FACULTAD DE INGENIERÍA

FUNDAMENTOS DE COMPUTACION Y PROGRAMACION

PEP N°2

ASPECTOS GENERALES DE LA PRUEBA

- Queda prohibido hablar con los compañeros(as) durante el desarrollo de la PEP.
- La PEP contiene 3 preguntas de desarrollo, con un total de **45** puntos y una exigencia del **60%**
- Tiene un límite de tiempo de **90** minutos para responder.
- El equipo docente tiene la prohibición de responder consultas.
- El/La estudiante que se sorprenda en actos deshonestos será calificado con la nota mínima.
- Los elementos tecnológicos deben permanecer apagados y guardados. Queda absolutamente prohibido el uso todo elemento tecnológico. Su uso puede significar la nota mínima o sanciones mayores.
- El alumno deberá identificarse con su Cédula de Identidad.
- Sobre el escritorio sólo podrá existir lápiz (obligatorio) y goma/lápiz corrector (opcional).
- Lea atentamente la prueba antes de comenzar a desarrollarla.
- Complete sus datos personales antes de comenzar la evaluación.
- Considere que la evaluación contempla el código, los comentarios y el seguimiento de las buenas prácticas de programación.
- Responda cada pregunta, a continuación de su enunciado, en el espacio que se le entrega para ello.

NOMBRE	RUT	SECCIÓN	

1. **(15 puntos)** Se está construyendo un programa que permite a un usuario adelantar o retroceder los subtítulos de una película, hasta ahora se han desarrollado la mayoría de las funciones requeridas, sin embargo falta una, que el programador comentó del siguiente modo:

```
# Función que retrocede el temporizador en una cantidad determinada de
# segundos
# Entradas: String con formato (HH:MM:SS), Entero positivo (Cantidad de
# segundos a retroceder el tiempo)
# Salida: String con formato (HH:MM:SS) con el tiempo reducido
```

Implemente la función retrocederTiempo, considere para su desarrollo los siguientes ejemplos:

- Si las entradas fuesen “00:00:03” y 5, el programa debería entregar “00:00:00”
- Si las entradas fuesen “00:34:59” y 4, el programa debería entregar “00:34:55”
- Si las entradas fuesen “01:10:03” y 10, el programa debería entregar “01:09:03”
- Si las entradas fuesen “01:25:00” y 3600, el programa debería entregar “02:25:00”

```
# Función que descuenta una cantidad de segundos a un tiempo en particular
# Entradas: string en formato HH:MM:SS, cantidad de segundos a descontar
# Salida: string en formato HH:MM:SS con el tiempo descontado
def retrocederTiempo(tiempoTotal, segundosADescontar):
    # Se transforma el string en una lista usando el método split
    listaTiempo = tiempoTotal.split(":")
    # Se realiza el cambio de tipo de los valores de horas,
    # minutos y segundos de string a entero
    horas = int(listaTiempo[0])
    minutos = int(listaTiempo[1])
    segundos = int(listaTiempo[2])

    # Se calcula el tiempo total en segundos a partir de los valores de
    # horas, minutos y segundos
    tiempoTotalEnSegundos = horas * 3600 + minutos * 60 + segundos

    # Se descuenta el tiempo total gastado
    tiempoDescontado = tiempoTotalEnSegundos - segundosADescontar

    # En caso de que el tiempoDescontado resulte negativo
    if tiempoDescontado <= 0 :
        # El tiempo final tiene valor 00:00:00
        tiempoFinal = "00:00:00"
    # En caso contrario
    else :
        # Se transforma el tiempo descontado a horas, minutos y segundos
        # Se obtienen las horas dividiendo por 3600
        horas = tiempoDescontado / 3600
        # El tiempo restante es el resto de la división anterior
        tiempoDescontado = tiempoDescontado % 3600
        # Se obtienen los minutos dividiendo por 60
        minutos = tiempoDescontado / 60
        # Los segundos restantes son el resto de la división anterior
        segundos = tiempoDescontado % 60

        # Se transforman los valores numéricos de horas,
        # minutos y segundos a string
        stringHoras = str(horas)
        stringMinutos = str(minutos)
        stringSegundos = str(segundos)

        # Si los valores de hora, minutos o segundos son menores a 10,
        # se añade manualmente el 0 restante
        if horas < 10 :
            stringHoras = "0" + stringHoras
        if minutos < 10 :
            stringMinutos = "0" + stringMinutos
        if segundos < 10 :
            stringSegundos = "0" + stringSegundos
```

```
        # Se construye el string en formato HH:MM:SS
        tiempoFinal = stringHoras + ":" + stringMinutos + ":" + stringSegundos

    # Se entrega el tiempo final para todos los casos
    return tiempoFinal
```

