡨 O(1)

d) Programa

Func SumaUno(x) 🡨 O(1)

SumaUno= x+1 🡨 O(1)

#----------------------------------------

Proc Misterio(n) 🡨 O(n2)

(\* ver código del ejercicio anterior \*) 🡨 O(n2)

#----------------------------------------

#prog. ppal

i=1 🡨 O(1)

mientras i<= 3 🡨 3\* O(1) \*O(1) \* O(1) \* O(k2) 🡨 O(3k2)

pedir k 🡨 O(1)

k= SumaUno(k) 🡨 O(1)

Misterio(k) 🡨 O(k2)

i= i+1 🡨 O(1)