**Objetivo de la materia:**

La asignatura de métodos numéricos aporta al perfil del ingeniero mecatrónico la capacidad de aplicar herramientas matemáticas, computacionales y métodos experimentales en la solución de problemas para formular modelos, analizar procesos y elaborar prototipos.

La asignatura requiere que el alumno haya cursado cálculo diferencial y cálculo integral, así como Programación Básica.

**Introducción a Python**

Python es un lenguaje de Programación orientados a objetos que fue desarrollado a finales de los 80’s como un lenguaje de tipo script. Python tiene el objetivo de ser un lenguaje para el desarrollo rápido de aplicaciones en ingeniería. Los programas de Python no son compilados, si no que trabajan bajo un intérprete.

Python tiene algunas ventajas sobre otros lenguajes de programación como:

* Python es un software de código abierto, lo que quiere decir que es libre y gratuito y es concluida en todas las computadoras con distribuciones Linux o Mac.
* Python puede correr sus scripts en todas las plataformas (Windows, Linux, Macos) sin tener que modificar ninguna parte del programa.
* Python es fácil de aprender y produce un código más amigable que otros lenguajes.
* Python y sus extensiones son fáciles de instalar.

**Introducción:**

En este capítulo, dos estrellas de solución de citas algebraicas lineales de conocimientos desconocidos. Es, con mucho, el tema más largo y posiblemente el más importante del libro. Hay una buena razón para su importancia: es casi imposible llevar a cabo análisis numéricos de cualquier tipo sin encontrar ecuaciones simultáneas. Además, los conjuntos de ecuaciones que surgen de problemas físicos a menudo son muy grandes y consumen muchos recursos computacionales. Por lo general, es posible reducir los requisitos de almacenamiento y el tiempo de ejecución mediante la explotación de propiedades especiales de la matriz eficiente, como la dispersión (la mayoría de los elementos de una matriz dispersa son cero). Por lo tanto, existen muchos algoritmos dedicados a la solución de conjuntos de ecuaciones grandes, cada uno de ellos adaptado a una forma particular de la matriz de coeficientes (simétrica, en bandas, dispersa, etc.). Una colección bien conocida de estas rutinas es LAPACK: Linear Algebra PACKage, originalmente escrita en Fortran77.1. No podemos discutir todos los algoritmos especiales en el espacio limitado disponible. Lo mejor que podemos hacer es presentar los métodos básicos de solución, complementados por algunos algoritmos de algoritmos de seguridad para las matrices de coeficientes.

**VARIABLES**

En la mayoría de lenguajes el nombre de una variable representa un valor de un determinado tipo guardado en una ubicación de memoria. El valor puede cambiar, pero no el tipo. Esto no pasa en Python ya que sus variables son tapadas dinámicamente. Una sesión del intérprete de Python cuyo pront son 3 símbolos de mayor que (>>>) ilustra esta característica. Un comentario en Python debe comenzar con el signo “#”.

**CADENAS**

Una cadena es una secuencia de caracteres encerrada entre apostrofes, las cadenas son concatenadas con el operador “+” mientras que el operador es usado para extraer una porción de la cadena.

Una cadena puede ser dividida en sus componentes utilizando el comando “Split” los componentes deben de aparecer como elementos en una lista.

**TUPLA**

Una tupla es una secuencia de objetos arbitrarios separados por comas “ , ” y encerrados entre paréntesis “ (,) ”. Si la tupla contiene un solo objeto, una coma al final del objeto será necesaria; por ejemplo x= (2,).

Las tuplas soportan el mismo tipo de operaciones que las cadenas, por ejemplo:

>>> rec = ('Smith', 'John', (6,23,68)) # Esta es una tupla

>>> apellido, nombre, fecha de nacimiento = rec # Desempacando la tupla

>>> print (nombre)

John

> >> año de nacimiento = fecha de nacimiento [2]

>>> print (año de nacimiento)

68

>>> nombre = rec [1] + '' + rec [0]

>>> print (nombre)

John Smith

>>> print (rec [0] : 2])

('Smith', 'John')

**Listas**

Una lista es similar a una tupler, sin embargo, sus elementos y su longitud pueden cambiar. Una lista es identificada por que sus datos están encerrados en corchetes.

**a • append (4.0)**

Se puede determinar la longitud de una lista con el comando “Len”, por ejemplo:

**(listan ant)**

**Print (len(a))**

También se puede **modificar elementos internos en una lista, por ejemplo:**

**Prin (a) … (0.0, 0.5) …**

Las matrices son representadas por listas anidadas, con cada fila siendo un elemento de la lista, por ejemplo, insertar matriz “(a)”.

A=

**Módulos matemáticos**

* Math
* Numpy

**El módulo math**

La mayoría de funciones matemáticas están incluidas en el núcleo de python, pero también pueden importarse. Hay 3 formas de acceder a las funciones en un módulo.

La primera forma es importando todo el módulo utilizando la instrucción:

**from math import\***

(Va llamar los módulos de math)

El uso de este método no es recomendable porque carga módulos y comandos que tal vez no usemos.

Una mejor manera de importar es utilizando el comando:

**from math import func1, func2, …**

**from math import.**

Hay una 3r forma de importar los módulos la cual es:

“import math”.

Un módulo puede ser accesible bajo un alias, por ejemplo, el módulo math puede ser llamado mediante el alias “m”, de la siguiente forma.

**import math as chicharo**

**chicharo. cos (0.15)**

Los contenidos de un módulo pueden ser llamados e impresos con la función

**“dir (math)”** importa en lista comandos de math.

**“Módulo numpy”**

El módulo numpy no está incluido en el módulo de python y debe descargarse e instalarse, además implementa funciones del algebra lineal y funciones de manejo de vectores.

Los arreglos pueden ser creados de diferentes formas. Uno de ellos es el uso de las funciones ARRAY para convertir una lista en un arreglo mediante el comando:

Arra(lista,tipo)

La función arange devuelve una secuencia de números entre 2 límites y con un elemento configurado.

Otra función disponible es la creación de matrices con componente CERO utilizando la función:

**#zeros((dim1,dim2,) tipo)**

El comando ONES crea un arreglo con definiciones del usuario :

**#ones((dim1,dim2,)tipo)**

**Condicionales**

La construcción if si condición: bloquear ejecuta un bloque de declaraciones (que deben tener sangría) si la condición devuelve Cierto. Si la condición devuelve False, se omite el bloque. El if condicional puede ser seguido por cualquier número de construcciones elif (abreviatura de "else if") condición elif: bloquear que funcionan de la misma manera. La cláusula else más: bloquear se puede usar para definir el bloque de sentencias que se ejecutarán si ninguna de las Las cláusulas if-elif son verdaderas. El signo de función de a ilustra el uso de los condicionales.

def sign\_of\_a(a):

if a < 0.0:

sign = ’negative’

elif a > 0.0:

sign = ’positive’

else:

sign = ’zero’

return sign

a = 1.5

print(’a is ’ + sign\_of\_a(a))

**SISTEMA DE ECUACIONES ALGEBRAICAS LINEALES**

**Objetivo:** Resolver ecuaciones simultaneas Ax=b

En este tema procedemos a la solución de N ecuaciones algebraicas lineales con N variables desconocidas. Es un tema muy importante ya que casi es imposible realizar análisis numéricos sin encontrar ecuaciones simultáneas. Además, los conjuntos de ecuaciones de problemas de ingeniería son de forma típica muy grandes y consumen demasiados recursos computacionales. Hay muchos algoritmos dedicados a la solución de grandes conjuntos de ecuaciones, cada uno diseña una forma particular una matriz de coeficientes.

El método que vamos a analizar es el de eliminación de gauss.

**NOTACIÓN**

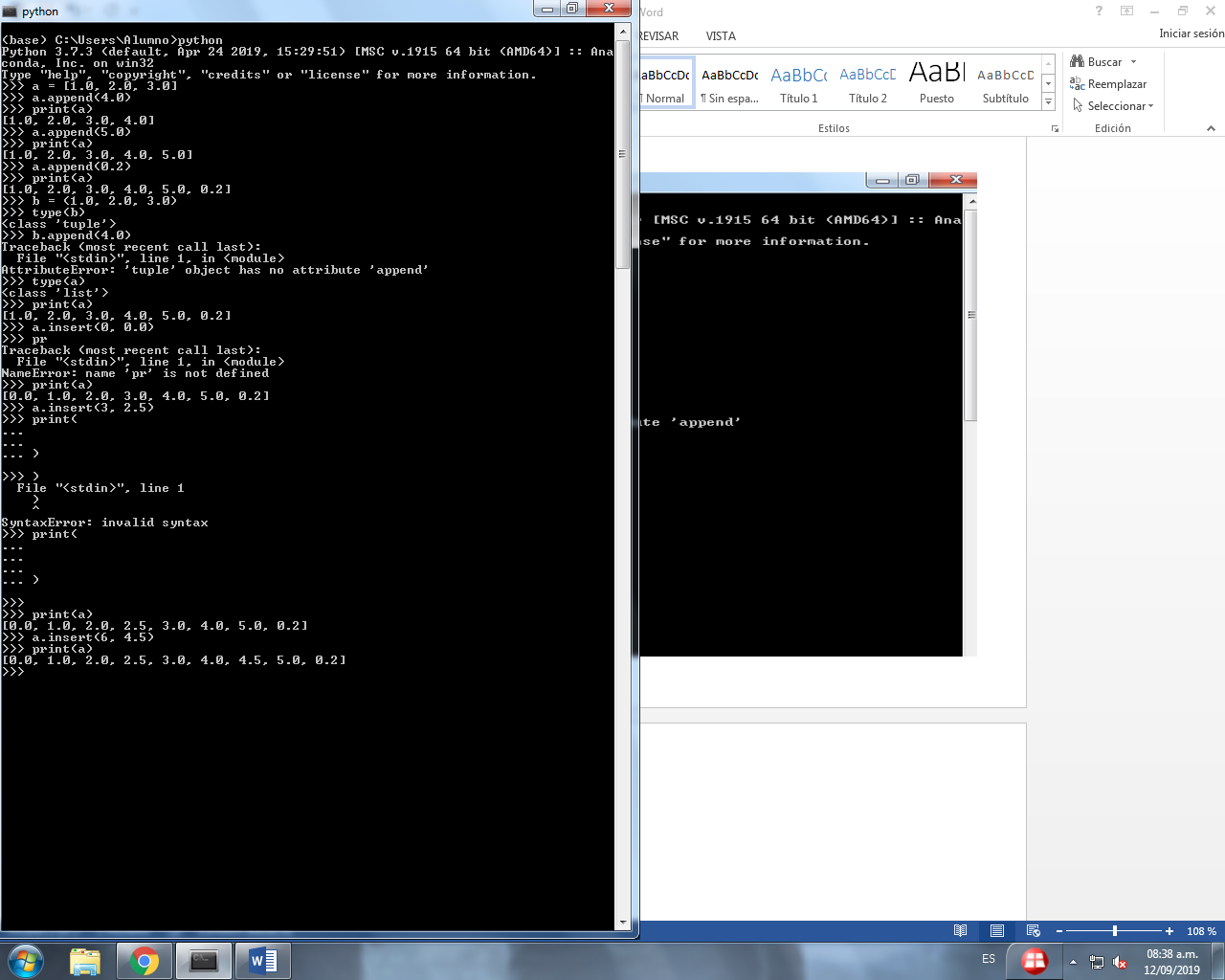
Un sistema de ecuaciones algebraicas tiene la forma:

