Tu calificación: 100 %

Tu calificación más reciente: **100 %** • Tu calificación más alta: **100 %** • Para aprobar necesitas al menos un 80 %. Guardamos tu puntaje más alto.

Próximo artículo

Estás viendo una versión traducida automáticamente de esta evaluación
Puedes volver a ver el contenido en su idioma original si lo prefieres. No perderás el progreso que hayas conseguido si cambias el idioma.
Mostrar la versión en Inglés
Desestimar

1. Está trabajando con un Archivo CSV que contiene información sobre los empleados. Cada registro tiene un campo de nombre, seguido de un campo de número de teléfono y un campo de función. El campo de número de teléfono contiene números de teléfono de EE.UU. y debe modificarse al formato internacional, con `+1-` delante del número de teléfono. El resto del número de teléfono no debe cambiar. Rellene la expresión regular, utilizando grupos, para utilizar la función `transform_record()` para hacerlo. **1 / 1 punto**

```
import re
def transform_record(record):
  new_record = re.sub(r"",(\d{3})",r",+1-\1",record)
  return new_record

print(transform_record("Sabrina Green,802-867-5309,System Administrator"))
# Sabrina Green,+1-802-867-5309,System Administrator

print(transform_record("Eli Jones,684-3481127,IT specialist"))
# Eli Jones,+1-684-3481127,IT specialist

print(transform_record("Melody Daniels,846-687-7436,Programmer"))
# Melody Daniels,+1-846-687-7436,Programmer

print(transform_record("Charlie Rivera,698-746-3357,Web Developer"))
# Charlie Rivera,+1-698-746-3357,Web Developer
```

Ejecutar
Restablecer

Correcto
¡Impresionante! Tu conocimiento de las expresiones regulares te será útil cuando trabajes aún más con archivos

2. La función `multi_vowel_words()` devuelve todas las palabras con 3 o más vocales consecutivas (a, e, i, o, u). Rellene la expresión regular para hacerlo. **1 / 1 punto**

```
import re
def multi_vowel_words(text):
  pattern = r'\b\w*[aeiouAEIOU]{3,}\w*\b'
  result = re.findall(pattern, text)
  return result

print(multi_vowel_words("Life is beautiful"))
# ['beautiful']

print(multi_vowel_words("Obviously, the queen is courageous and gracious."))
# ['Obviously', 'queen', 'courageous', 'gracious']

print(multi_vowel_words("The rambunctious children had to sit quietly and await their delicious dinner."))
# ['rambunctious', 'quietly', 'delicious']

print(multi_vowel_words("The order of a data queue is First In First Out (FIFO)"))
# ['queue']

print(multi_vowel_words("Hello world!"))
# []
```

Ejecutar
Restablecer

Correcto
¡Woohoo! En serio, tu trabajo es glorioso, notorio y ¡victorioso!

3. Al capturar grupos regex, ¿qué tipo de datos devuelve el método `groups`? **1 / 1 punto**

- Una cadena
- Una tupla
- Una Lista
- Un flotador

Correcto
Buen trabajo Como se devuelve una tupla, podemos acceder a cada índice individualmente.

4. La función `transform_comments()` convierte los comentarios de un script Python en comentarios utilizables por un compilador C. Esto significa buscar texto que empiece con una almohadilla (`#`) y sustituirla por barras dobles (`//`), que es el indicador de comentario de una sola línea de C. Para el propósito de este ejercicio, ignoraremos la posibilidad de una almohadilla incrustada dentro de un comando Python, y asumiremos que sólo se usa para indicar un comentario. También queremos tratar las marcas de almohadilla repetitivas (`##`), (`###`), etc., como un único indicador de comentario, que se sustituirá sólo por (`//`) y no por (`#//`) o (`//#`). Rellene los parámetros del método de sustitución para completar esta función: **1 / 1 punto**

```
import re
def transform_comments(line_of_code):
  result = re.sub(r'\#{1,}', r'//', line_of_code)
  return result

print(transform_comments("### Start of program"))
# Should be "// Start of program"
print(transform_comments("  number = 0   ## Initialize the variable"))
# Should be "  number = 0   // Initialize the variable"
print(transform_comments("  number += 1   # Increment the variable"))
# Should be "  number += 1   // Increment the variable"
print(transform_comments("  return(number)"))
# Should be "  return(number)"
```

Ejecutar
Restablecer

Correcto
¡Excelente! Ahora puedes convertir tus comentarios a otros lenguajes de programación, sólo tienes que convertir el código para que lo acompaña

5. La función `convert_phone_number()` busca un formato de número de teléfono estadounidense: XXX-XXX-XXXX (3 dígitos seguidos de un guión, 3 dígitos más seguidos de un guión, y 4 dígitos), y lo convierte a un formato más formal que tiene este aspecto: (XXX) XXX-XXXX. Rellene la expresión regular para completar esta función. **1 / 1 punto**

```
import re
def convert_phone_number(phone):
  result = re.sub(r"\b(\d{3})-(\d{3})-(\d{4})\b",r"(\1) \2-\3", phone)
  return result

print(convert_phone_number("My number is 212-345-9999.")) # My number is (212) 345-9999.
print(convert_phone_number("Please call 888-555-1234")) # Please call (888) 555-1234
print(convert_phone_number("123-123-12345")) # 123-123-12345
print(convert_phone_number("Phone number of Buckingham Palace is +44 303 123 7300")) # Phone number of Buckingham Palace is +44 30
```

Ejecutar
Restablecer

Correcto
Bien hecho Usted ha capturado los grupos adecuados para identificar lo que estamos buscando, ¡y nada más!