Recuperación 1

Programación de Redes

Profesor: Gabriel Barron Rodríguez

Alumno: Alan Francisco Emmanuel Aguilar Fuentes

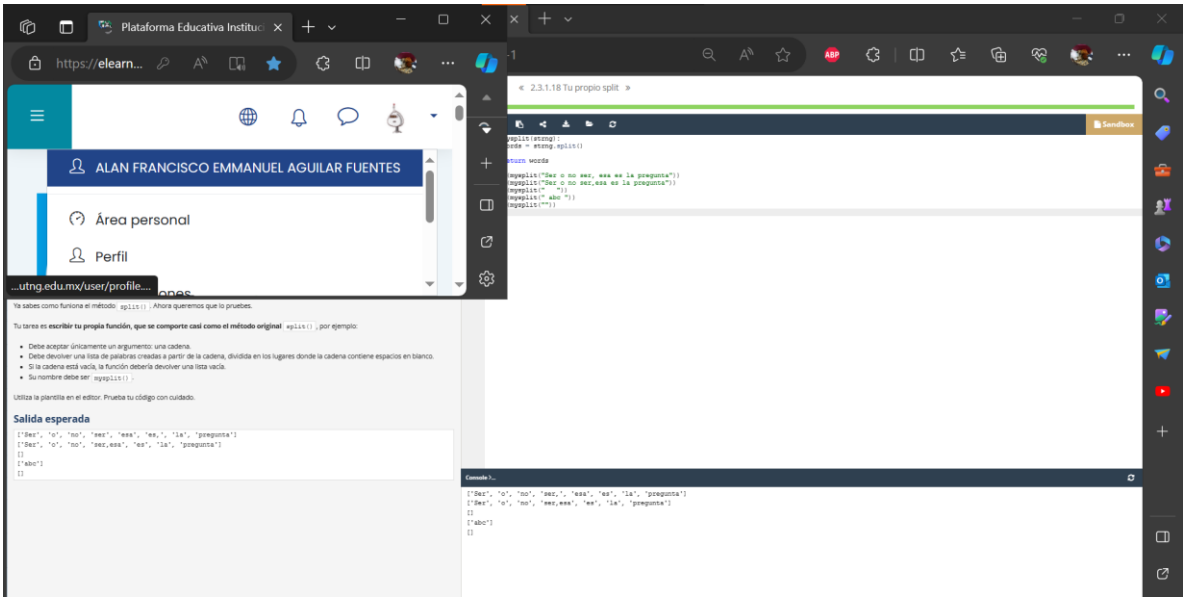
Laboratorios Python 2

Contenido

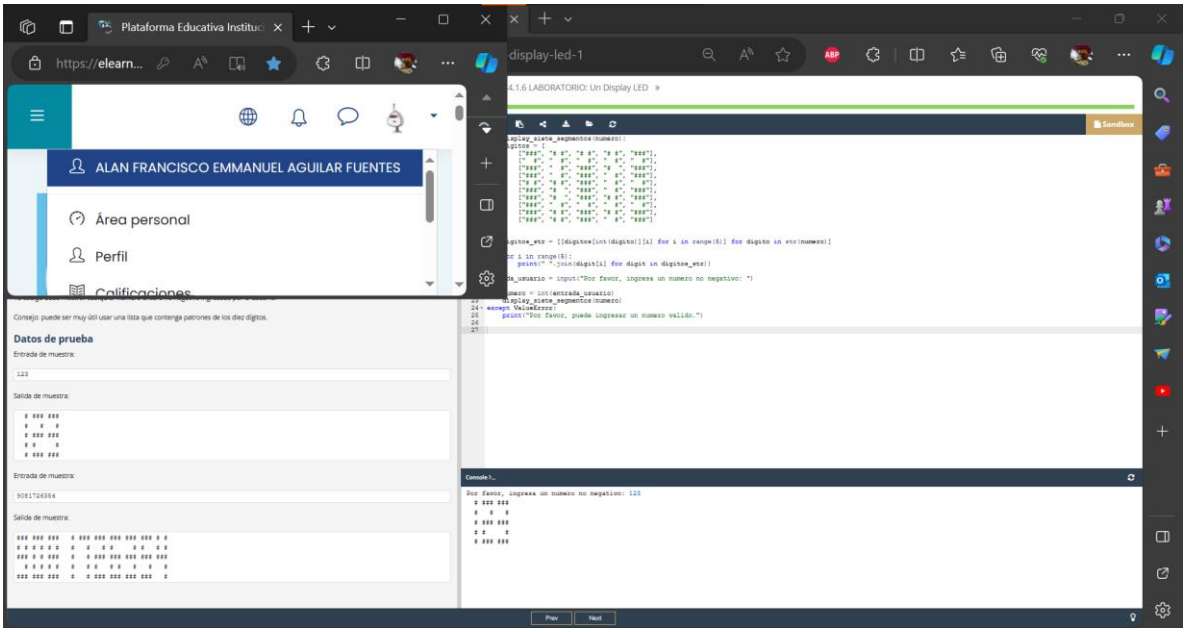
Laboratorios 1
2.3.1.18 1
2.4.1.6 1
2.8.1.4 2
3.2.1.14 2
3.2.1.15 3
3.4.1.12 3
3.4.1.13 4
3.4.1.14 4
3.4.1.15 5
4.3.1.15 5
4.3.1.16 6
4.3.1.17 6
4.4.1.8 7
4.5.1.22 7
4.6.1.13 8