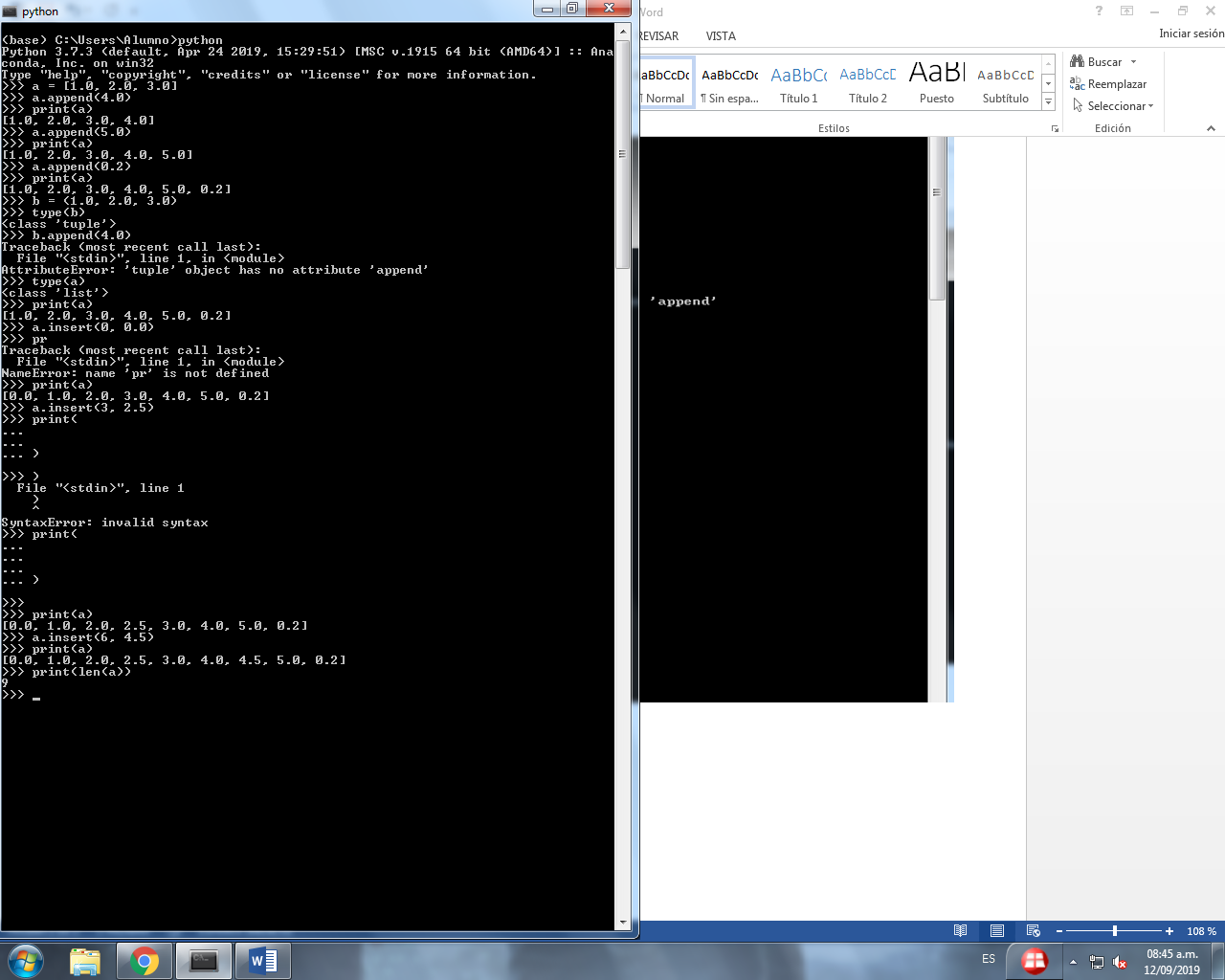
A11 X1 + A12 x2 + … + A 1 nxn = b1

**Ejercicio 1:**

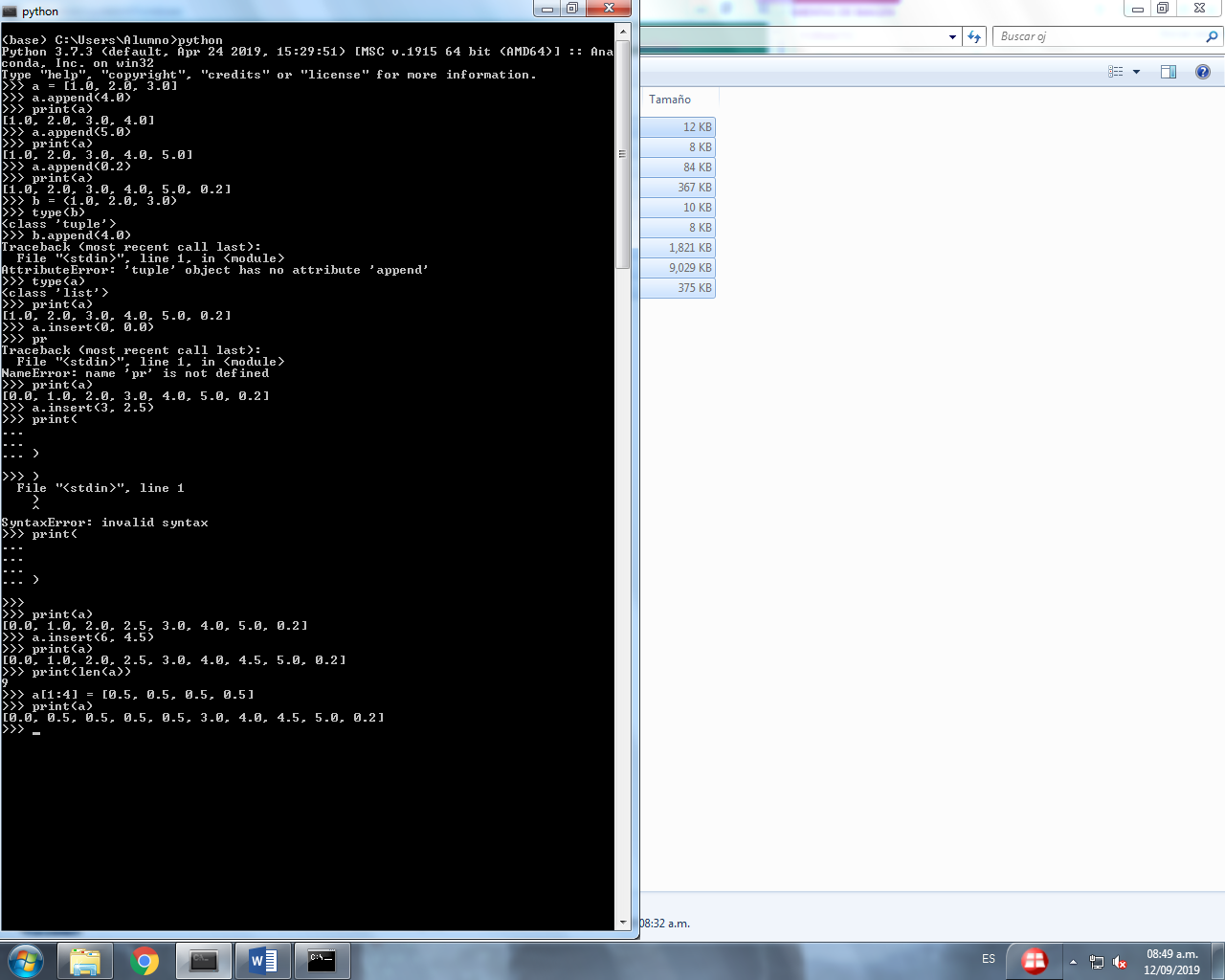
Hacer un programa que obtenga las raíces “x, y, x2”

**Listas**

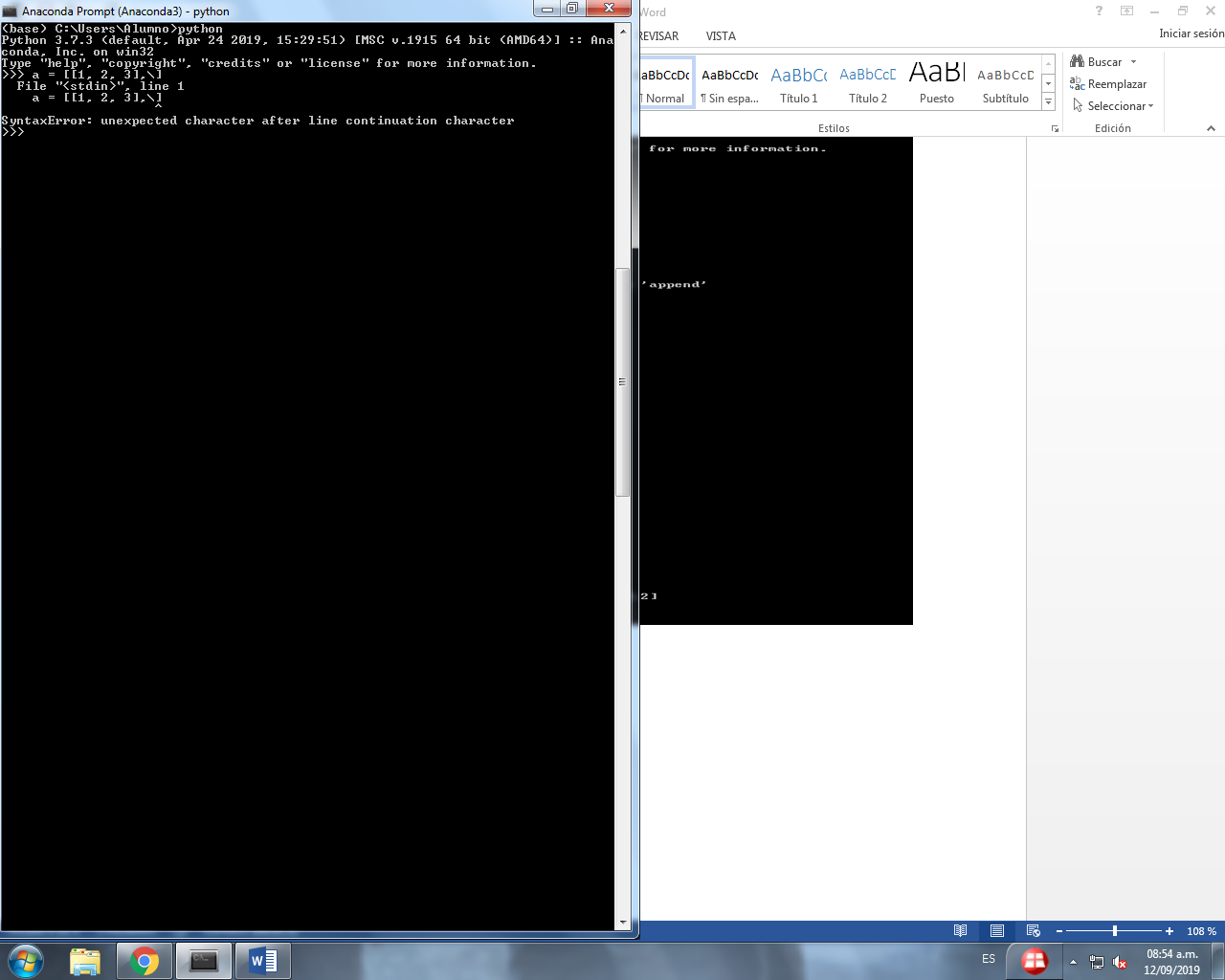


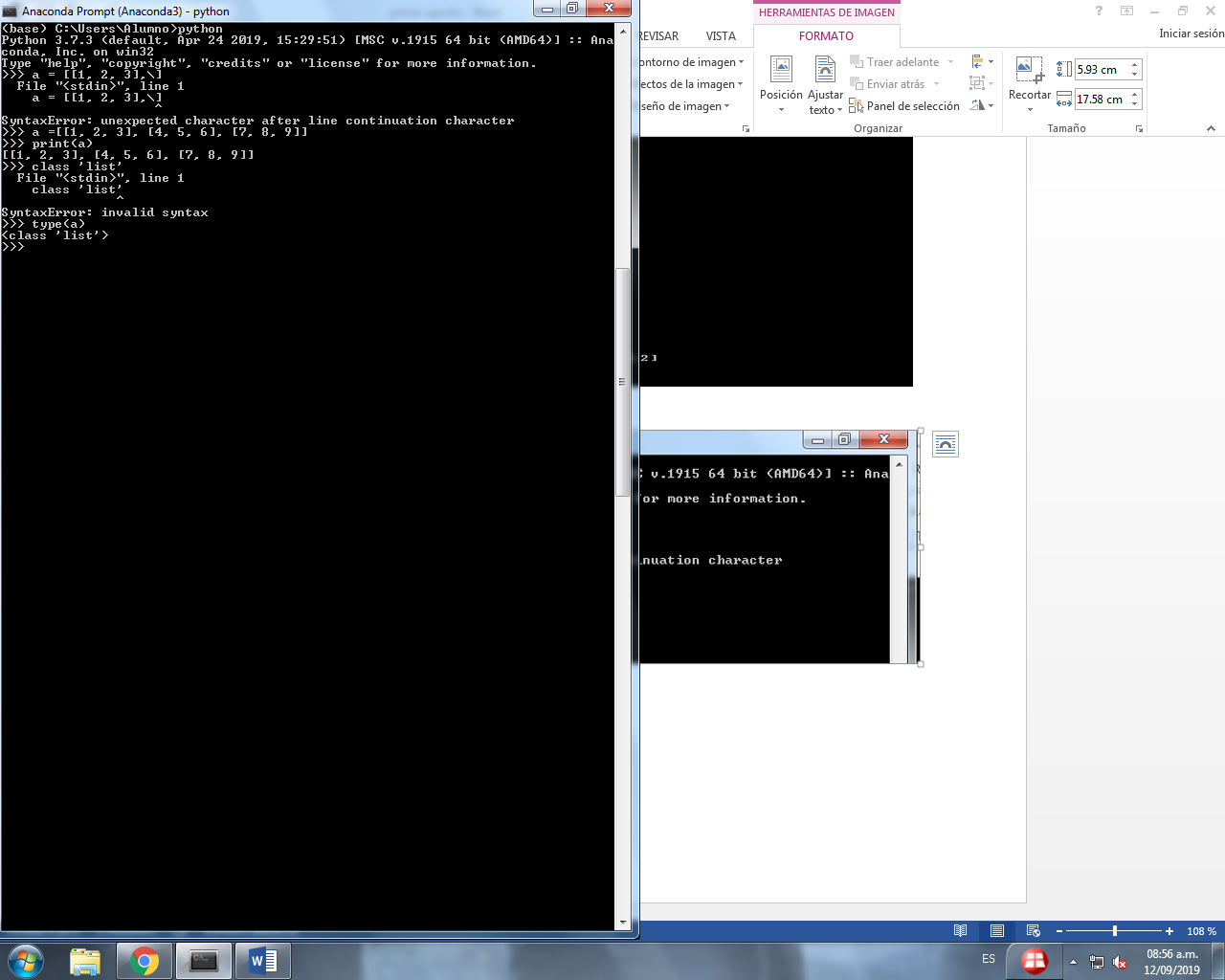


También se puede modificar



Las matrices son:





Un sistema de N ecuaciones lineales con N incógnitas tiene una solución única, esto se sabe si el determinante de la matriz de coeficientes |A| es no singular, esto es |A|≠0 las filas y columnas de una matriz no singular son linealmente independientes en el sentido de que ninguna fila o columna es una combinación lineal de otras filas columnas.

Si el coeficiente de una matriz a es singular las ecuaciones pueden tener un número infinito resoluciones o no tener ninguno.

Tomando como ejemplo el último sistema de ecuaciones la segunda ecuación puede obtenerse multiplicando la primer ecuación por dos, cualquier combinación de x y y se satisface la primera ecuación es también una solución de la primer ecuación.

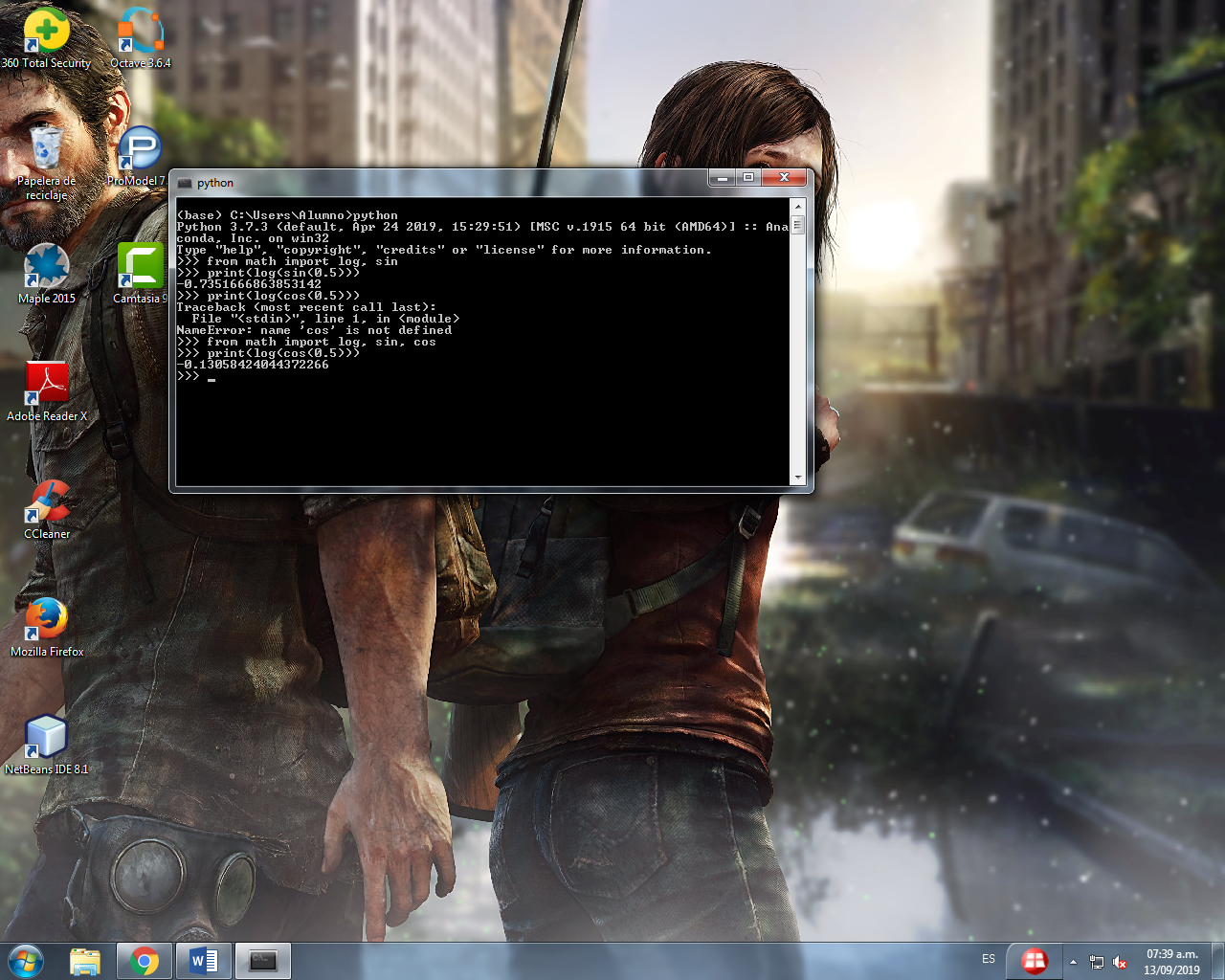
**Módulos matemáticos (numpy & math)**

**El modulo “math”.**

La mayoría de las funciones matemáticas no están incluidas en el núcleo de Python, pero pueden importarse. Hay 3 formas de acceder a las funciones en un modulo.

