Universidad del Desarrollo
Facultad de Ingeniería

Taller de Programación

Certamen Recuperativo

17 de Diciembre del 2019

Instrucciones:

- El certamen contiene 4 problemas. Lea atentamente el enunciado de cada uno de ellos.
- Seleccione **tres** problemas de los enunciados 1,2, 3 y 4.
- Para cada problema cree un archivo.py distinto. El nombre del archivo debe ser el número del problema. Por ejemplo: uno.py, dos.py, tres.py o cuatro.py
- Suba sus respuestas como un archivo **ZIP** a la sección Evaluación en http://canvas.udd.cl. Solo tendrá una oportunidad para subir sus respuestas.
- Recuerde que usaremos un software de detección de plagio, confiamos en su honestidad.
- Tiempo total: **2 horas**.

1. Progresión de Paula (2 pts.)

La *progresión de Paula* corresponde a todos los números enteros cuyos dígitos se encuentran ordenados de menor a mayor. Por ejemplo:

> 11, 12, 13, 14, 15, 16, 17, 18, 19, 22, 23, 24, 25,..., 1224, 1225....

Cree un programa que calcule todos los números de la progresión de Paula contenidos entre 1 y `N`, donde `N` es ingresado por el usuario.

2. Frases Palíndromos (2 pts.)

Programe la función `check_palindromo` la que retorna `True` si una frase es palíndromo y `False` si no lo es. Una frase es palíndromo si se lee igual tanto de derecha a izquierda, como de izquierda a derecha, sin considerar los espacios en blanco. Ejemplo `luz azul`.

Nota:

- La función `s.replace(' ', '')` remueve los espacios en blanco del string s.

```
def check_palindromo(s):
    aux= s.replace(' ', '')
    if len(aux) < 2: return True
    if aux[0] != aux[-1]: return False
    return check_palindromo2(aux[1:-1])

print(check_palindromo1('luz azul'))
```

Puntaje

- 0.5 Elimina los espacios del texto
- 0.2 Considera el caso de palabras con un solo caracter
- 0.8 Compara la frase sin espacios con la la frase inversa
- 0.5 Retorna `True` si la frase es palindromo y `False` si no lo es

3. Desencriptación (2 pts.)

Programe una código que reciba una palabra encriptada ingresada por teclado e imprima la versión desencriptada. Para esto cree la función `desencriptar`, la que **recibe** un string encriptado y **retorna** un string desencriptado.

El proceso de desencriptación se describe a continuación:

1. Existe la palabra mágica `murcielago` de tipo tupla, que actua como clave de desencriptación, `clave=('m','u','r','c','i','e','l','a','g','o')`.

2. La palabra a desencriptar se recorre caracter por caracter, y si el caracter es un dígito, este se reemplaza por la letra de la clave en la posición que indica el dígito. Pero si el caracter no es un dígito, este se mantiene.

3. El resultado final de los reemplazos de caracteres, dan origen a la palabra desencriptada.

Ejemplo: La función `desencriptar('01nd9')` retorna `mundo`. Esto es porque `m` está en la posición `0` de la pálabra mágica, `u` está en la posición `1`, `n` y `d` no se encuentran, y `o` está en la posición `9`.

Notas:

- Asuma que sólo recibirá palabras en minúsculas.
- Dada una tupla `t`, la función `t.index(elem)` retorna la posición donde aparece el elemento `elem` dentro de la tupla `t`.
- La función `s.isdigit()` retorna `True` si el string s es un dígito y `False` si no lo es.

```
clave = tuple('murcielago')

def desencriptar(palabra):
    salida = []
    for i in range(len(palabra)):
        if palabra[i].isdigit()== True:
            salida.append(clave[int(palabra[i])])
        else:
            salida.append(str(palabra[i]))
    return ''.join(salida)

if desencriptar('01nd9') != 'mundo':
    print('Hay un error en tu código :(')

entrada = input('Ingrese palabra encriptada: ')
salida = desencriptar(entrada)
print('Palabra desencriptada:', salida)
```

Puntaje

- 0.5 Utiliza un ciclo para recorrer correctamente la palabra a desencriptar
- 0.5 Verifica si los elementos son dígitos
- 0.5 Reemplaza el caracter de la palabra encriptada cuando corresponde
- 0.5 Imprime o retorna la palabra desencriptada correctamente

4. Notas Finales Taller de Programación (2 pts.)

En el curso programación queremos conocer los resultados finales de los alumnos de forma rápida. En particular, nos interesa saber los nombres de todos los alumnos que obtuvieron una nota entre ciertos valores y la frecuencia de cada nota en el curso.

Utilizando los datos del archivo de texto `notas_praguadd.txt` y la función `get_dicc()`, la cual retorna un diccionario a partir del archivo de texto:

1. Programe la función `rango_alumnos(minimo, maximo, D)`, la cual recibe un valor mínimo, un valor máximo y un diccionario D y retorna una lista de strings con los nombres de todos los alumnos que tuvieron nota final en el curso entre los valores mínimo y máximo, segun los datos del diccionario D. (1.0 pt.)

2. Programe la función `frecuencia(nota, D)`, la cual debe recibir una nota y un diccionario D, y entregar la frecuencia de la nota final en el curso. (1.0 pt.)

Utilice el siguiente código como base de su solución:

```
def get_dicc(archivo):
    D = dict()
    f = open(archivo)
    for linea in f:
        valores = linea.split()
        D[valores[0]] = valores[1]
    f.close()
    return D

 def rango_alumnos(minimo, maximo, D):
    lista = []
    for alumno in D:
        notaAlumno = float(D[alumno])
        if minimo <= notaAlumno and notaAlumno <= maximo:
            lista.append(alumno)
    return lista

 def frecuencia(nota, D):
    resp = 0
    for alumno in D:
        if float(D[alumno]) == nota:
            resp = resp + 1
    return resp
```

Puntaje

- 1.0 Programa la función `rango_alumnos` correctamente
- 1.0 Programa la `frecuencia` correctamente