Laboratorios

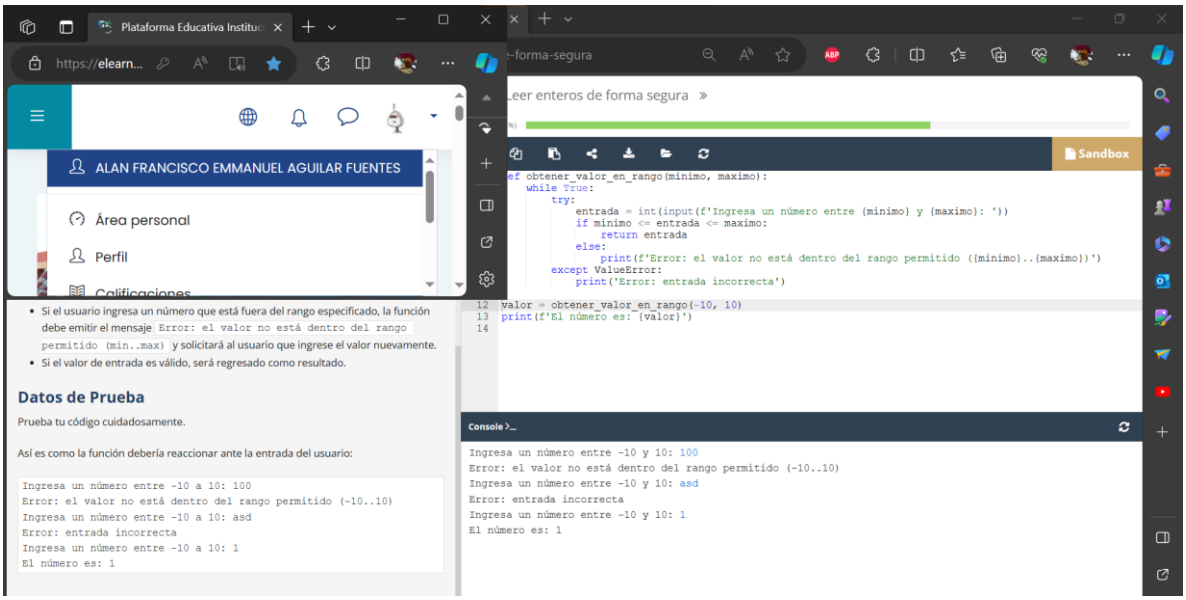
2.3.1.18



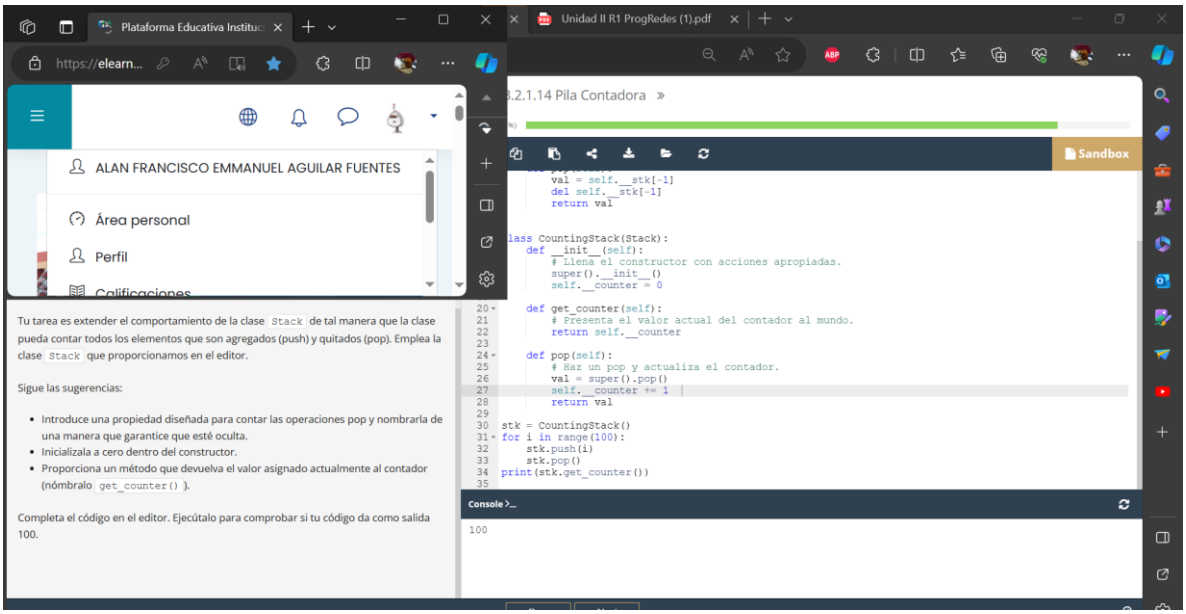
2.4.1.6



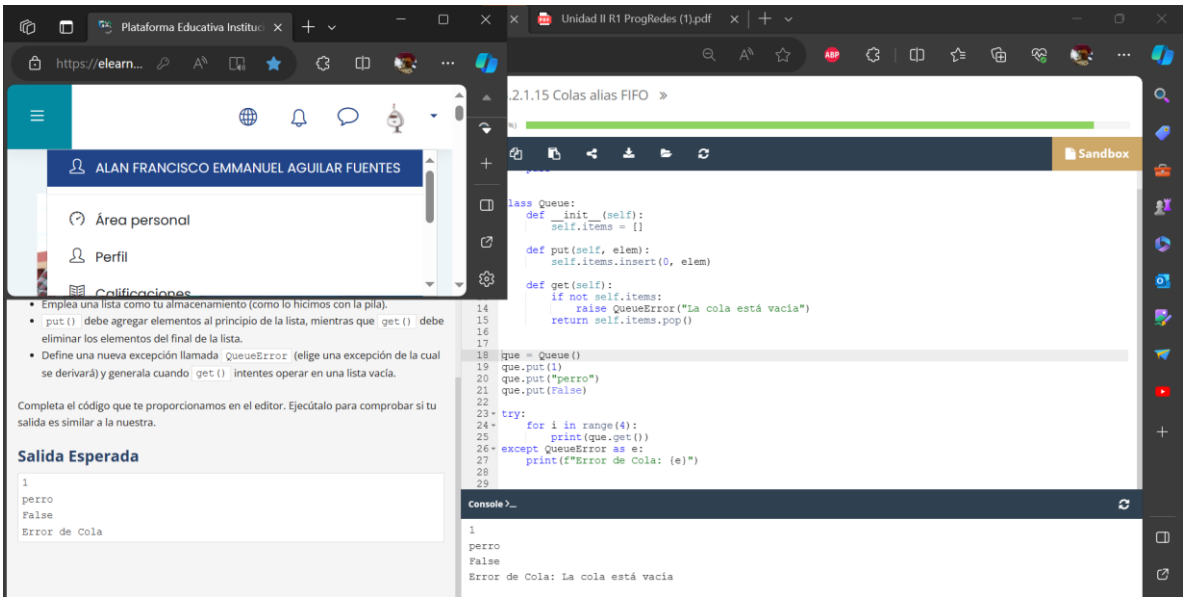
2.8.1.4


Plataforma Educativa Institu
https://elearn...
ALAN FRANCISCO EMMANUEL AGUILAR FUENTES
Área personal
Perfil
Calificaciones
Leer enteros de forma segura
Sandbox
def obtener_valor_en_rango(minimo, maximo):
while True:
try:
entrada = int(input(f'Ingresa un número entre {minimo} y {maximo}: '))
if minimo <= entrada <= maximo:
return entrada
else:
print(f'Error: el valor no está dentro del rango permitido ({minimo}..{maximo})')
except ValueError:
print('Error: entrada incorrecta')
valor = obtener_valor_en_rango(-10, 10)
print(f'El número es: {valor}')
Si el usuario ingresa un número que está fuera del rango especificado, la función debe emitir el mensaje Error: el valor no está dentro del rango permitido (min..max) y solicitará al usuario que ingrese el valor nuevamente.
Si el valor de entrada es válido, será regresado como resultado.
Datos de Prueba
Prueba tu código cuidadosamente.
Así es como la función debería reaccionar ante la entrada del usuario:
Ingresa un número entre -10 a 10: 100
Error: el valor no está dentro del rango permitido (-10..10)
Ingresa un número entre -10 a 10: asd
Error: entrada incorrecta
Ingresa un número entre -10 a 10: 1
El número es: 1
Console >_
Ingresa un número entre -10 y 10: 100
Error: el valor no está dentro del rango permitido (-10..10)