La primera forma es importando todo el modulo utilizando la instrucción:

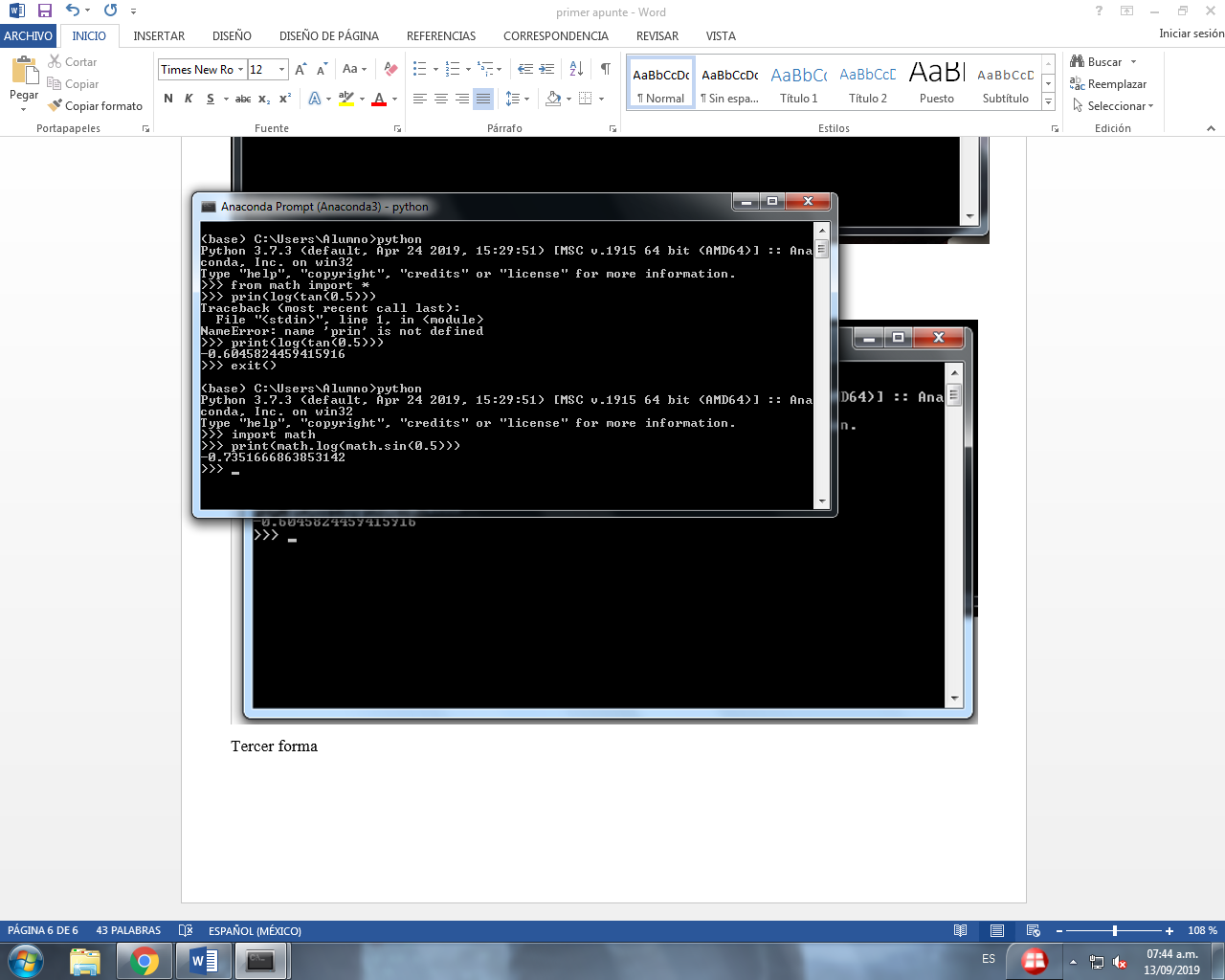
**“From math import”**



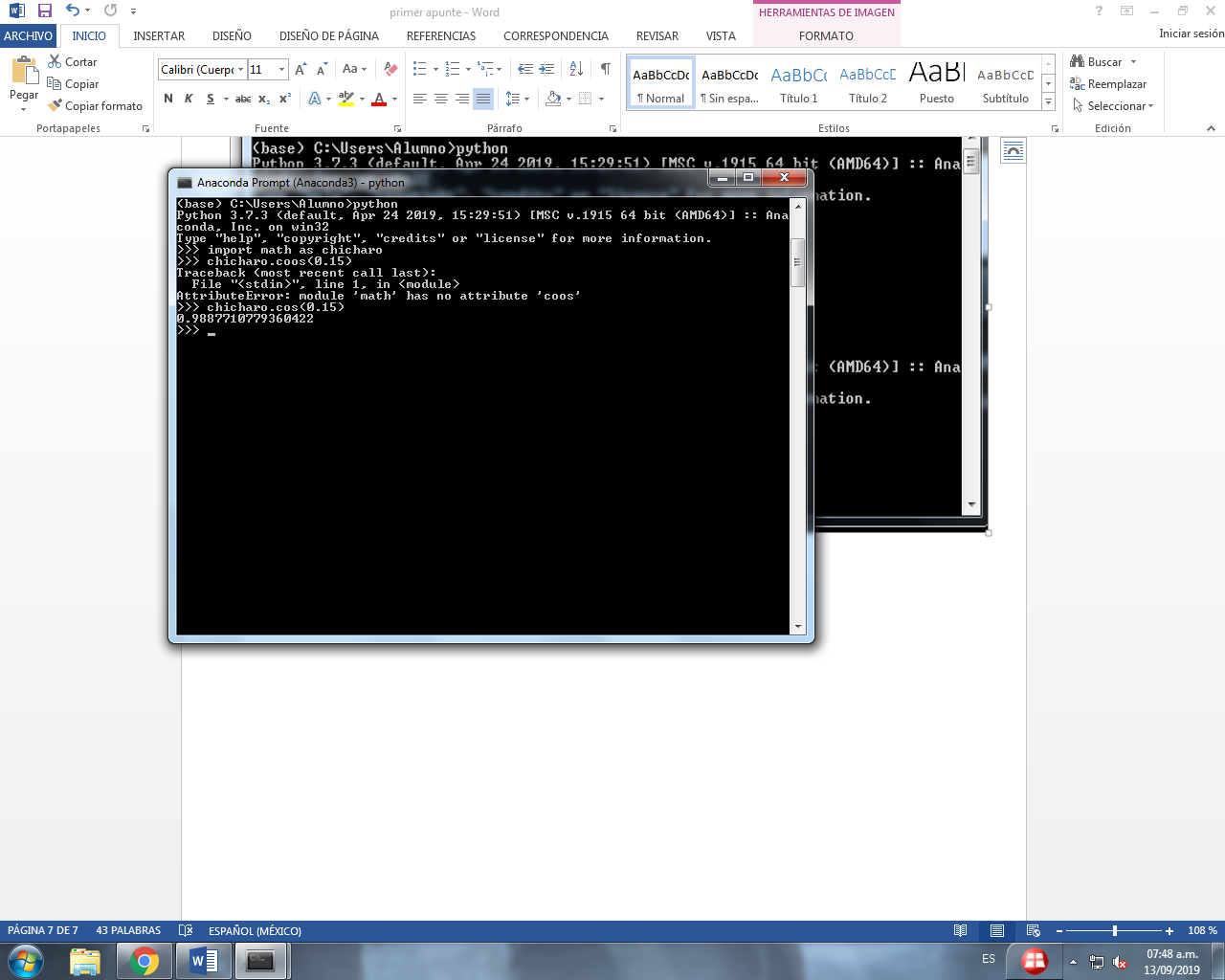
Otro



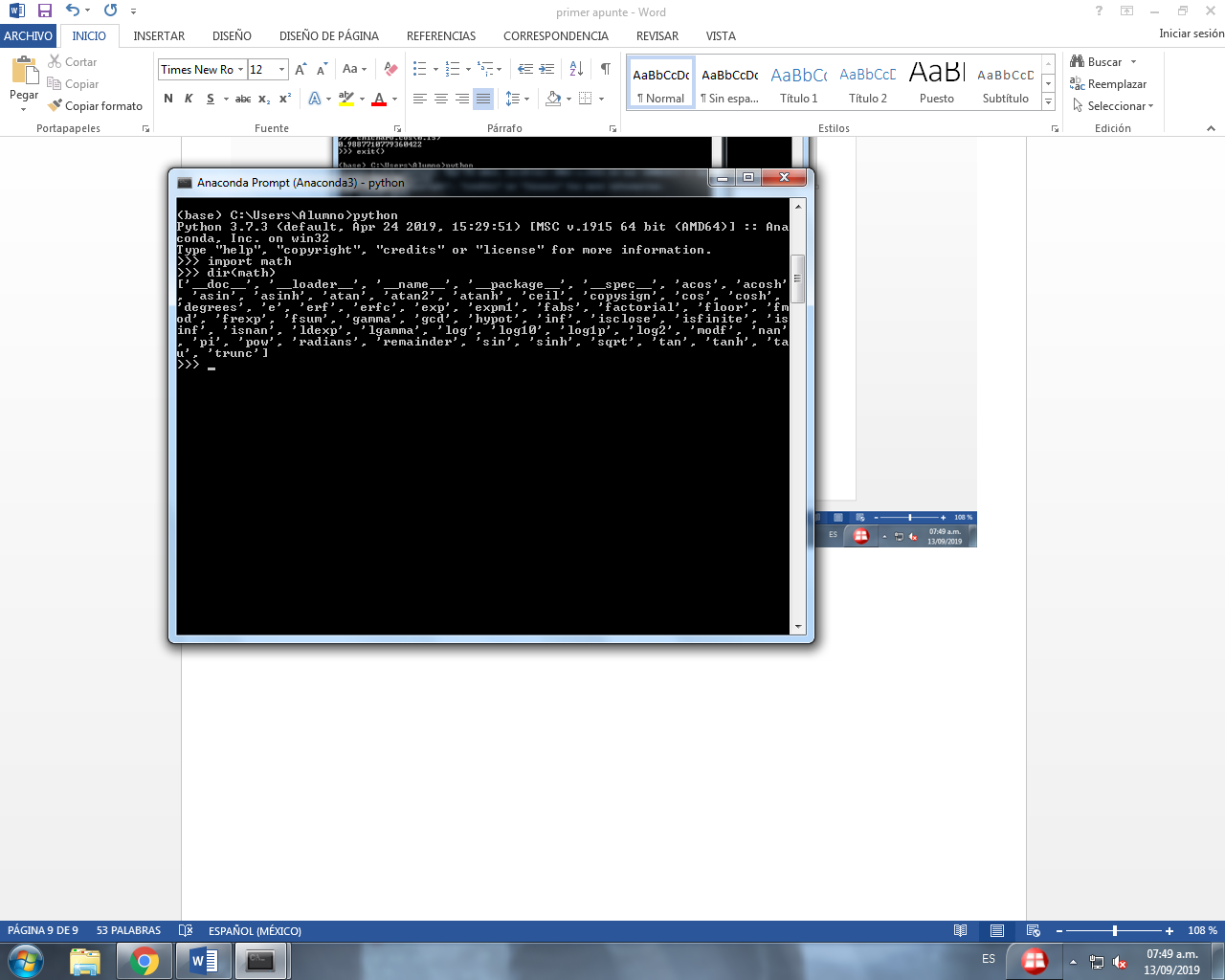
Tercer forma



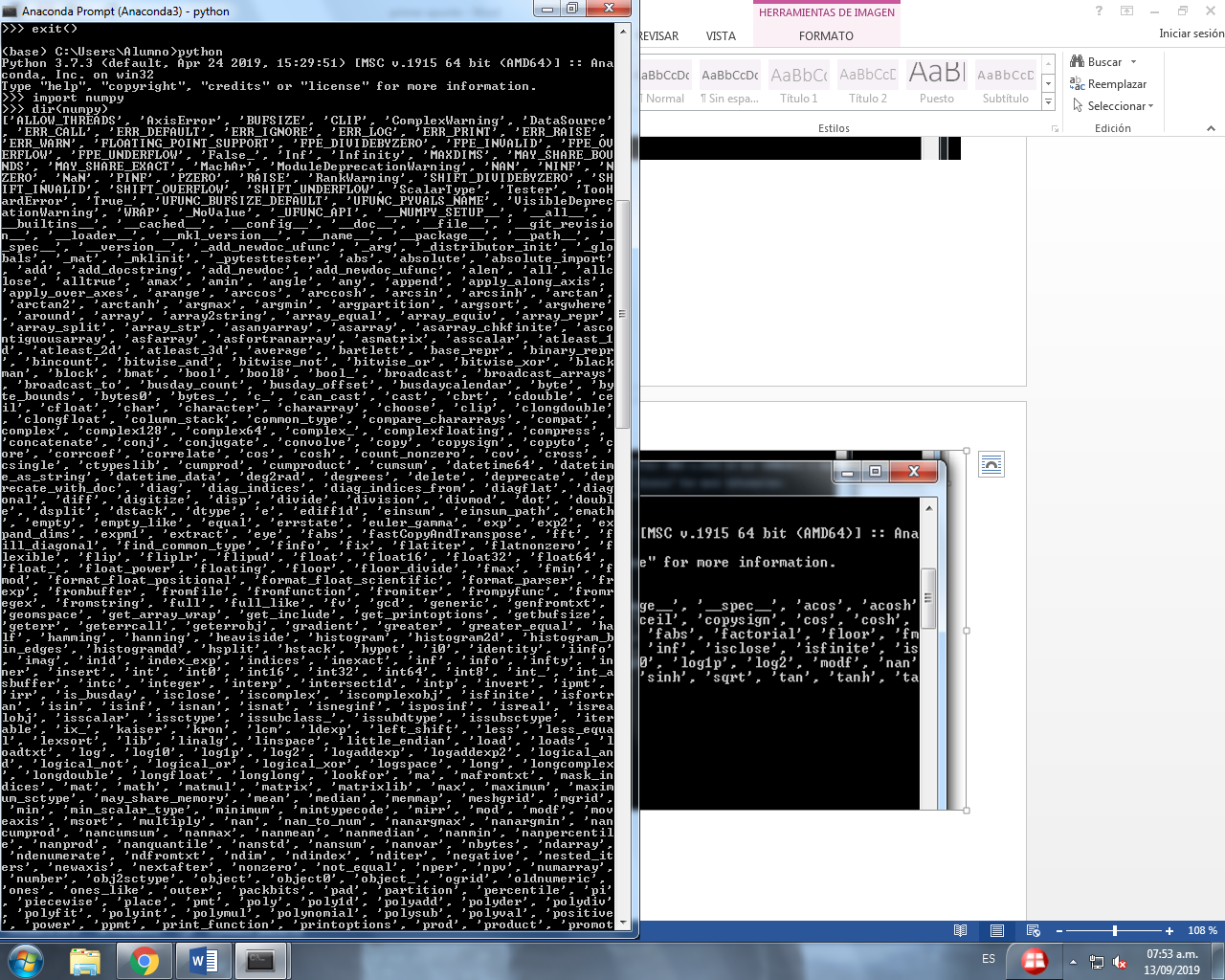
Se puede llamar a math de cualquier forma, por ejemplo:

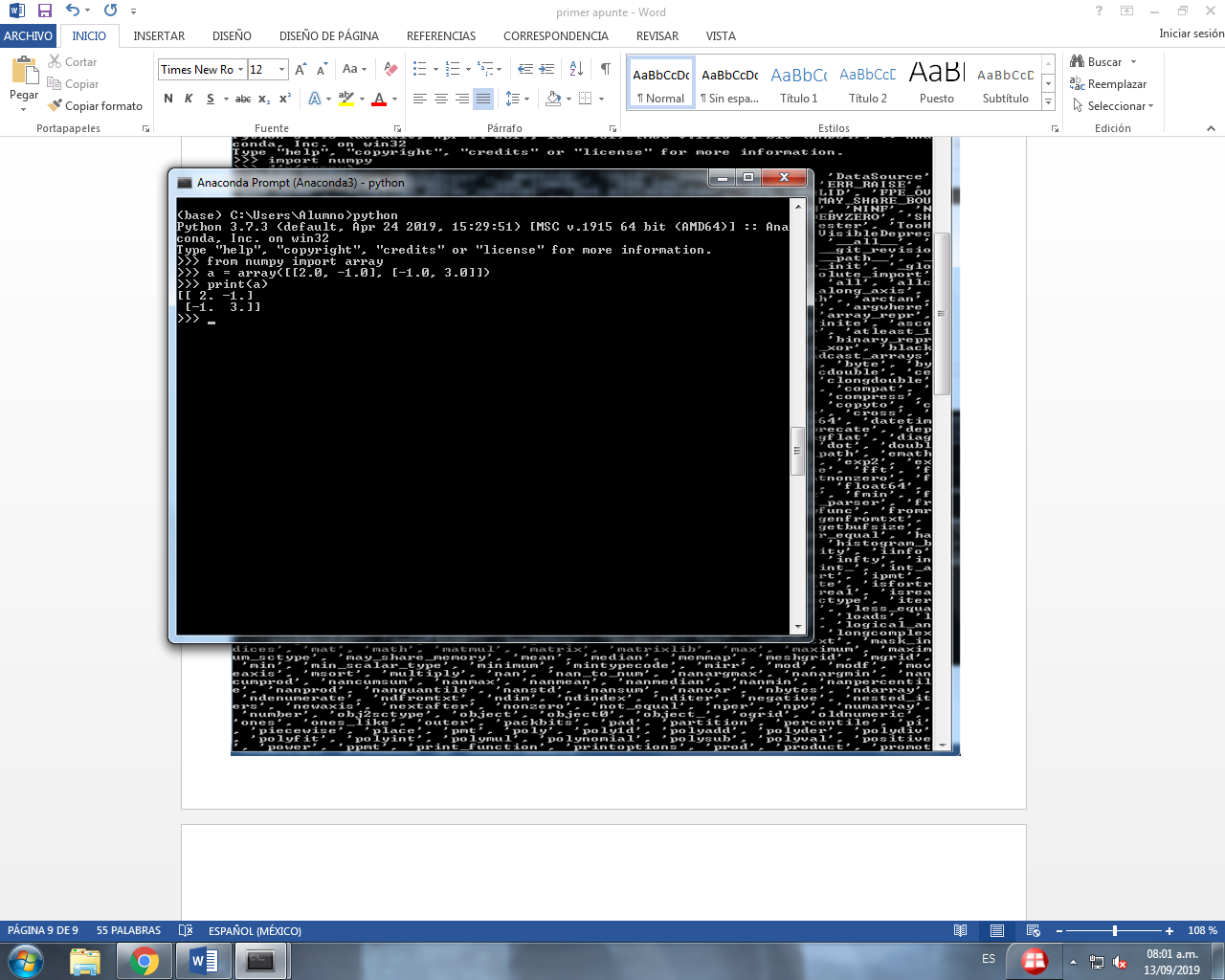


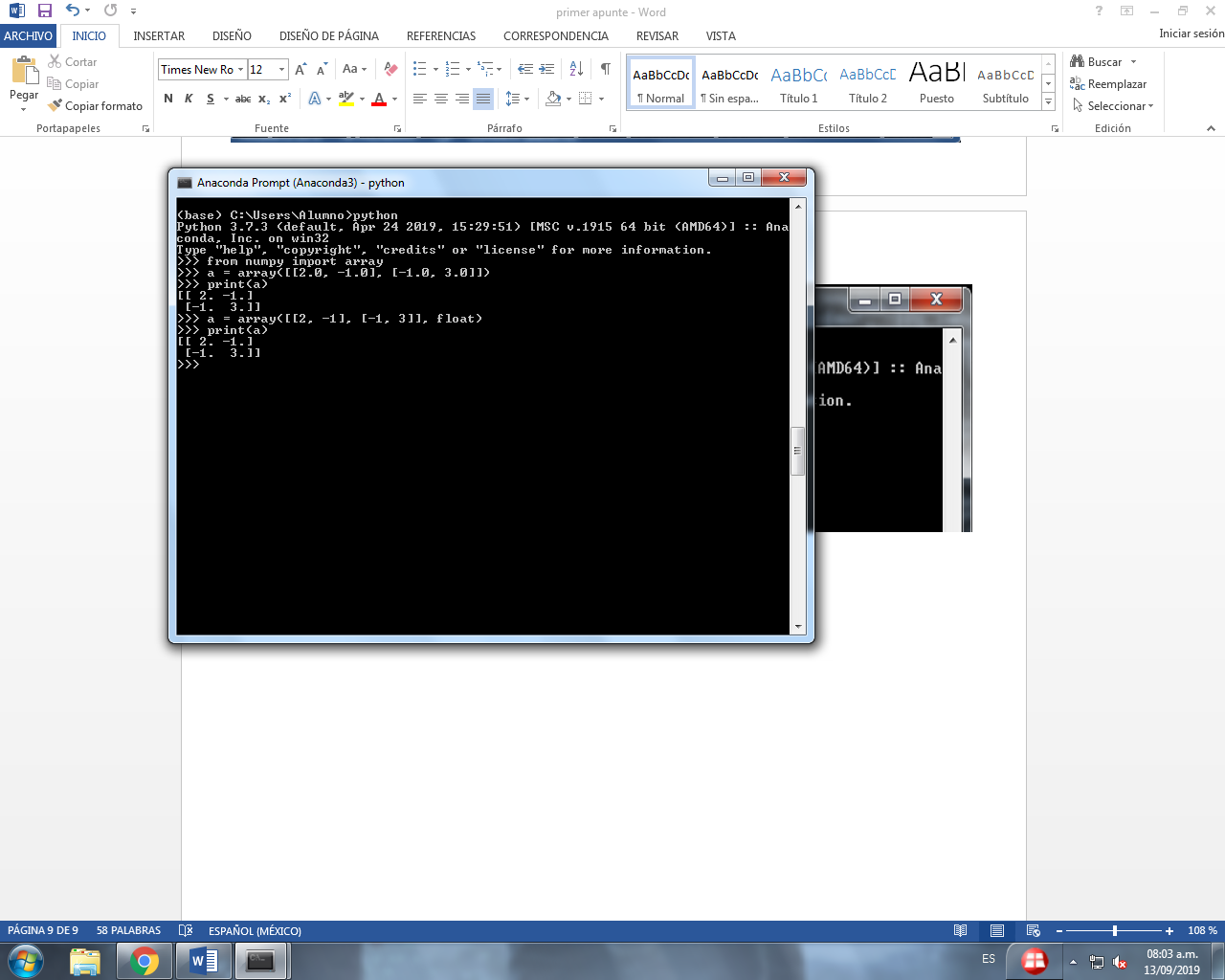
Dir(math)