2. **(15 puntos)** Una forma común de realizar campeonatos de fútbol es enfrentar a todos los equipos entre sí y entregar puntajes por el resultado de cada partido. De esta forma, al equipo ganador de un partido se le conceden 3 puntos, al perdedor se le conceden 0 puntos y en caso de empate, a ambos equipos se les otorga un punto. Al final del torneo se declara ganador y se premia al equipo con más puntos.

   Los organizadores de los torneos simplemente registran esta información y revisan finalmente los resultados para ver a los ganadores, sin embargo, se desea implementar un programa en Python que determine al campeón. Para ello considere como entrada el archivo "liga.txt" en el cuál se almacena el nombre del equipo, los puntos obtenidos y la diferencia de goles, separados por comas.

   Con esta información y el ejemplo que se suministra a continuación, construya un programa en Python que permita presentar por pantalla la información solicitada. (Considere que el archivo que se entrega y la salida generada son un ejemplo y que el programa debe funcionar también para casos distintos al del ejemplo).

   Para simplificar el problema, considere que no existe la posibilidad de que existan dos equipos que tengan el puntaje máximo.

ARCHIVO DE PUNTAJES (liga.txt)	SALIDA POR PANTALLA
Colegio Anglosajón,10 La Católica,2 Colegio Alemán,8 Colegio Superior,11 Franco-Canadiense,5 Furano,6 Niupi,14 San Francis,12	>>> RESULTADOS DE LA LIGA: CAMPEON: Niupi con 14 puntos

```
# Función que lee un archivo
# Entrada: nombre del archivo (string)
# Salida: Contenido del archivo (lista de strings)
def leerArchivo(nombreArchivo):
    # Se abre el archivo en modo de lectura
    archivo = open(nombreArchivo, 'r')
    # Con readlines se obtiene el contenido del archivo
    # como una lista de strings, dónde cada string es una línea del archivo
    contenido = archivo.readlines()
    # Se cierra el archivo
    archivo.close()
    # Se retorna el contenido del archivo
```

```
    return contenido

# Función que separa el nombre del equipo del puntaje obtenido
# Entrada: lista de strings
# Salida: lista de listas
def separarContenido(lista):
    # Se genera una nueva lista en blanco para guardar el contenido
    listaProcesada = []
    # Para cada elemento de la lista
    for elemento in lista :
        # Se separa el string equipo,puntaje a partir de la coma
        # quedando una lista [equipo, puntaje]
        nuevoElemento = elemento.split(",")
        # Se convierte el puntaje a entero para poder realizar comparaciones
        nuevoElemento[1] = int(nuevoElemento[1])
        # Se agrega el nuevo elemento a la lista
        listaProcesada.append(nuevoElemento)
    # Se entrega la listaProcesada
    return listaProcesada

# Función que determina el equipo con mayor puntaje
# Entrada: lista de listas
# Salida: lista con par equipo, puntaje
def determinarCampeon(lista):
    # Para comparar digo que el campeón será el primer equipo de la lista
    campeon = lista[0]
    # Para cada elemento en la lista
    for elemento in lista :
        # Si el puntaje del campeón es menor que el elemento a revisar
        if campeon[1] < elemento[1] :
            # El campeón es el elemento a revisar
            campeon = elemento
            # Tras recorrer el ciclo el programa se queda con el
            # elemento con mayor puntaje
    # Al salir del ciclo se retorna el campeón
    return campeon

# BLOQUE PRINCIPAL

# ENTRADAS

# Se lee el contenido del archivo
contenidoArchivo = leerArchivo("liga.txt")

# PROCESAMIENTO

# Se separan los equipos de los puntajes
contenidoProcesado = separarContenido(contenidoArchivo)
# Se determina al campeón
campeon = determinarCampeon(contenidoProcesado)
```

```
# SALIDA

# Se informa al usuario del equipo campeón
print "El campeón de la liga es", campeon[0],
print "con un total de", campeon[1], "puntos"
```

3. **(15 puntos)** Para ayudar con el cálculo de sueldos de una empresa, se desea construir un programa que entregue el sueldo líquido de todos los trabajadores de una empresa y los montos que se deben pagar por cada trabajador para cotizaciones previsionales y de salud.