Ingresa un número entre -10 y 10: asd
Error: entrada incorrecta
Ingresa un número entre -10 y 10: 1
El número es: 1

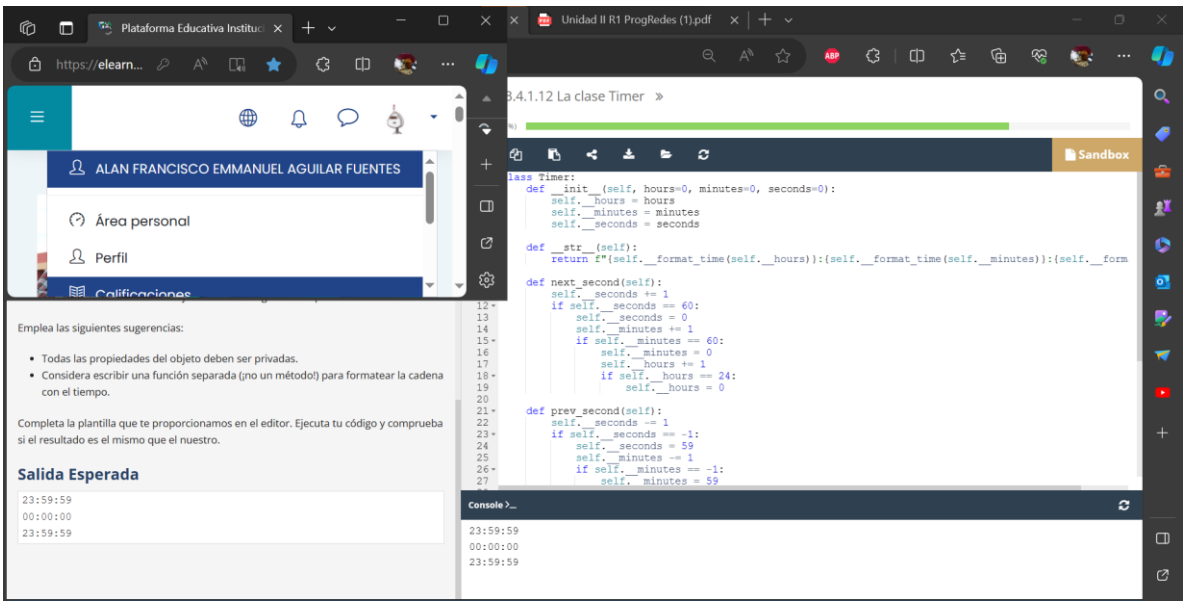
3.2.1.14


Plataforma Educativa Institu
Unidad II R1 ProgRedes (1).pdf
https://elearn...
ALAN FRANCISCO EMMANUEL AGUILAR FUENTES
Área personal
Perfil
Calificaciones
3.2.1.14 Pila Contadora
Sandbox
val = self.__stk[-1]
del self.__stk[-1]
return val
class CountingStack(Stack):
def __init__(self):
# Llena el constructor con acciones apropiadas.
super().__init__()
self.__counter = 0
def get_counter(self):
# Presenta el valor actual del contador al mundo.
return self.__counter
def pop(self):
# Haz un pop y actualiza el contador.
val = super().pop()
self.__counter += 1
return val
stk = CountingStack()
for i in range(100):
stk.push(i)
stk.pop()
print(stk.get_counter())
Tu tarea es extender el comportamiento de la clase Stack de tal manera que la clase pueda contar todos los elementos que son agregados (push) y quitados (pop). Emplea la clase Stack que proporcionamos en el editor.
Sigue las sugerencias:
Introduce una propiedad diseñada para contar las operaciones pop y nombrarla de una manera que garantice que esté oculta.
Inicializala a cero dentro del constructor.
Proporciona un método que devuelva el valor asignado actualmente al contador (nómbralo get_counter()).
Completa el código en el editor. Ejecútalo para comprobar si tu código da como salida 100.
Console >_
100