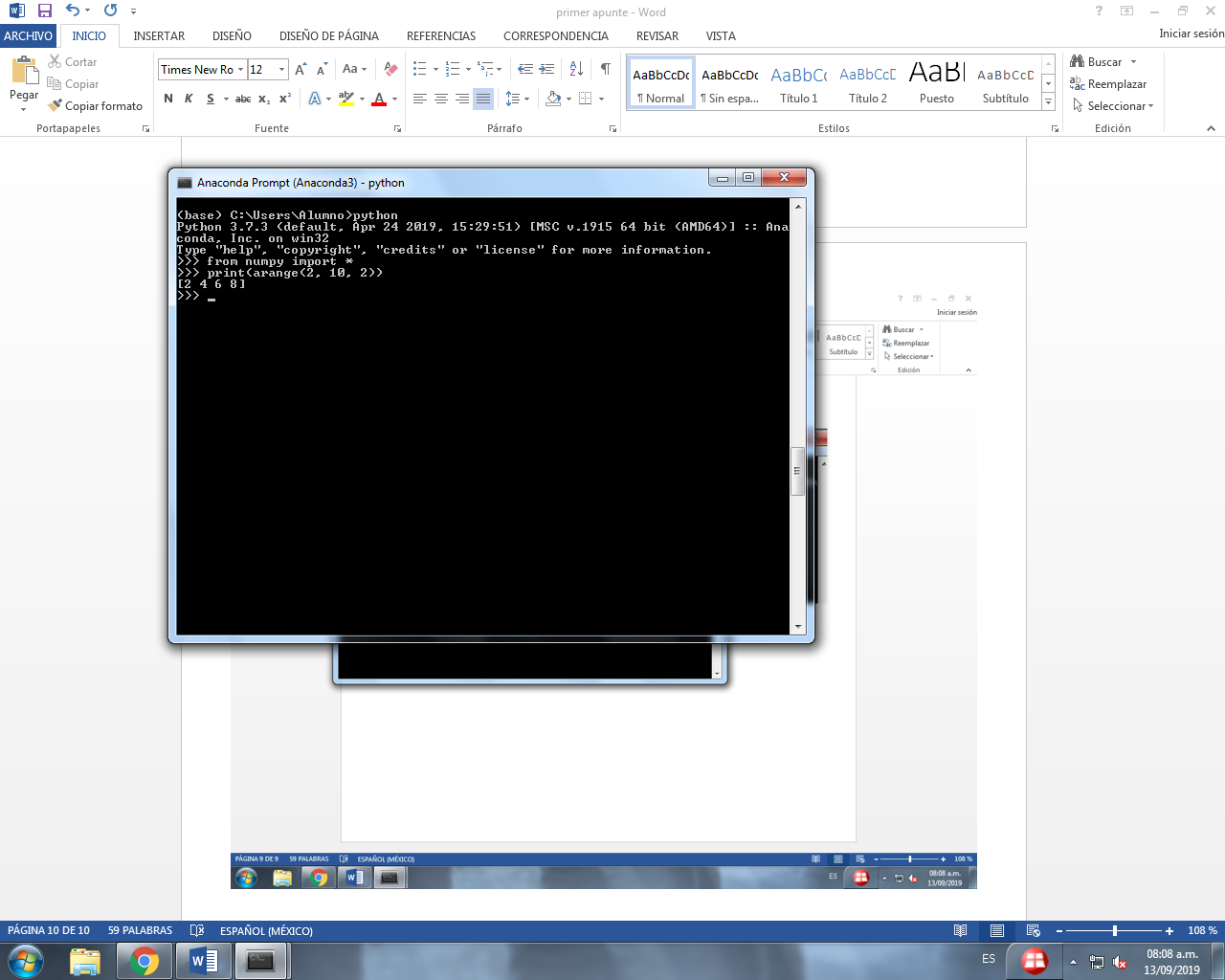
Numpy



Arreglo en Numpy

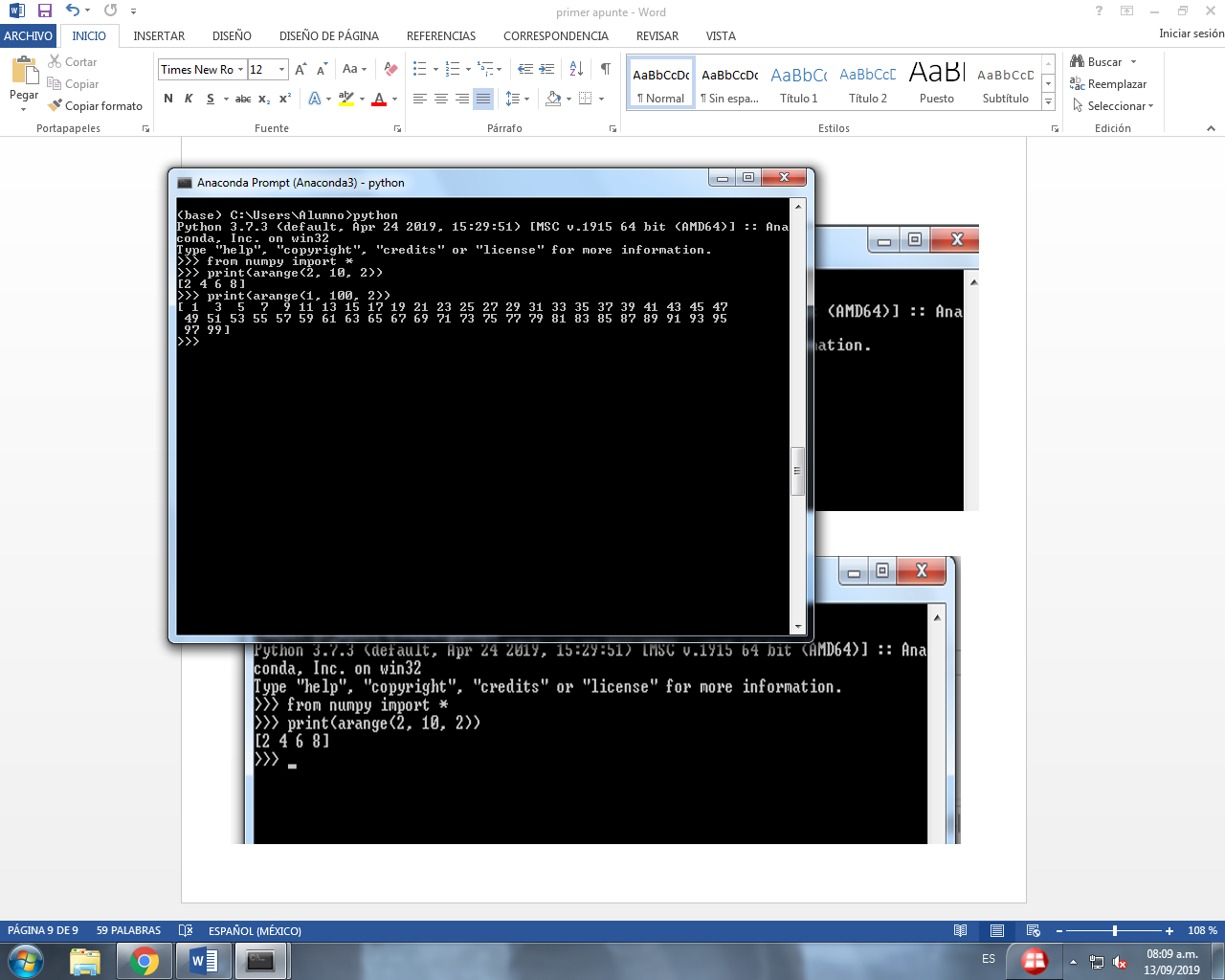


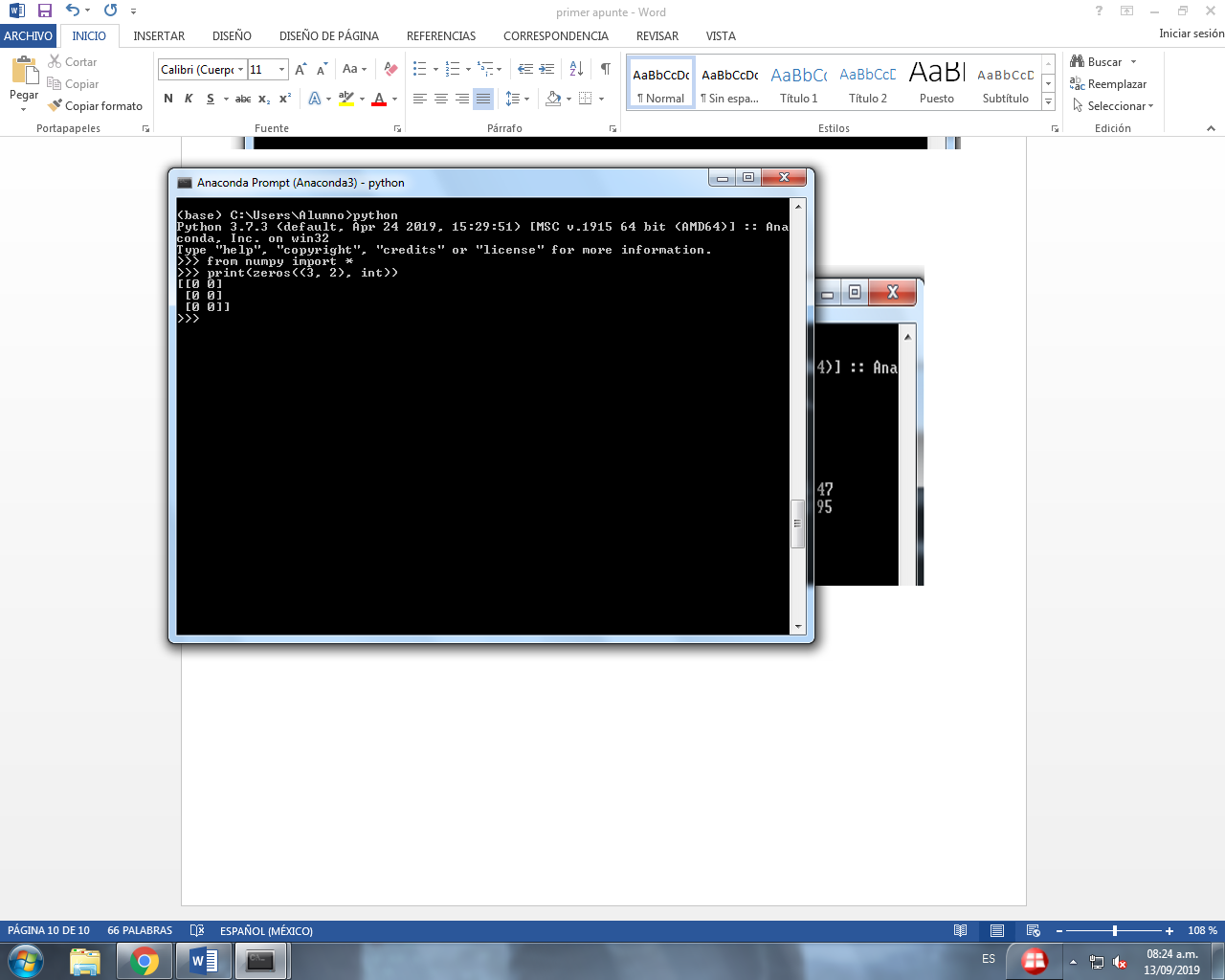
Arange



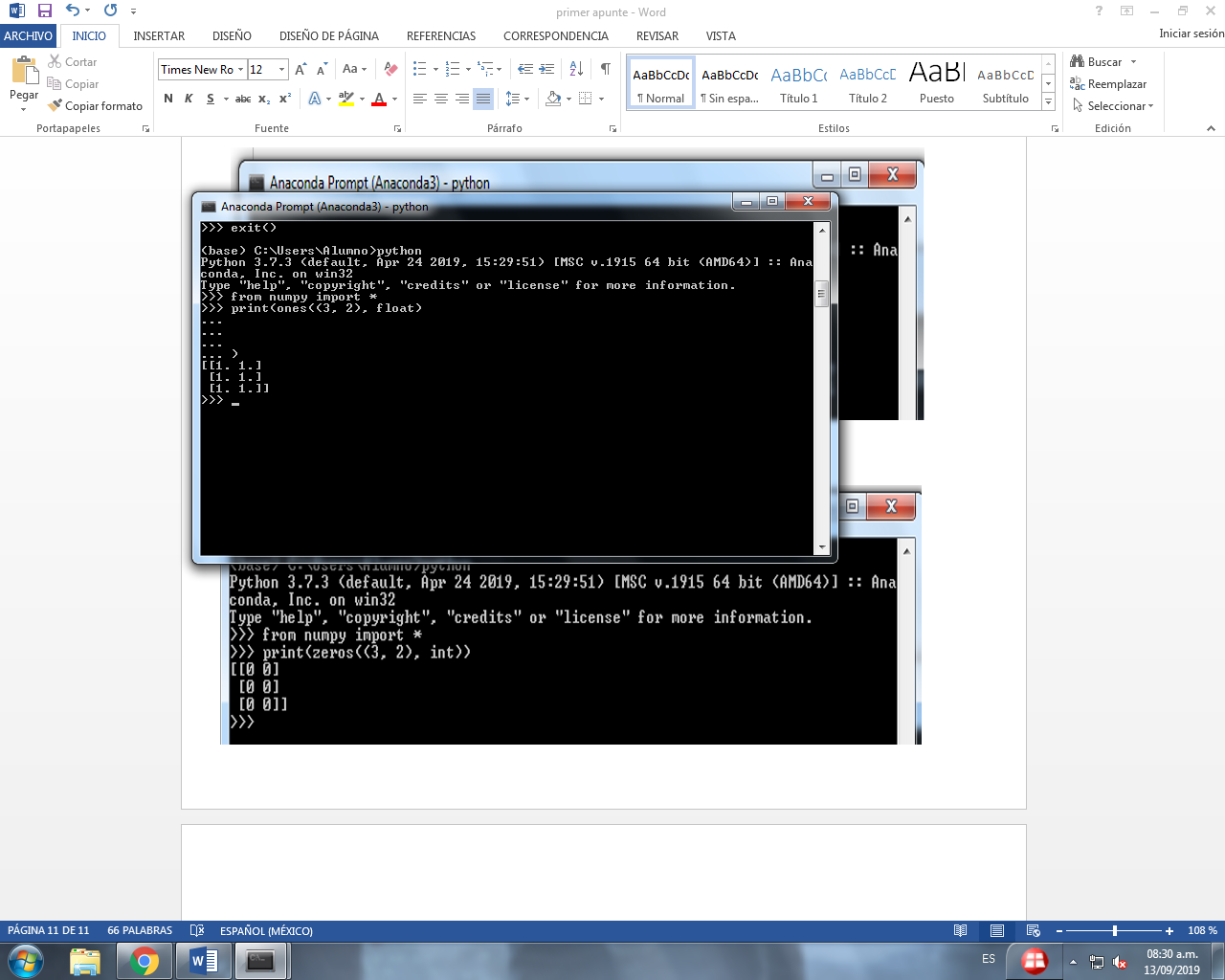
Ejercicio

Números impares 1-99

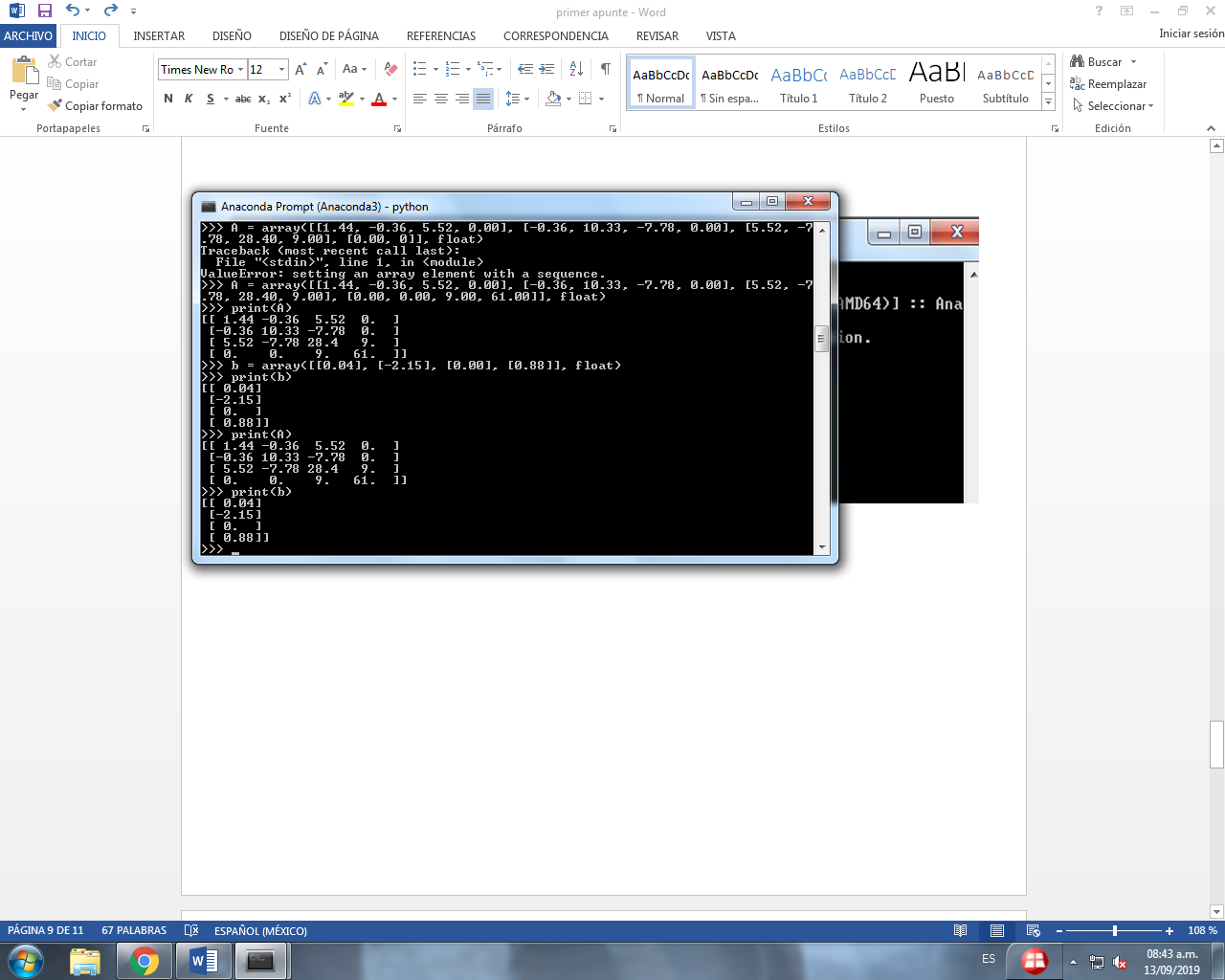


Otra función disponible

Ones



Ejercicio de matiz A, b



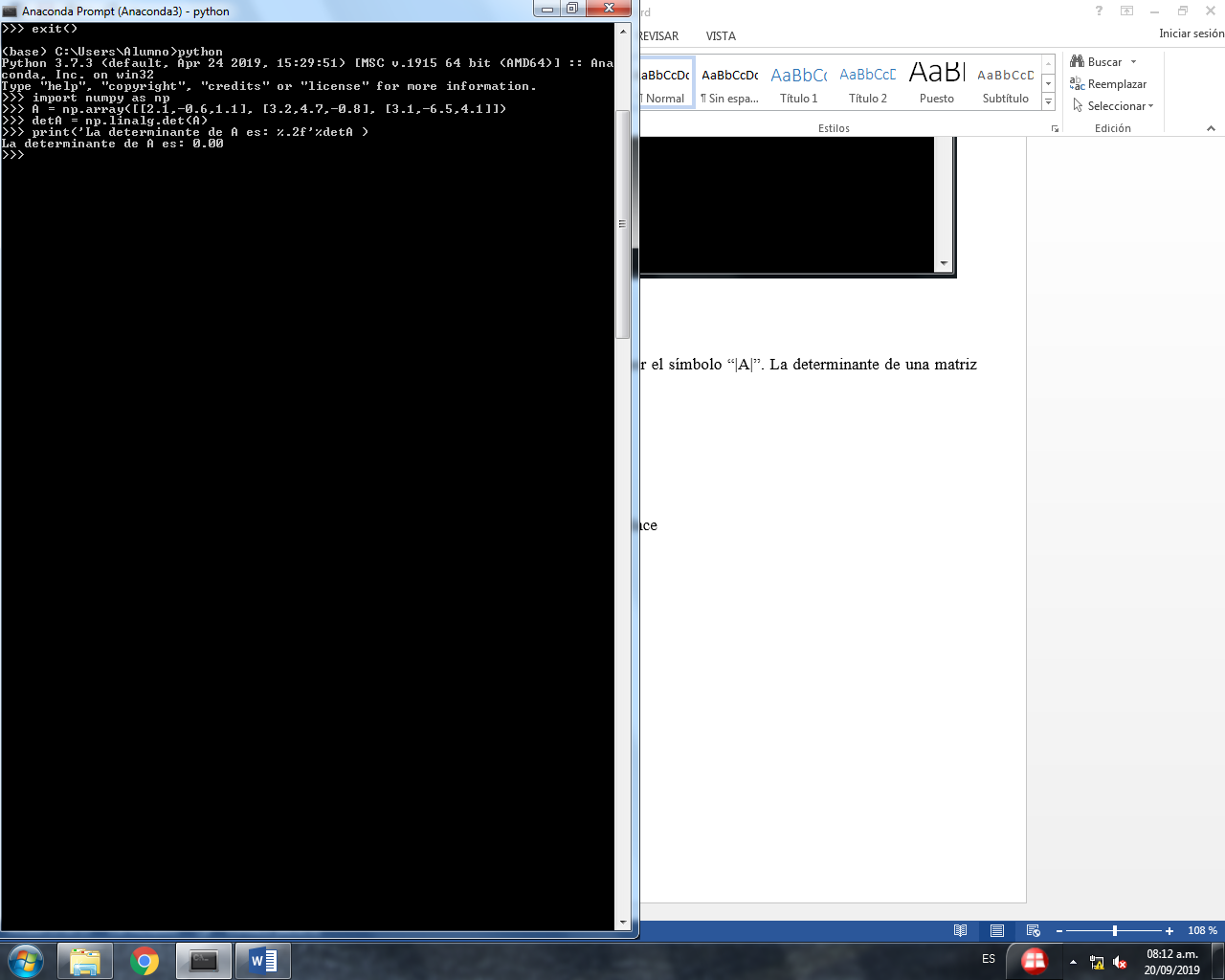


Determinar si la matriz A es singular

La determinante de una matriz cuadrada “A” es denotada por el símbolo “|A|”. La determinante de una matriz 2x2