Para implementarlo, se ha decidido realizar los cálculos con listas de largo *n* (dónde *n* es el número de trabajadores de la empresa), y en cada posición de las listas, se tienen los datos de un empleado en particular, se ha determinado que para el cálculo se requieren las siguientes listas:

- sueldoBase: Donde cada elemento representa el sueldo base, en pesos, de un trabajador.
- asignacionesImponibles: Donde cada elemento representa el monto en pesos de asignaciones imponibles del trabajador.
- porcentajePrevision: Donde cada elemento representa el porcentaje que debe destinarse a las cotizaciones previsionales (AFP).
- porcentajeSalud: Donde cada elemento representa el porcentaje que debe destinarse a cotizaciones de salud (Fonasa o Isapre).
- bonosNoImponibles: Donde cada elemento representa el monto en pesos que tiene el trabajador en bonos no imponibles, es decir, que no se consideran en el cálculo de las cotizaciones.

Para el cálculo se debe tomar la suma del sueldo base y las asignaciones imponibles, para calcular el total imponible, a dicho total se le debe descontar el porcentaje destinado a las cotizaciones de previsión y salud, para finalmente sumar los bonos no imponibles, resultando de ello el sueldo líquido. Considere que las listas tienen los siguientes formatos:

```
sueldoBase = [450000, 350000, 600000]
asignacionesImponibles = [20000, 30000, 15000]
porcentajePrevision = [11.44, 11.48, 10.77]
porcentajeSalud = [7, 8.2, 9.23]
bonosNoImponibles = [65000, 75000, 45000]
```

Y que en el primer caso las salidas serían

Sueldo Líquido: 448332
Monto de cotización previsional: 53768
Monto cotización de salud: 32900

Asumiendo que cada lista se solicita por entrada estándar (a través de input), construya un programa en Python que entregue el sueldo líquido y los montos de las cotizaciones previsionales y de salud.

```
# Desde numpy se importa el objeto array
from numpy import array

# ENTRADAS

# Se solicitan las entradas
sueldoBase = input("Ingrese lista de sueldos base: ")
asignacionesImponibles = input("Ingrese lista de asignaciones imponibles: ")
porcentajePrevision = input("Ingrese lista de porcentajes de descuento
   previsional: ")
porcentajeSalud = input("Ingrese lista de porcentajes de descuento de salud:
   ")
bonosNoImponibles = input("Ingrese lista de bonos no imponibles: ")

# PROCESAMIENTO

# Se transforman las listas en arrays de Numpy
arraySueldoBase = array(sueldoBase)
arrayAsignacionesImponibles = array(asignacionesImponibles)
arrayPorcentajePrevision = array(porcentajePrevision)
arrayPorcentajeSalud = array(porcentajeSalud)
arrayBonosNoImponibles = array(bonosNoImponibles)

#  Se calcula el total imponible
totalImponible = arraySueldoBase + arrayAsignacionesImponibles

# Se calcula el descuento previsional
descuentoPrevisional = totalImponible * (arrayPorcentajePrevision / 100)

# Se calcula el descuento de salud
descuentoSalud = totalImponible * (arrayPorcentajeSalud / 100)

# Se calcula el sueldo líquido
sueldoLiquido = totalImponible + arrayBonosNoImponibles - descuentoPrevisional
   - descuentoSalud

# SALIDA

# Se entregan las salidas solicitadas
print "Los sueldos líquidos a pagar son:"
print sueldoLiquido
print "Los montos previsionales a pagar son: "
print descuentoPrevisional
print "Los montos de salud a pagar son: "
print descuentoSalud
```