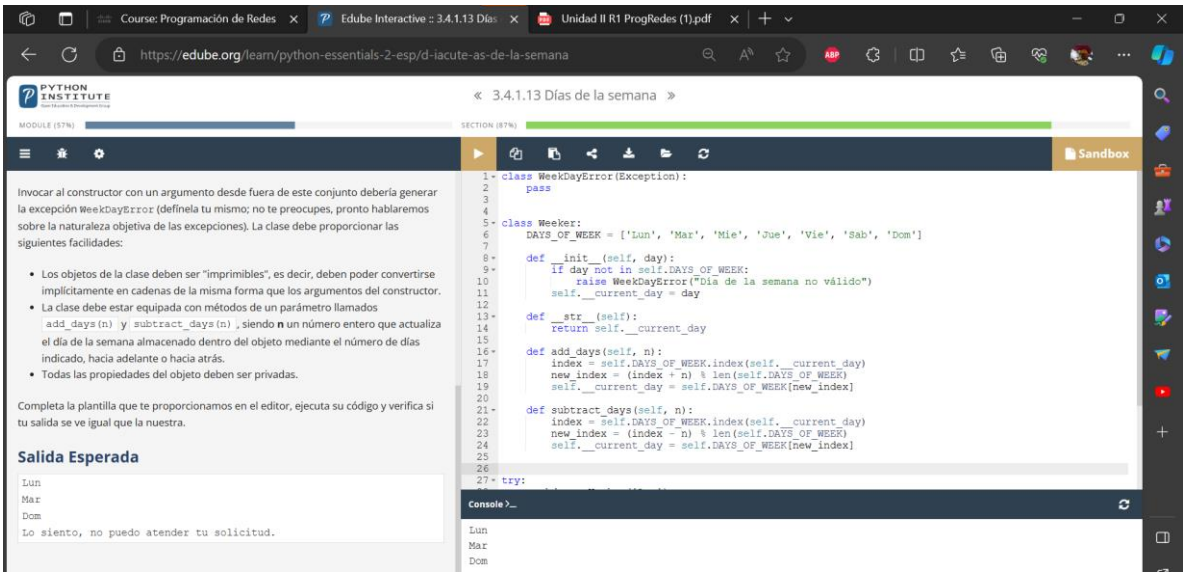
3.2.1.15


Plataforma Educativa Institu
https://elearn...
ALAN FRANCISCO EMMANUEL AGUILAR FUENTES
Área personal
Perfil
Calificaciones
Unidad II R1 ProgRedes (1).pdf
.2.1.15 Colas alias FIFO
Sandbox
Emplea una lista como tu almacenamiento (como lo hicimos con la pila).
put() debe agregar elementos al principio de la lista, mientras que get() debe eliminar los elementos del final de la lista.
Define una nueva excepción llamada QueueError (elige una excepción de la cual se derivará) y generala cuando get() intentes operar en una lista vacía.
Completa el código que te proporcionamos en el editor. Ejecútalo para comprobar si tu salida es similar a la nuestra.
Salida Esperada
1
perro
False
Error de Cola
class Queue:
def __init__(self):
self.items = []
def put(self, elem):
self.items.insert(0, elem)
def get(self):
if not self.items:
raise QueueError("La cola está vacía")
return self.items.pop()
que = Queue()
que.put(1)
que.put("perro")
que.put(False)
try:
for i in range(4):
print(que.get())
except QueueError as e:
print(f"Error de Cola: {e}")
Console >_
1
perro
False
Error de Cola: La cola está vacía

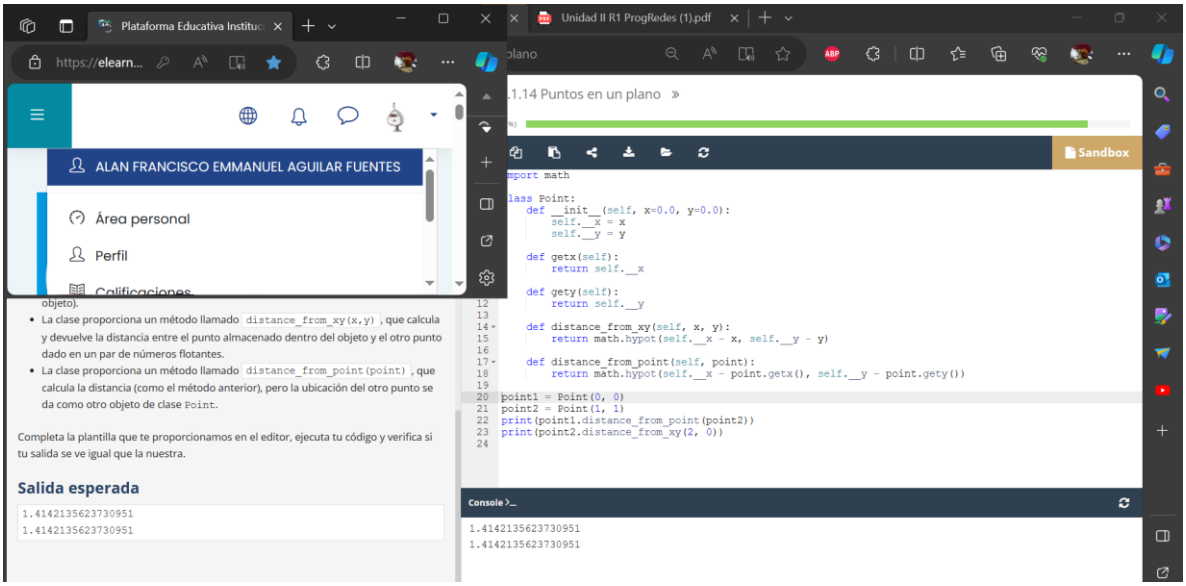
3.4.1.12


Plataforma Educativa Institu
https://elearn...
ALAN FRANCISCO EMMANUEL AGUILAR FUENTES
Área personal
Perfil
Calificaciones
Unidad II R1 ProgRedes (1).pdf
3.4.1.12 La clase Timer
Sandbox
Emplea las siguientes sugerencias:
Todas las propiedades del objeto deben ser privadas.
Considera escribir una función separada (¡no un método!) para formatear la cadena con el tiempo.
Completa la plantilla que te proporcionamos en el editor. Ejecuta tu código y comprueba si el resultado es el mismo que el nuestro.
Salida Esperada
23:59:59
00:00:00
23:59:59
class Timer:
def __init__(self, hours=0, minutes=0, seconds=0):
self.__hours = hours
self.__minutes = minutes
self.__seconds = seconds
def __str__(self):
return f"{self.__format_time(self.__hours)}:{self.__format_time(self.__minutes)}:{self.__form
def next_second(self):
self.__seconds += 1
if self.__seconds == 60:
self.__seconds = 0
self.__minutes += 1
if self.__minutes == 60:
self.__minutes = 0
self.__hours += 1
if self.__hours == 24:
self.__hours = 0
def prev_second(self):
self.__seconds -= 1
if self.__seconds == -1:
self.__seconds = 59
self.__minutes -= 1
if self.__minutes == -1:
self.__minutes = 59
Console >_
23:59:59
00:00:00
23:59:59