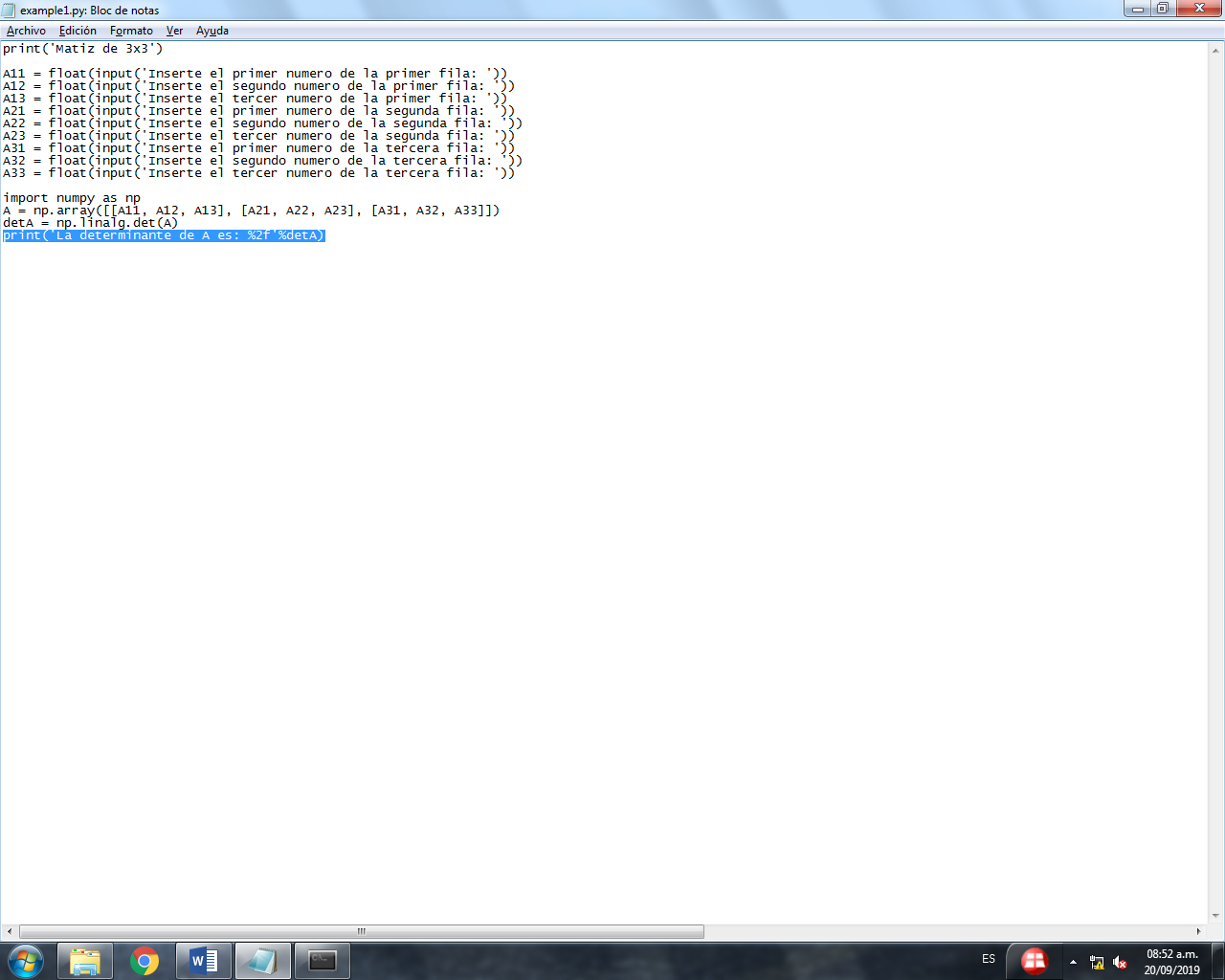
A, |A| = A11\*A22-

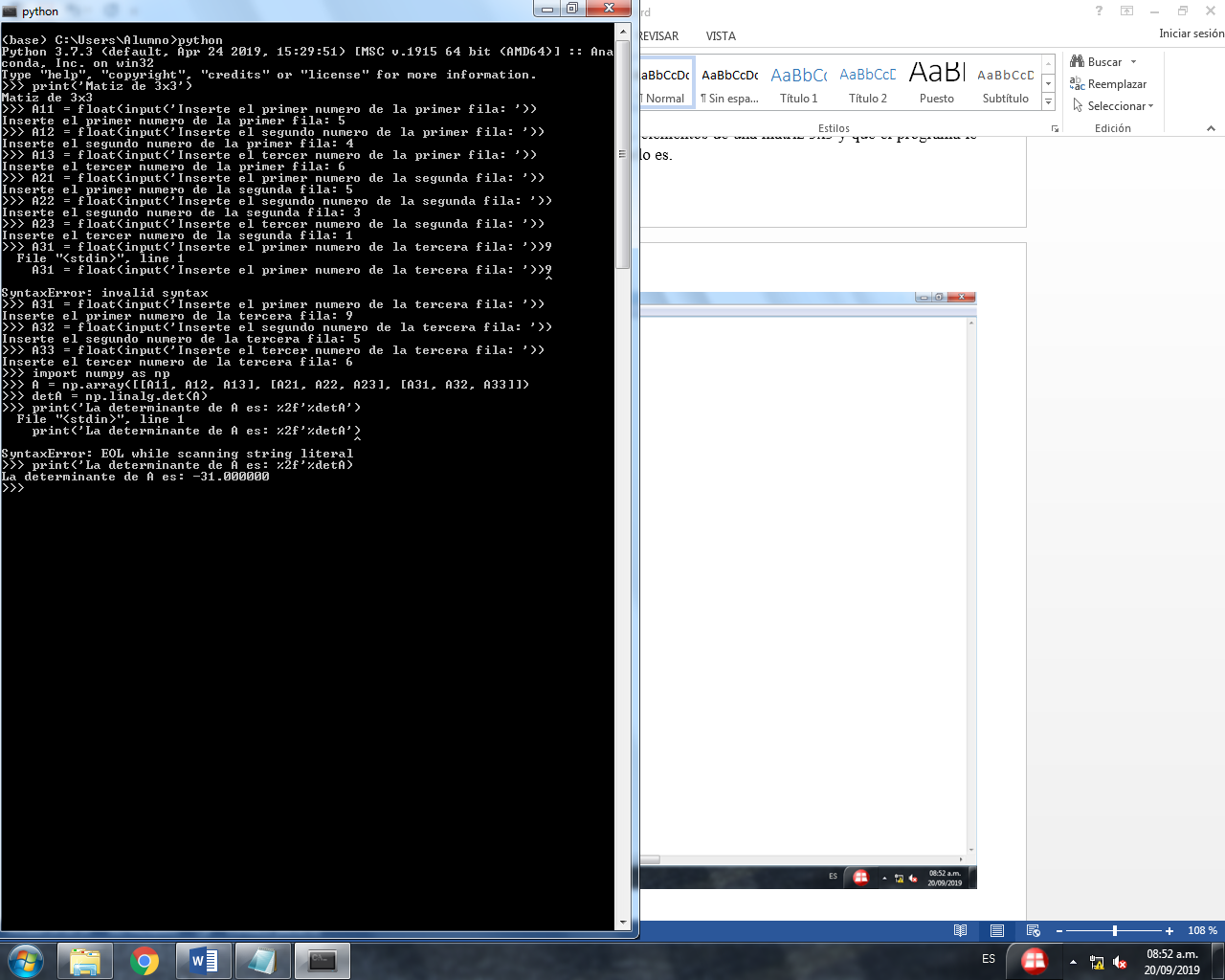
La ecuación anterior es conocida como el desarrollo de Laplace



**Ejercicio.**

Realizar un programa en el que un usuario introduzca los 9 elementos de una matriz 3x3 y que el programa le devuelva al usuario si la matriz que inserta es singular o no lo es.





**Métodos numéricos para la solución de sistemas lineales**

**Directos e Indirectos**

Las ecuaciones algebraicas lineales ocurren en todos los análisis numéricos

**Métodos numéricos**

Existen tres métodos directos

|  |  |  |
| --- | --- | --- |
| Método | Forma Inicial | Forma Final |
| Eliminación de Gauss | Ax = b |  |
| Descomposición de LU | Ax = b |  |
| Eliminación de Gauss-Jordan | Ax = b |  |

**Ejercicio**

Obtener Lx = c

Si x = c =

**Método de eliminación de Gauss**

El método de eliminación de Gauss consiste en dos partes: la fase de eliminación y la fase de sustitución hacia atrás. La función de la fase de eliminación es transformar las ecuaciones en una forma Ux = c, las ecuaciones, entonces son resueltas con sustitución hacia atrás.

**Por ejemplo**

**Fase de eliminación**

La fase de eliminación multiplica una ecuación…

**Fase de sustitución hacia atrás**

Las variables desconocidas pueden ser calculadas usando sustitución hacia atrás

3x3 = 9 x3 = = 3

22

11

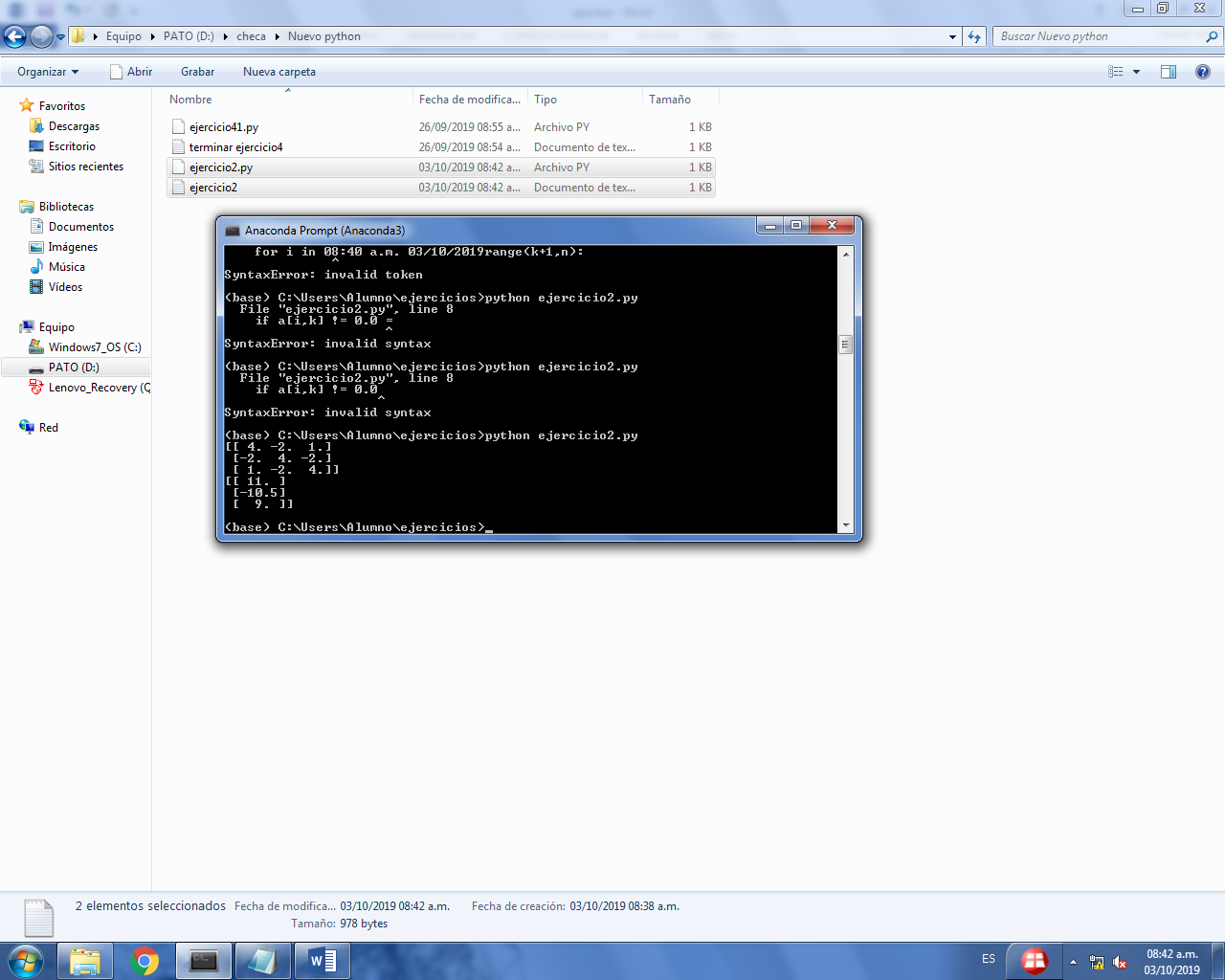
**Ejercicio**

Resolver el siguiente sistema de ecuaciones

**Algoritmo para el método**

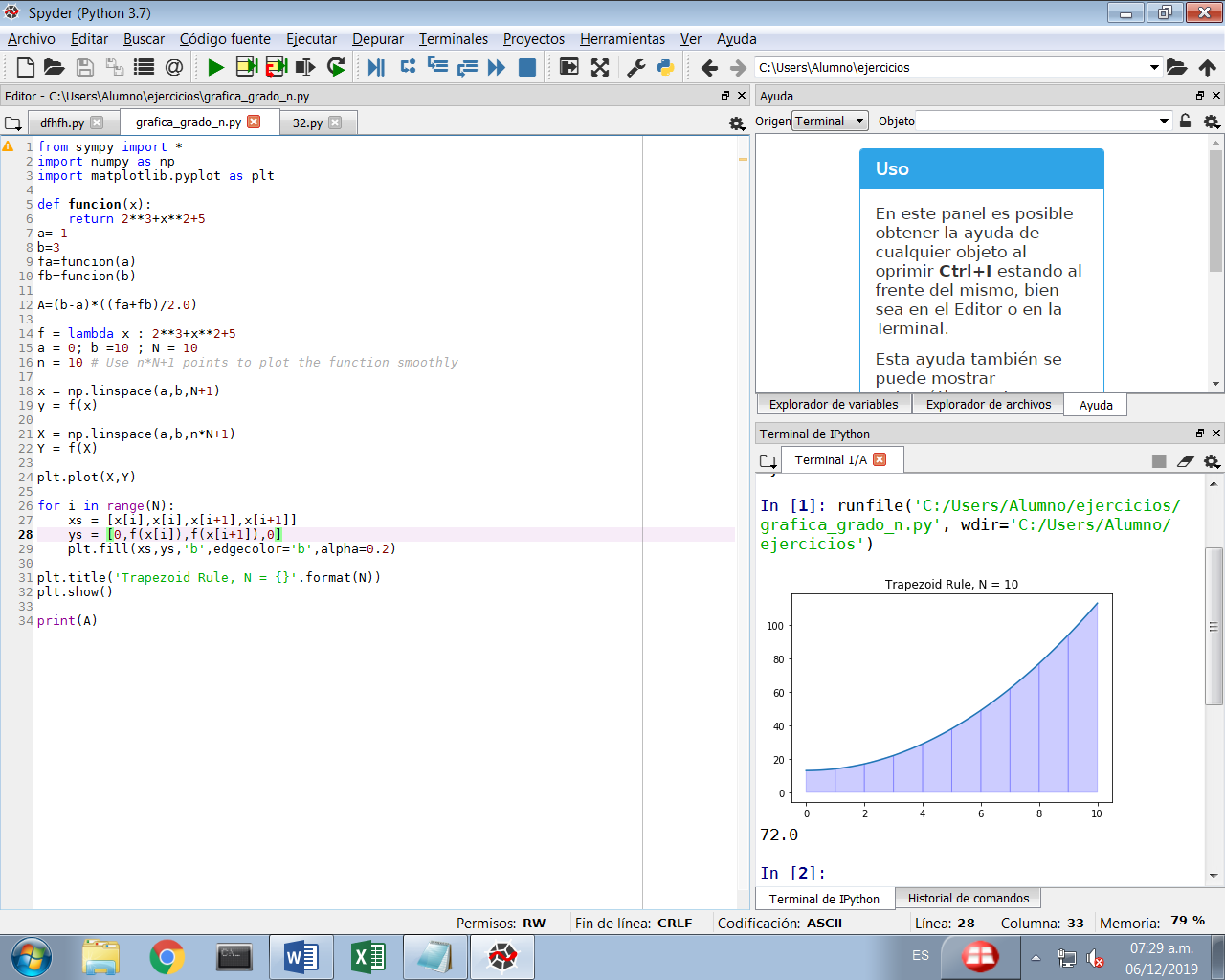
**Fase de eliminación**

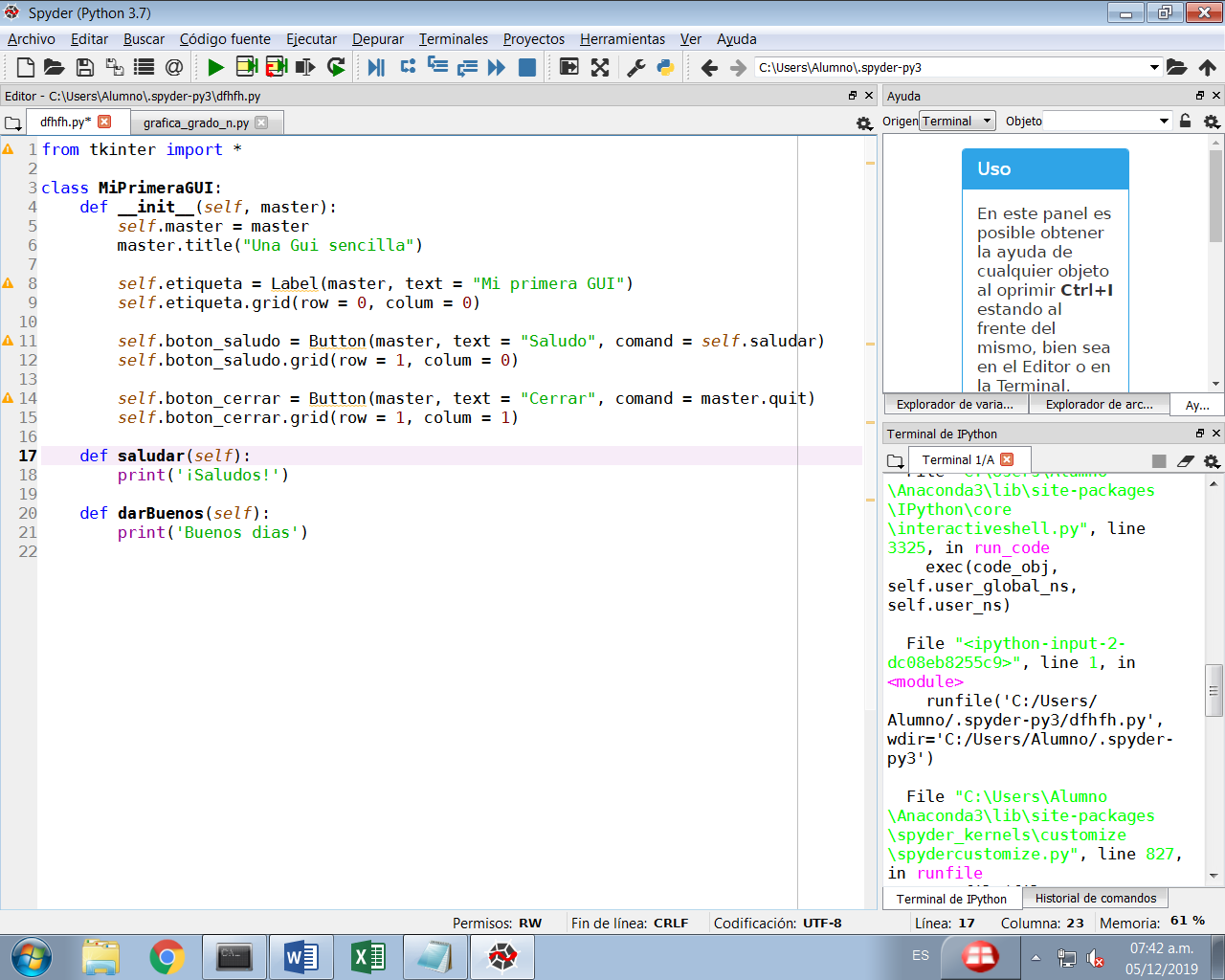
Veamos en las ecuaciones en algún instante de la fase de eliminación. Asumamos que las primeras “k - filas” de “A” han sido transformadas a una forma de matriz triangular.