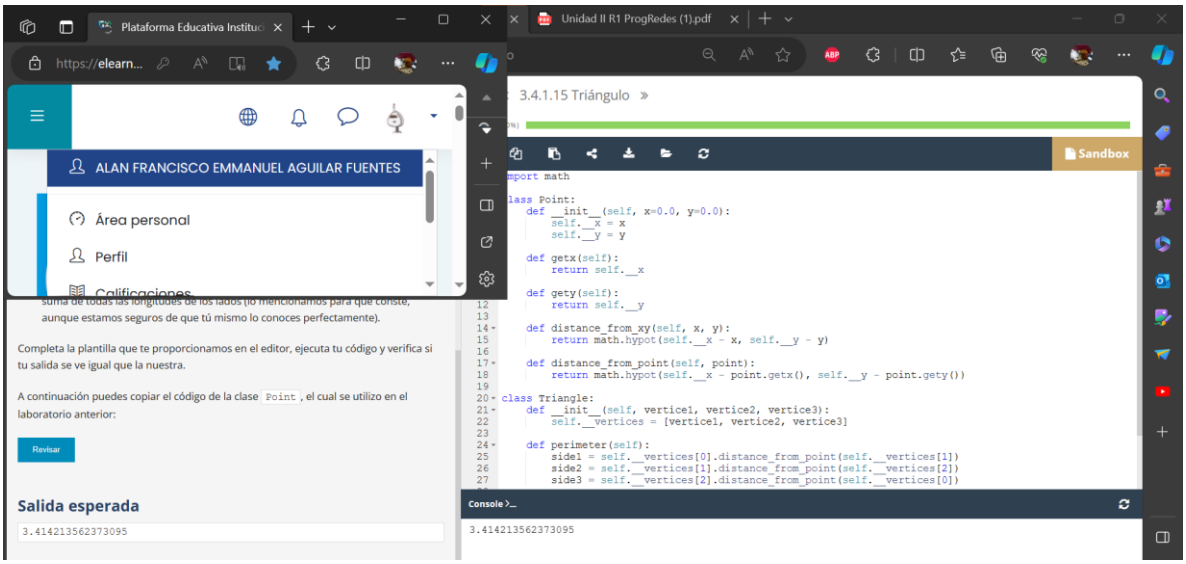
3.4.1.13


Course: Programación de Redes
Edube Interactive :: 3.4.1.13 Días
Unidad II R1 ProgRedes (1).pdf
https://edube.org/learn/python-essentials-2-esp/d-iacute-as-de-la-semana
PYTHON INSTITUTE
« 3.4.1.13 Días de la semana »
Sandbox
Invocar al constructor con un argumento desde fuera de este conjunto debería generar la excepción WeekDayError (defínela tu mismo; no te preocupes, pronto hablaremos sobre la naturaleza objetiva de las excepciones). La clase debe proporcionar las siguientes facilidades:
Los objetos de la clase deben ser "imprimibles", es decir, deben poder convertirse implícitamente en cadenas de la misma forma que los argumentos del constructor.
La clase debe estar equipada con métodos de un parámetro llamados add_days(n) y subtract_days(n), siendo n un número entero que actualiza el día de la semana almacenado dentro del objeto mediante el número de días indicado, hacia adelante o hacia atrás.
Todas las propiedades del objeto deben ser privadas.
Completa la plantilla que te proporcionamos en el editor, ejecuta su código y verifica si tu salida se ve igual que la nuestra.
Salida Esperada
Lun
Mar
Dom
Lo siento, no puedo atender tu solicitud.
class WeekDayError(Exception):
pass
class Weeker:
DAYS_OF_WEEK = ['Lun', 'Mar', 'Mie', 'Jue', 'Vie', 'Sab', 'Dom']
def __init__(self, day):
if day not in self.DAYS_OF_WEEK:
raise WeekDayError("Día de la semana no válido")
self.__current_day = day
def __str__(self):
return self.__current_day
def add_days(self, n):
index = self.DAYS_OF_WEEK.index(self.__current_day)
new_index = (index + n) % len(self.DAYS_OF_WEEK)
self.__current_day = self.DAYS_OF_WEEK[new_index]
def subtract_days(self, n):
index = self.DAYS_OF_WEEK.index(self.__current_day)
new_index = (index - n) % len(self.DAYS_OF_WEEK)
self.__current_day = self.DAYS_OF_WEEK[new_index]
try:
Console >_
Lun
Mar
Dom

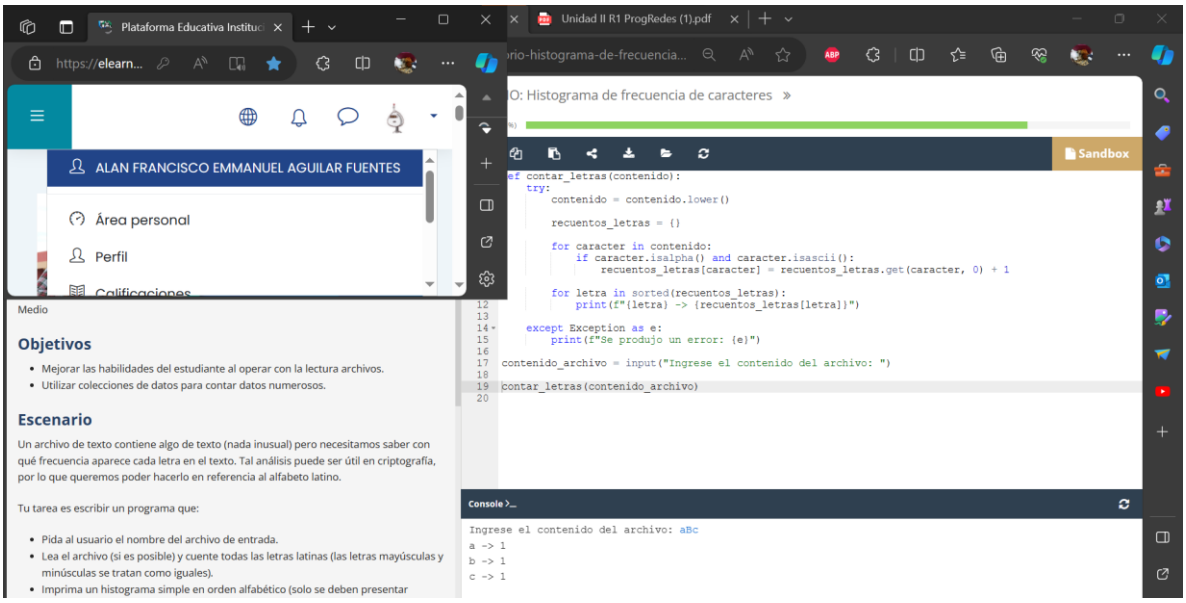
3.4.1.14


Plataforma Educativa Institu
Unidad II R1 ProgRedes (1).pdf
https://elearn...
ALAN FRANCISCO EMMANUEL AGUILAR FUENTES
Área personal
Perfil
Calificaciones
objeto).
La clase proporciona un método llamado distance_from_xy(x,y), que calcula y devuelve la distancia entre el punto almacenado dentro del objeto y el otro punto dado en un par de números flotantes.
La clase proporciona un método llamado distance_from_point(point), que calcula la distancia (como el método anterior), pero la ubicación del otro punto se da como otro objeto de clase Point.
Completa la plantilla que te proporcionamos en el editor, ejecuta su código y verifica si tu salida se ve igual que la nuestra.
Salida esperada
1.4142135623730951
1.4142135623730951
.1.14 Puntos en un plano »
Sandbox
import math
class Point:
def __init__(self, x=0.0, y=0.0):
self.__x = x
self.__y = y
def getx(self):
return self.__x
def gety(self):
return self.__y
def distance_from_xy(self, x, y):
return math.hypot(self.__x - x, self.__y - y)
def distance_from_point(self, point):
return math.hypot(self.__x - point.getx(), self.__y - point.gety())
point1 = Point(0, 0)
point2 = Point(1, 1)
print(point1.distance_from_point(point2))
print(point2.distance_from_xy(2, 0))
Console >_
1.4142135623730951
1.4142135623730951