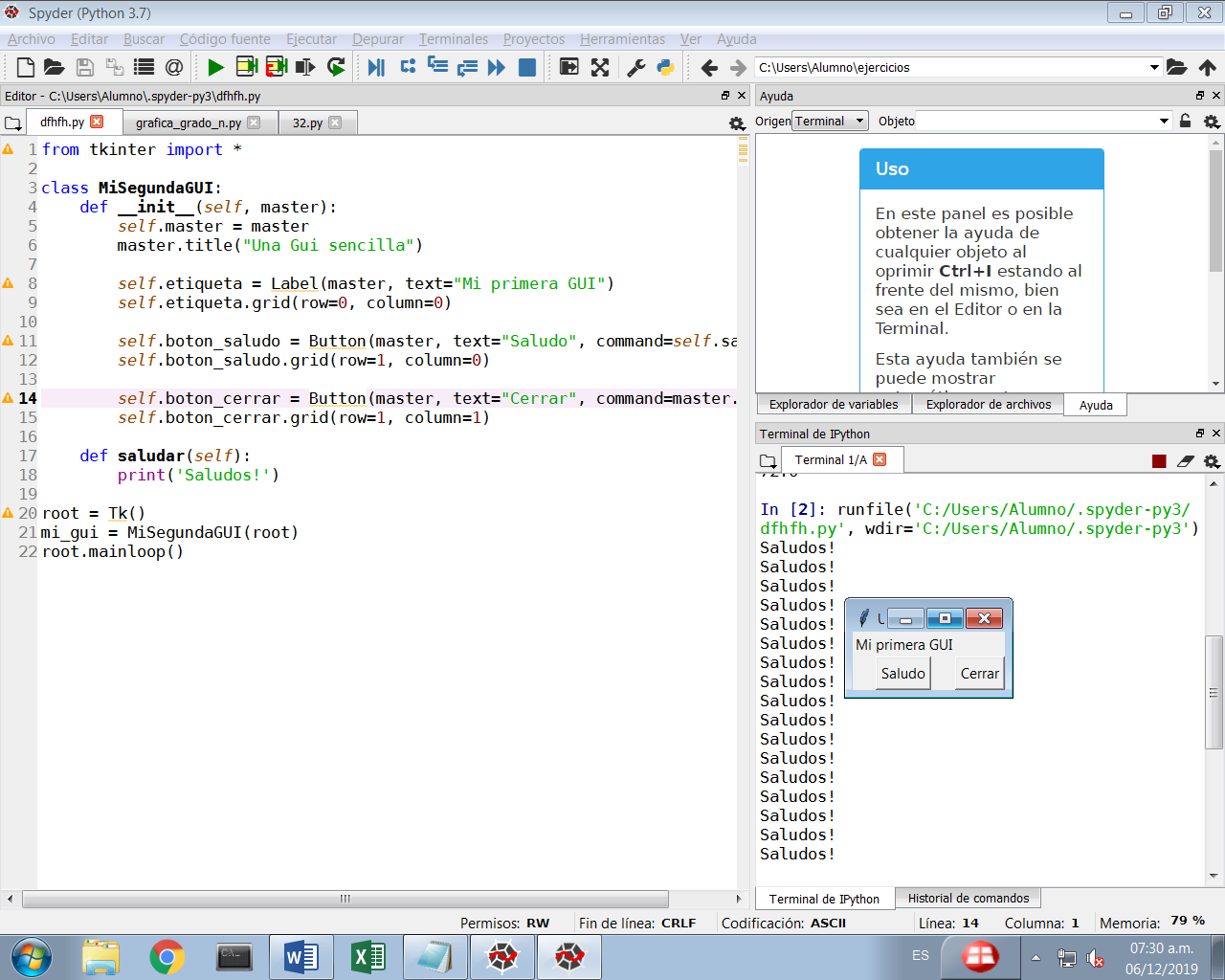


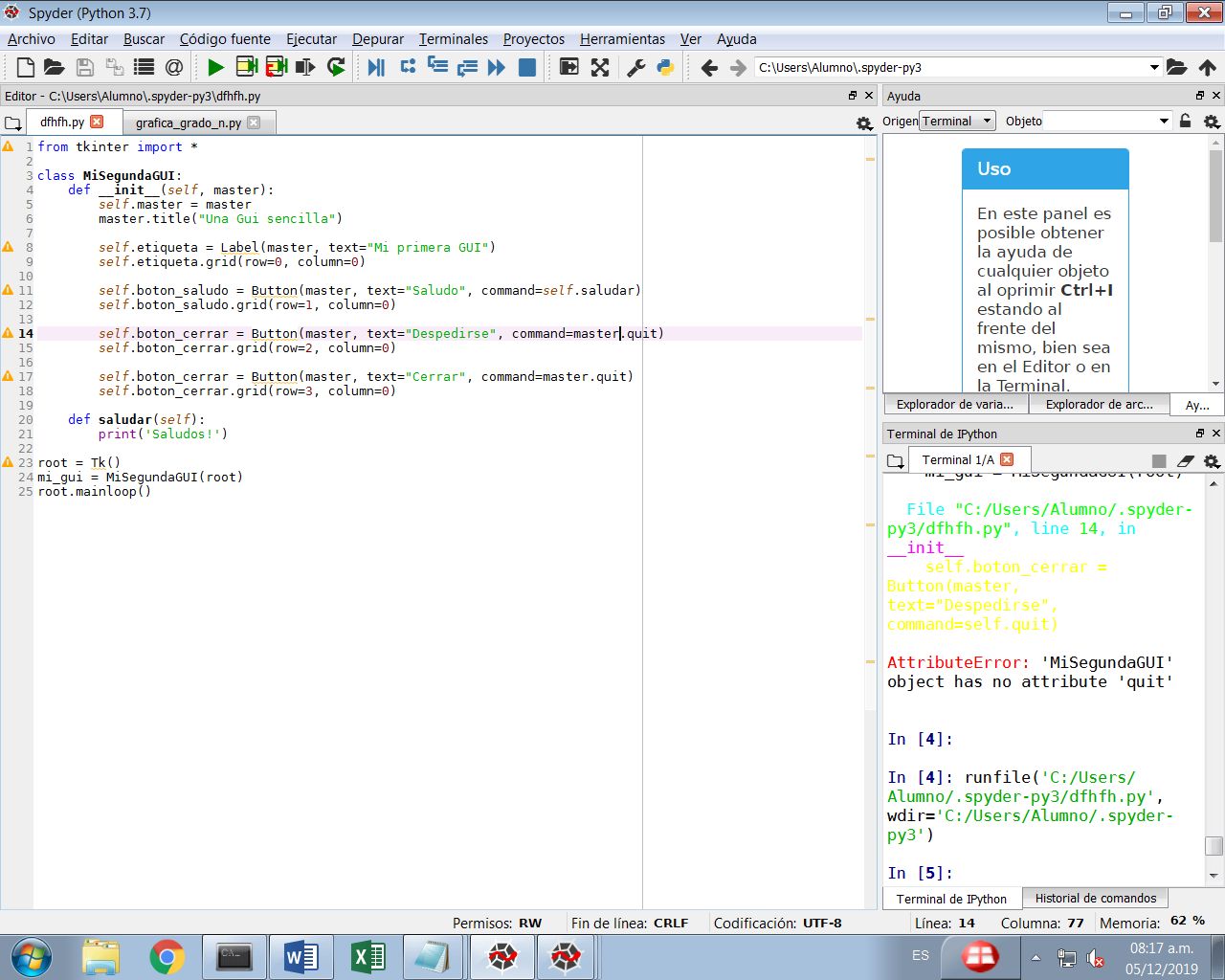
**1. Usar el método de eliminación de Gauss para encontrar la solución del sistema de ecuaciones**

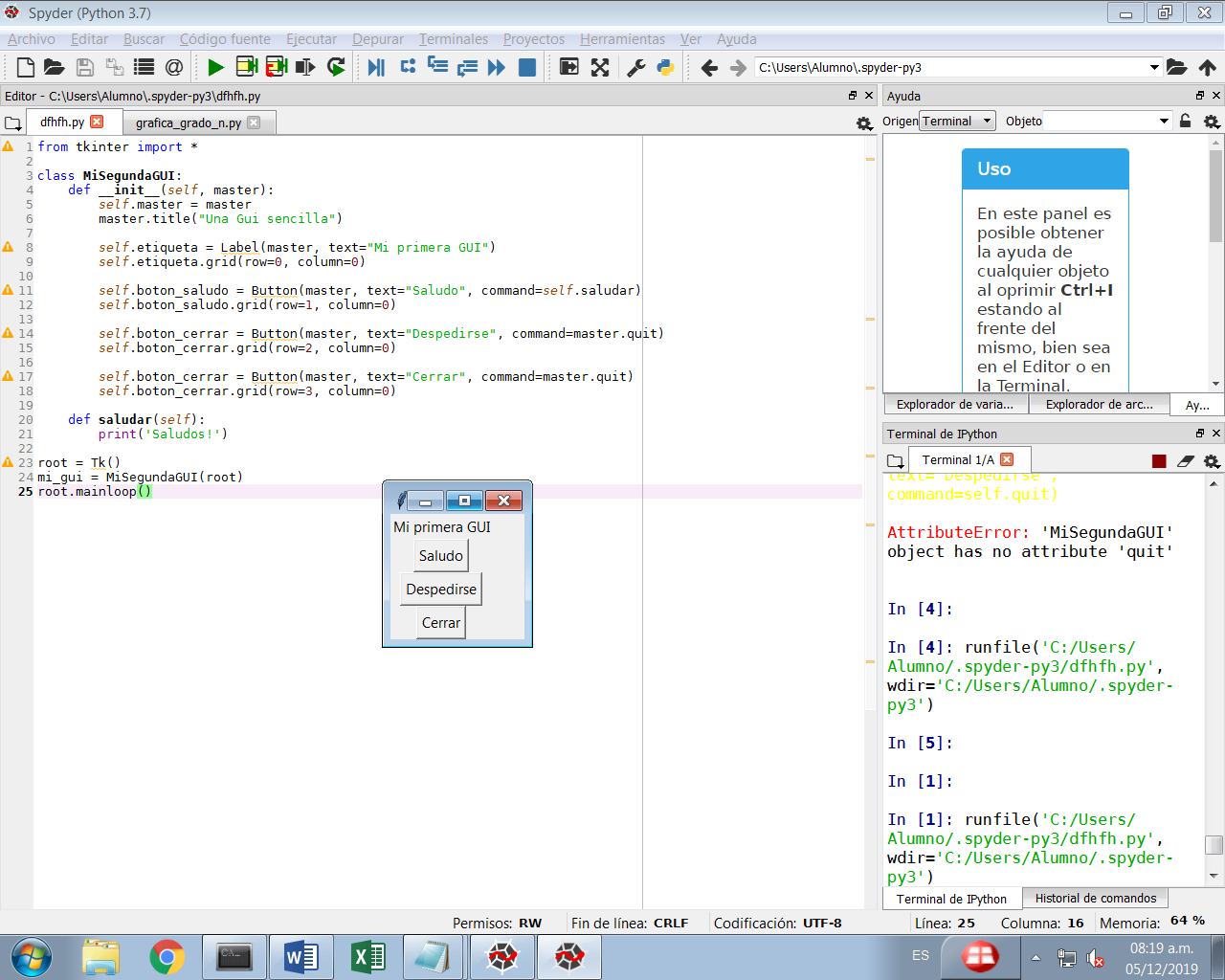
**Ejercicio 2. El círculo**



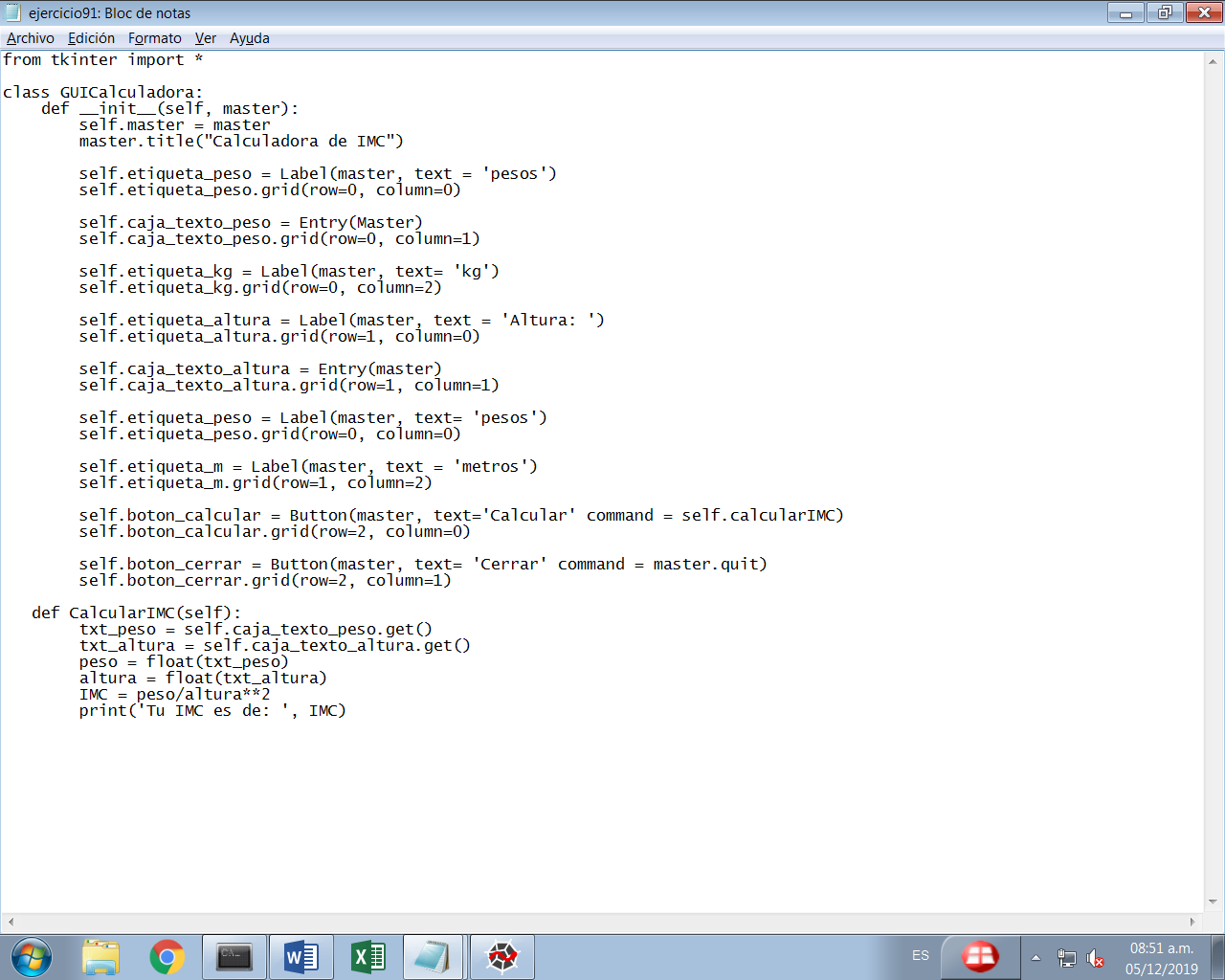


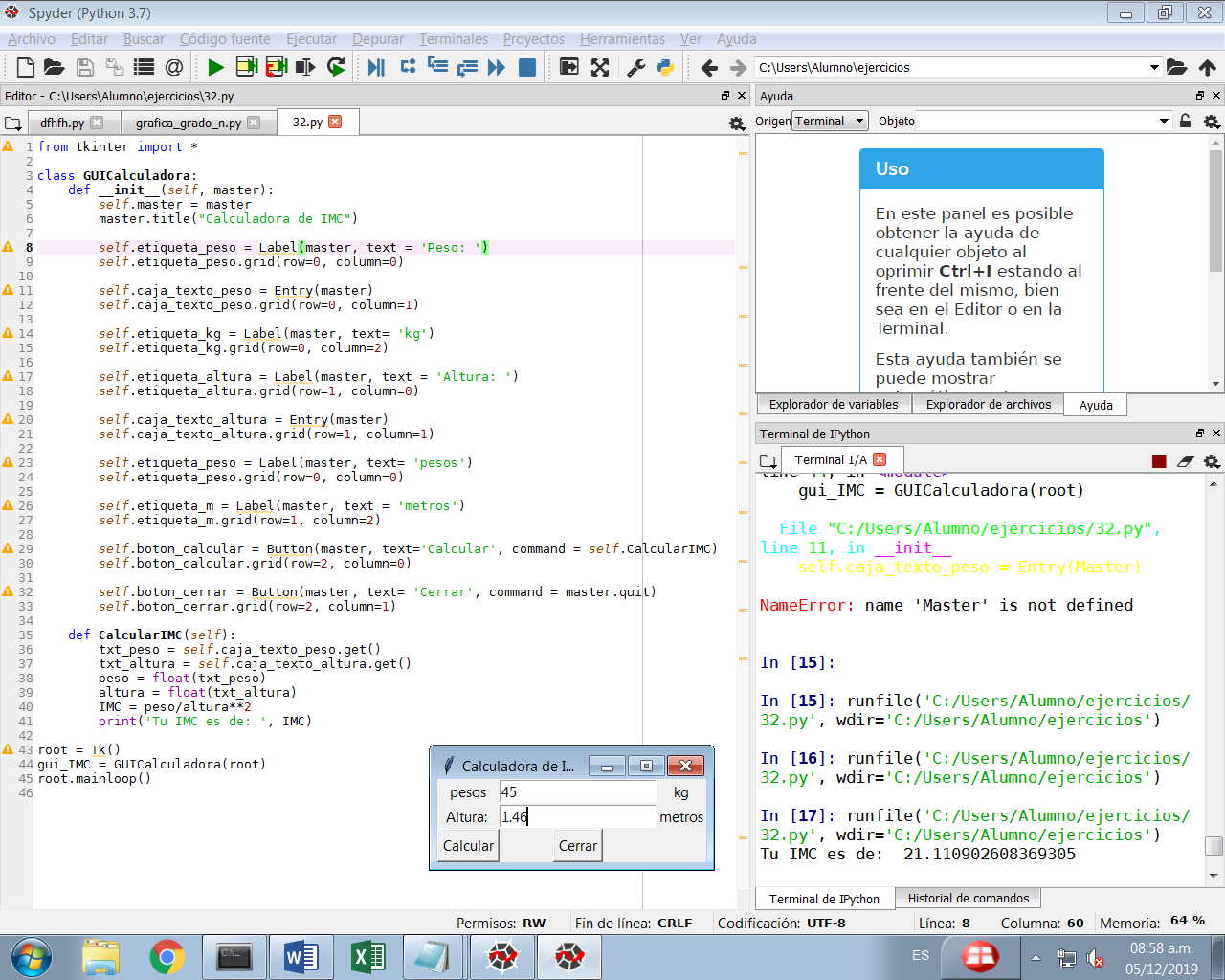


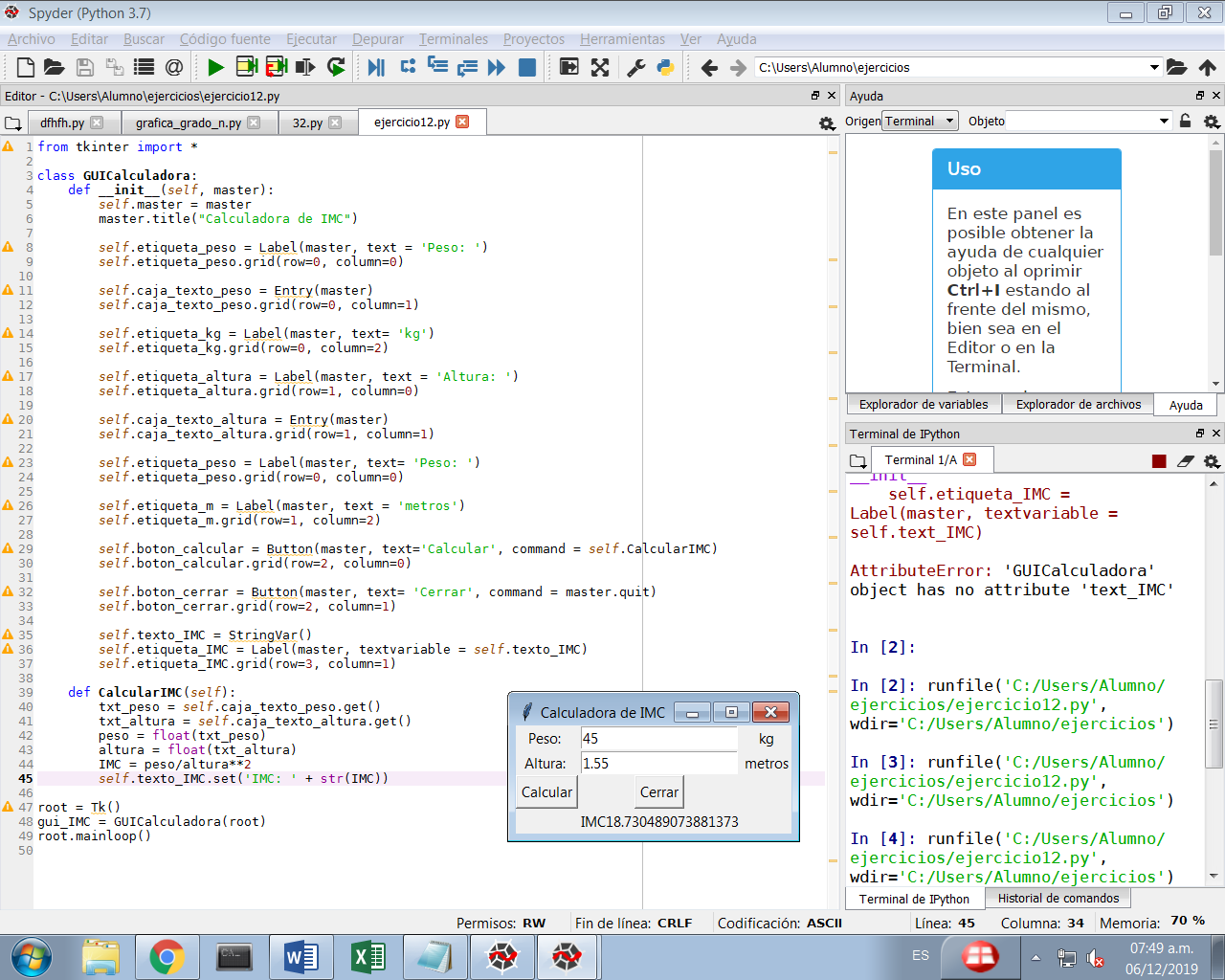




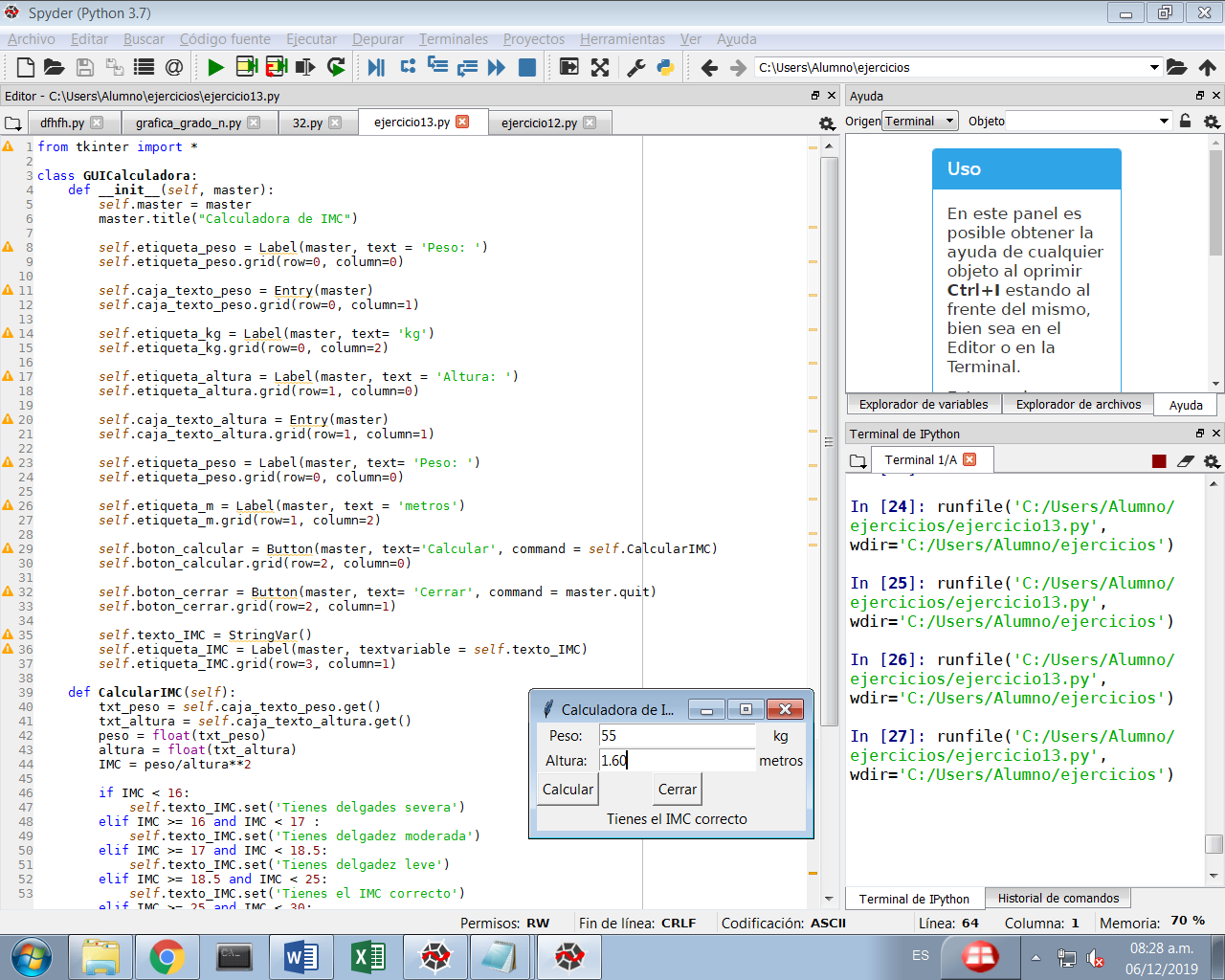
**Segunda parte tabla de IMC**







**Calcular IMC**



**Imagen que contiene captura de pantalla

Descripción generada automáticamente**

**Eliminación de Gauss con pivote de fila**

**escalado**

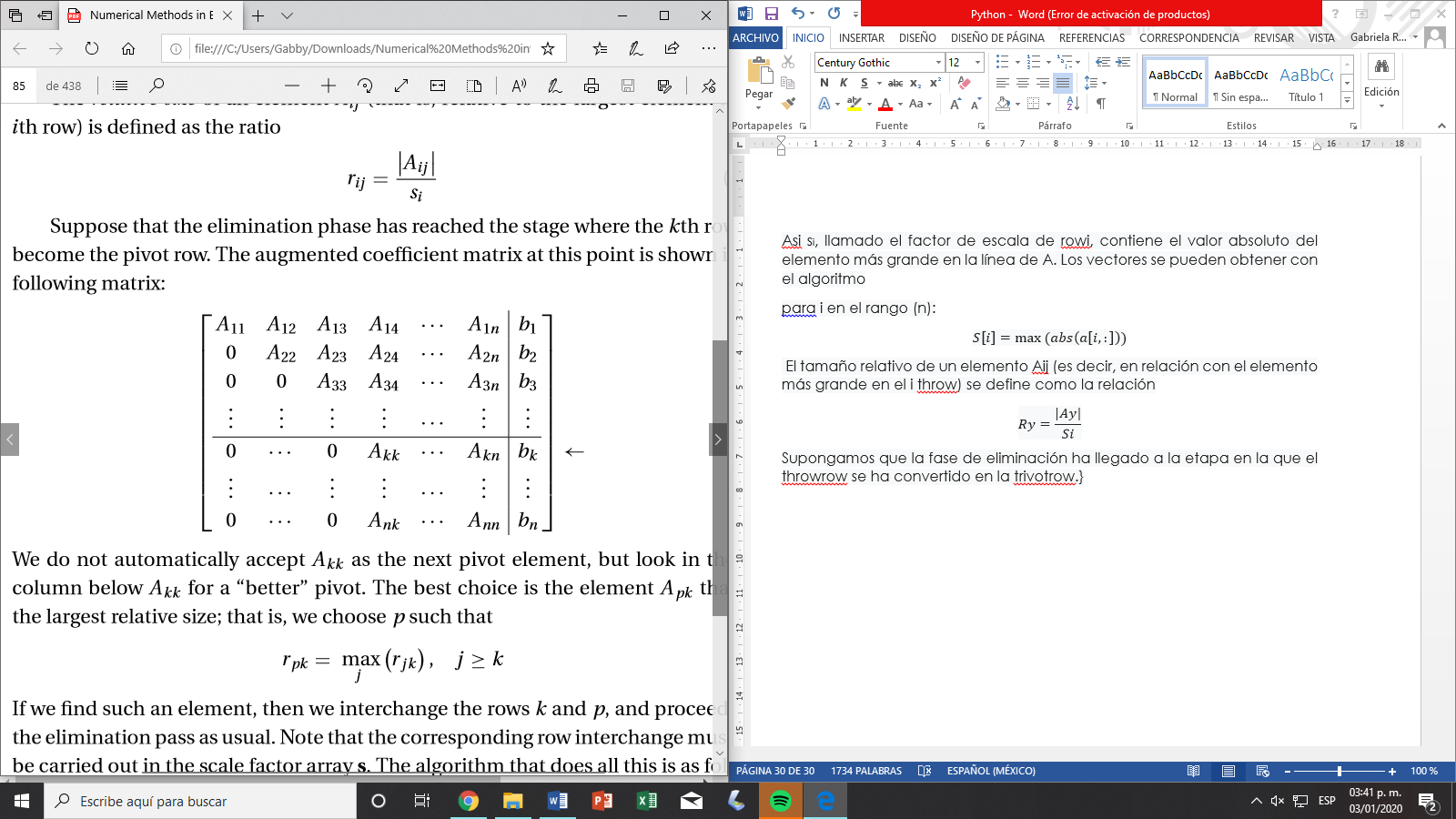
Considere la solución de Ax = b mediante la eliminación de Gauss con pivote de fila. Recuerde que los objetivos pivotantes mejoran el dominio diagonal de la matriz de coeficientes (es decir, hacer que el elemento pivote sea lo más grande posible en comparación con otros elementos en la fila pivote). La comparación se hace más fácil si esta matriz se presenta con los elementos

Así si, llamado el factor de escala de rowi, contiene el valor absoluto del elemento más grande en la línea de A. Los vectores se pueden obtener con el algoritmo

Para i en el rango (n):

El tamaño relativo de un elemento Aij (es decir, en relación con el elemento más grande en el i throw) se define como la relación

Supongamos que la fase de eliminación ha llegado a la etapa en la que el throwrow se ha convertido en la trivotrow.}



No aceptamos automáticamente Akk como el próximo elemento pivote, pero busque en la columna kth debajo de Akk un "pivote" mejor. La mejor opción es el elemento Apk que tiene el mayor tamaño relativo; es decir, elegimos p tal que

Si encontramos un elemento así, entonces intercambiamos las filas k y p, y procedemos con la eliminación del paso de manera inusual.

Para k en el rango (0, n-1):

# Buscar fila que contiene un elemento con el tamaño relativo más grande

p = argmax (abs (a [k: n, k]) / s [k: n]) + k

# Si este elemento es muy pequeño, la matriz es singular

Import numpy as np

Import swap

Import error

def gaussPivot(a,b,tol=1.0e-12):

n = len(b)

# Set up scale factors

s = np.zeros(n)

for i in range(n):

s[i] = max(np.abs(a[i,:]))

for k in range(0,n-1):

# Row interchange, if needed

p = np.argmax(np.abs(a[k:n,k])/s[k:n]) + k

if abs(a[p,k]) < tol: error.err(’Matrix is singular’)

if p != k:

swap.swapRows(b,k,p)

swap.swapRows(s,k,p)

swap.swapRows(a,k,p)

# Elimination

for i in range(k+1,n):

if a[i,k] != 0.0:

lam = a[i,k]/a[k,k]

a[i,k+1:n] = a[i,k+1:n] - lam\*a[k,k+1:n] b[i] =

b[i] - lam\*b[k]

if abs(a[n-1,n-1]) < tol: error.err(’Matrix is singular’)

# Back substitution

b[n-1] = b[n-1]/a[n-1,n-1]

for k in range(n-2,-1,-1): b[k] =

(b[k] - np.dot(a[k,k+1:n],b[k+1:n]))/a[k,k]

return b

El algoritmo de eliminación de Gauss se puede cambiar a la descomposición de Doolittle con pequeños cambios. El más importante de estos cambios es mantener un registro de los intercambios de filas durante la fase de descomposición. En LUdecomp, este registro se mantiene en la matriz seq. Inicialmente, seq contiene [0, 1, 2, ...]. Siempre que se intercambian dos filas, el intercambio correspondiente también se realiza en seq. Por lo tanto, seq muestra el orden en que se han reorganizado las filas originales. Esta información se pasa a la fase de solución (LUsolve), que reorganiza los elementos del vector constante en el mismo orden antes de proceder a las sustituciones hacia adelante y hacia atrás.