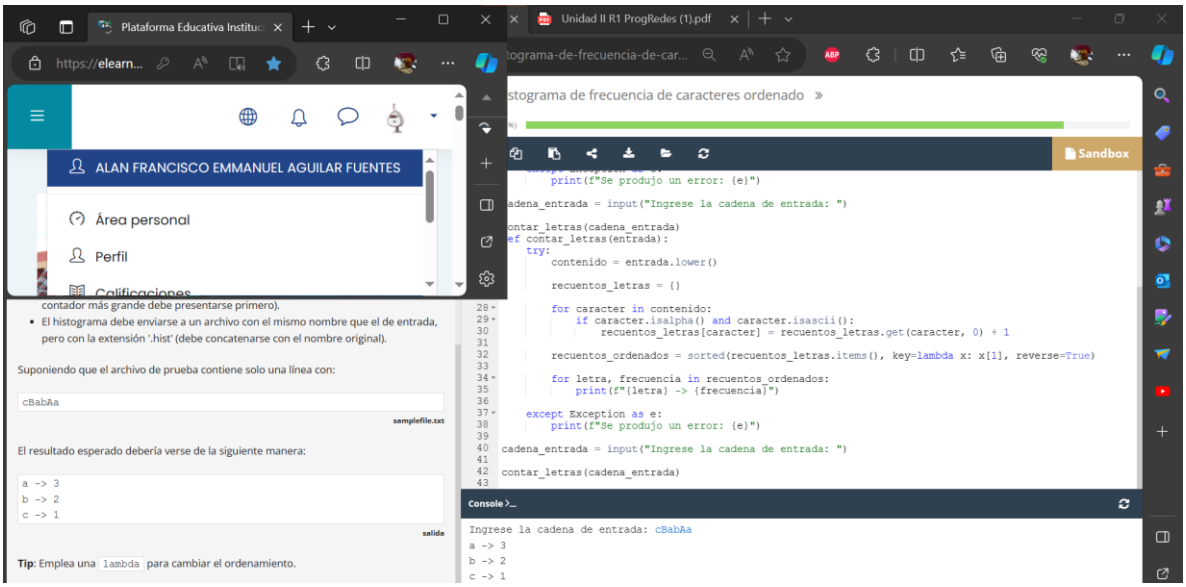
3.4.1.15


ALAN FRANCISCO EMMANUEL AGUILAR FUENTES
Área personal
Perfil
Calificaciones
Completa la plantilla que te proporcionamos en el editor, ejecuta tu código y verifica si tu salida se ve igual que la nuestra.
A continuación puedes copiar el código de la clase Point, el cual se utilizo en el laboratorio anterior:
Revisar
Salida esperada
3.414213562373095
3.4.1.15 Triángulo
Sandbox
import math
class Point:
def __init__(self, x=0.0, y=0.0):
self.__x = x
self.__y = y
def getx(self):
return self.__x
def gety(self):
return self.__y
def distance_from_xy(self, x, y):
return math.hypot(self.__x - x, self.__y - y)
def distance_from_point(self, point):
return math.hypot(self.__x - point.getx(), self.__y - point.gety())
class Triangle:
def __init__(self, vertice1, vertice2, vertice3):
self.__vertices = [vertice1, vertice2, vertice3]
def perimeter(self):
side1 = self.__vertices[0].distance_from_point(self.__vertices[1])
side2 = self.__vertices[1].distance_from_point(self.__vertices[2])
side3 = self.__vertices[2].distance_from_point(self.__vertices[0])
Console >_
3.414213562373095

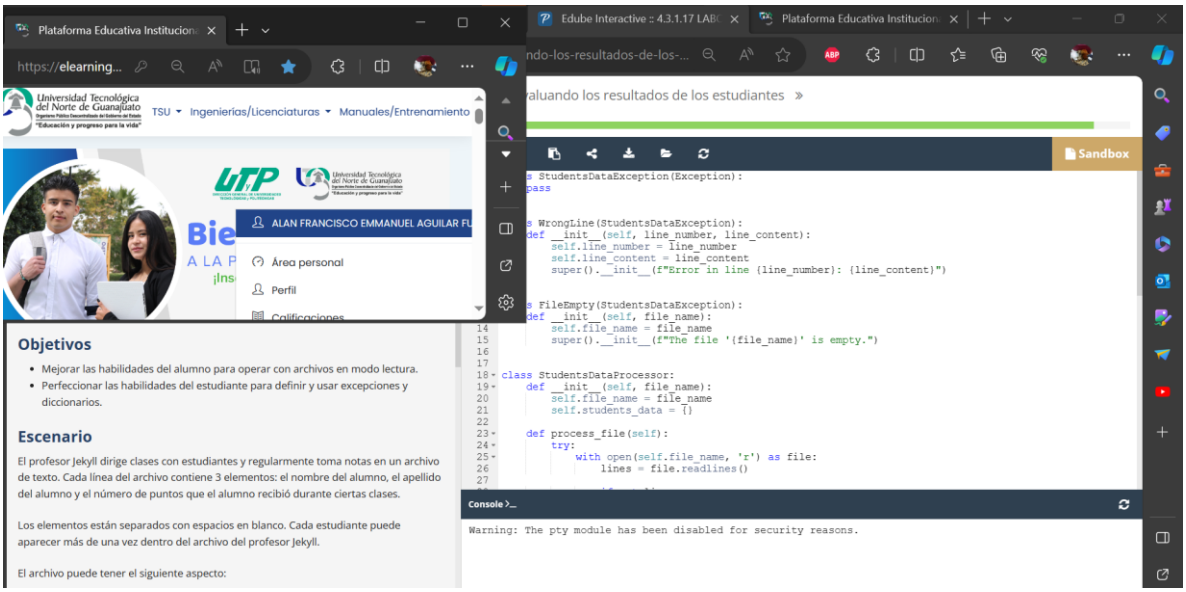
4.3.1.15


ALAN FRANCISCO EMMANUEL AGUILAR FUENTES
Área personal
Perfil
Calificaciones
Medio
Objetivos
Mejorar las habilidades del estudiante al operar con la lectura archivos.
Utilizar colecciones de datos para contar datos numerosos.
Escenario
Un archivo de texto contiene algo de texto (nada inusual) pero necesitamos saber con qué frecuencia aparece cada letra en el texto. Tal análisis puede ser útil en criptografía, por lo que queremos poder hacerlo en referencia al alfabeto latino.
Tu tarea es escribir un programa que:
Pida al usuario el nombre del archivo de entrada.
Lea el archivo (si es posible) y cuente todas las letras latinas (las letras mayúsculas y minúsculas se tratan como iguales).
Imprima un histograma simple en orden alfabético (solo se deben presentar
O: Histograma de frecuencia de caracteres
Sandbox
def contar_letras(contenido):
try:
contenido = contenido.lower()
recuentos_letras = {}
for caracter in contenido:
if caracter.isalpha() and caracter.isascii():
recuentos_letras[caracter] = recuentos_letras.get(caracter, 0) + 1
for letra in sorted(recuentos_letras):
print(f"{letra} -> {recuentos_letras[letra]}")
except Exception as e:
print(f"Se produjo un error: {e}")
contenido_archivo = input("Ingrese el contenido del archivo: ")
contar_letras(contenido_archivo)
Console >_
Ingrese el contenido del archivo: aBc
a -> 1
b -> 1
c -> 1

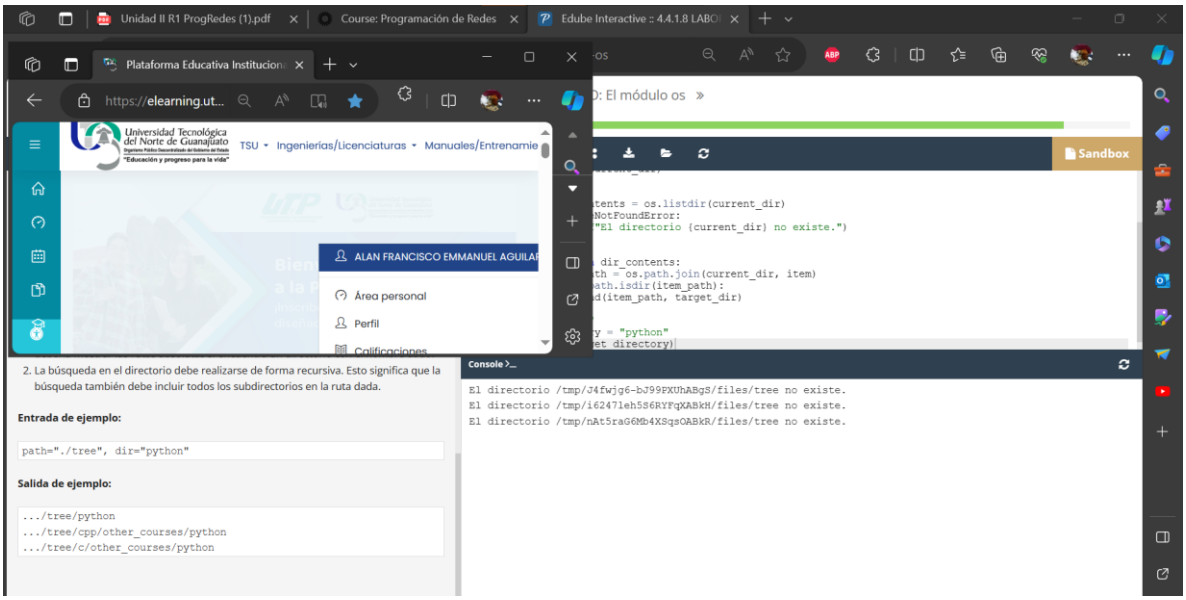
4.3.1.16


Plataforma Educativa Institu
https://elearn...
ALAN FRANCISCO EMMANUEL AGUILAR FUENTES
Área personal
Perfil
Calificaciones
Unidad II R1 ProgRedes (1).pdf
stograma de frecuencia de caracteres ordenado
Sandbox
contador más grande debe presentarse primero).
El histograma debe enviarse a un archivo con el mismo nombre que el de entrada, pero con la extensión '.hist' (debe concatenarse con el nombre original).
Suponiendo que el archivo de prueba contiene solo una línea con:
cBabAa
samplefile.txt
El resultado esperado debería verse de la siguiente manera:
a -> 3
b -> 2
c -> 1
salida
Tip: Emplea una lambda para cambiar el ordenamiento.
Console >_
Ingrese la cadena de entrada: cBabAa
a -> 3
b -> 2
c -> 1

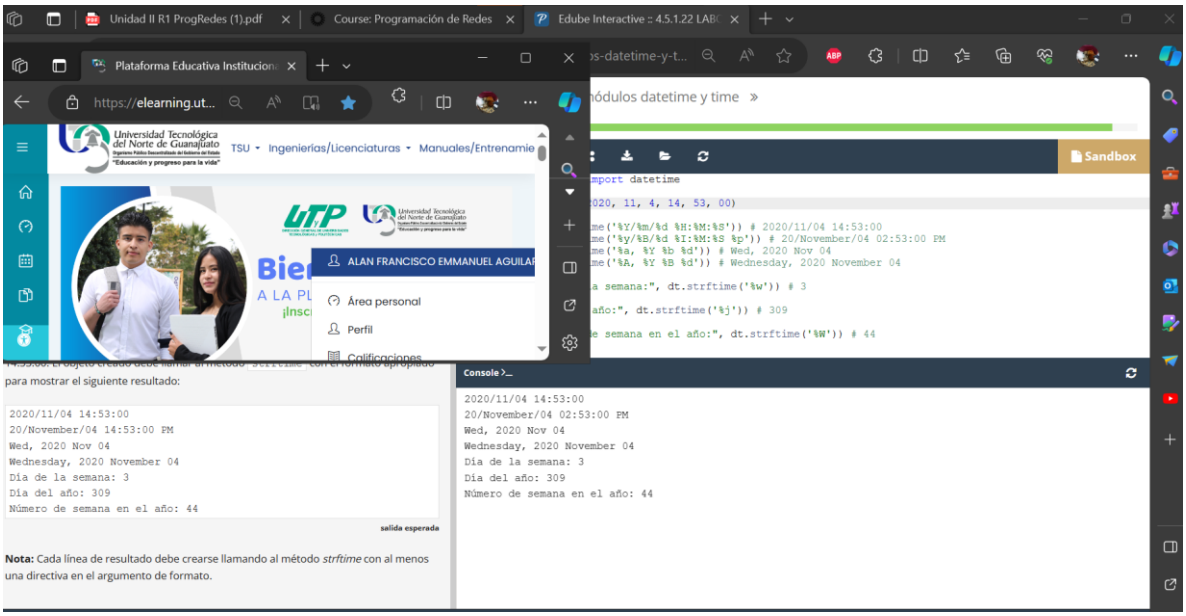
4.3.1.17


Plataforma Educativa Institucion
https://elearning...
Universidad Tecnológica del Norte de Guanajuato
TSU
Ingenierías/Licenciaturas
Manuales/Entrenamiento
ALAN FRANCISCO EMMANUEL AGUILAR FL
Área personal
Perfil
Calificaciones
Edube Interactive :: 4.3.1.17 LAB
Sandbox
Objetivos
Mejorar las habilidades del alumno para operar con archivos en modo lectura.
Perfeccionar las habilidades del estudiante para definir y usar excepciones y diccionarios.
Escenario
El profesor Jekyll dirige clases con estudiantes y regularmente toma notas en un archivo de texto. Cada línea del archivo contiene 3 elementos: el nombre del alumno, el apellido del alumno y el número de puntos que el alumno recibió durante ciertas clases.
Los elementos están separados con espacios en blanco. Cada estudiante puede aparecer más de una vez dentro del archivo del profesor Jekyll.
El archivo puede tener el siguiente aspecto:
Console >_
Warning: The pty module has been disabled for security reasons.

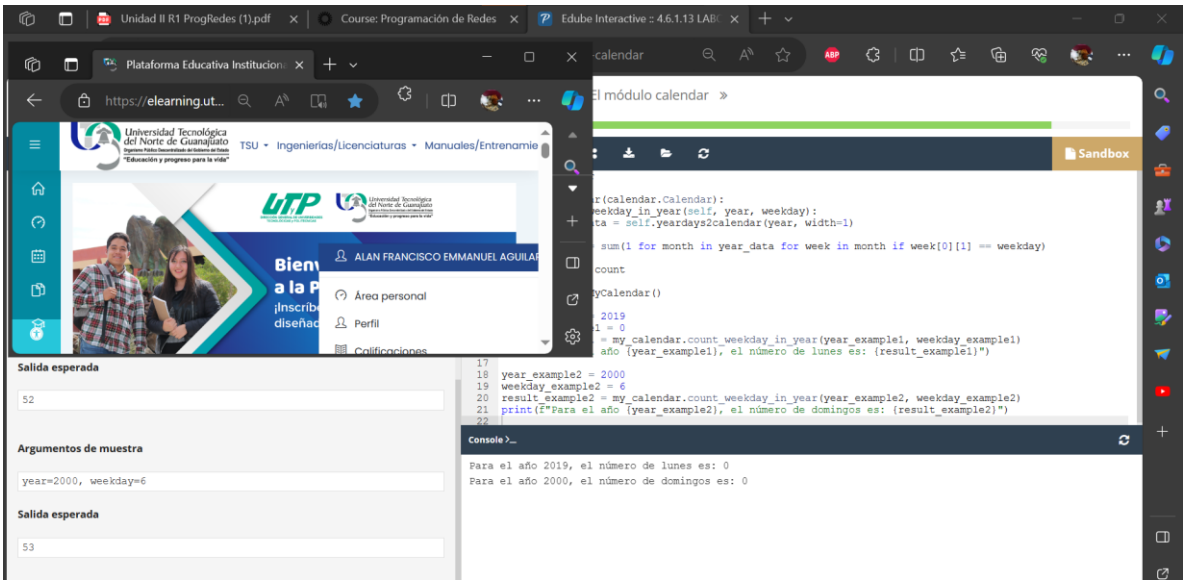
4.4.1.8


Unidad II R1 ProgRedes (1).pdf
Course: Programación de Redes
Edube Interactive :: 4.4.1.8 LABO
Plataforma Educativa Institucion
https://elearning.ut...
Universidad Tecnológica del Norte de Guanajuato
TSU
Ingenierías/Licenciaturas
Manuales/Entrenamie
ALAN FRANCISCO EMMANUEL AGUILA
Área personal
Perfil
Calificaciones
D: El módulo os
Sandbox
2. La búsqueda en el directorio debe realizarse de forma recursiva. Esto significa que la búsqueda también debe incluir todos los subdirectorios en la ruta dada.
Entrada de ejemplo:
path="./tree", dir="python"
Salida de ejemplo:
.../tree/python
.../tree/cpp/other_courses/python
.../tree/c/other_courses/python
Console >_
El directorio /tmp/J4fwjg6-bJ99PXUhABgS/files/tree no existe.
El directorio /tmp/i62471eh5S6RYFqXABkH/files/tree no existe.
El directorio /tmp/nAt5raG6Mb4XSqsOABkR/files/tree no existe.

4.5.1.22


Unidad II R1 ProgRedes (1).pdf
Course: Programación de Redes
Edube Interactive :: 4.5.1.22 LAB
Plataforma Educativa Institucion
https://elearning.ut...
Universidad Tecnológica del Norte de Guanajuato
TSU
Ingenierías/Licenciaturas
Manuales/Entrenamie
ALAN FRANCISCO EMMANUEL AGUILA
Área personal
Perfil
Calificaciones
módulos datetime y time
Sandbox
para mostrar el siguiente resultado:
2020/11/04 14:53:00
20/November/04 14:53:00 PM
Wed, 2020 Nov 04
Wednesday, 2020 November 04
Día de la semana: 3
Día del año: 309
Número de semana en el año: 44
salida esperada
Nota: Cada línea de resultado debe crearse llamando al método strftime con al menos una directiva en el argumento de formato.
Console >_
2020/11/04 14:53:00
20/November/04 02:53:00 PM
Wed, 2020 Nov 04
Wednesday, 2020 November 04
Día de la semana: 3
Día del año: 309
Número de semana en el año: 44

4.6.1.13


Unidad II R1 ProgRedes (1).pdf
Course: Programación de Redes
Edube Interactive :: 4.6.1.13 LAB
Plataforma Educativa Institucion
https://elearning.ut...
Universidad Tecnológica del Norte de Guanajuato
TSU
Ingenierías/Licenciaturas
Manuales/Entrenamie
ALAN FRANCISCO EMMANUEL AGUILAR
Área personal
Perfil
Calificaciones
Salida esperada
52
Argumentos de muestra
year=2000, weekday=6
Salida esperada
53
El módulo calendar
Sandbox
year_example2 = 2000
weekday_example2 = 6
result_example2 = my_calendar.count_weekday_in_year(year_example2, weekday_example2)
print(f"Para el año {year_example2}, el número de domingos es: {result_example2}")
Console >_
Para el año 2019, el número de lunes es: 0
Para el año 2000, el número de domingos es: 0