Instituto Tecnológico de Costa Rica
Ingeniería en Computación
Curso: Introducción a la Programación
Profesor: Ing. Cristian Campos Agüero

I Semestre 2020
16 Julio 2020
Total de puntos: 100

Examen Parcial 3

1. Sublistas de impares (25 puntos)

Escriba un programa con sintaxis Python cuya función principal se llame **susbistas_impares**(lista), que reciba como entrada una **lista con número enteros positivos** denominada lista y que retorne una lista, donde por cada número impar, se genera una sublista de los dígitos del número.

Ejemplos del comportamiento de la función:
>>> susbistas_impares ([23, 6756, 5533, 811])
[[2,3], [5,5,3,3], [8, 1, 1]]
>>> susbistas_impares ([152, 45, 60])
[[4,5]]
>>> susbistas_impares ([])
"Error en la entrada, no se permite lista vacía"

2. Número Computable (25 puntos)

Escriba una función en Python llamada **es_computable**(num), que recibe un número entero positivo mayor que cero. La función debe indicar si el número es Computable o no. Un número es Computable si cumple con lo siguiente:

- Al menos la mitad de los dígitos del número ingresado son impares.
- La representación en sistema quinario[1] (base cinco) del número ingresado inicia y termina con el mismo dígito.

Ejemplos del comportamiento de la función:

>>> es_computable(332) #en sistema quinario es 2312
True
>>> es_computable(6877) #en sistema quinario es 210002
True
>>> es_computable(33) # en sistema quinario es 113
False

3. Matriz diamante (25 puntos)

Escriba una solución con sintaxis Python cuya función principal se llame **matriz_diamante(matriz)**, recibe como parámetro de entrada una **matriz cuadrada** impar donde la **cantidad de mínima de filas y columnas debe ser 5**, y devolverá **True** en caso de que todos los valores internos de la matriz formen **un rombo o diamante con ceros**, **False** en caso

[1] https://es.wikipedia.org/wiki/Sistema_quinario

de que al menos un valor no sea cero dentro del rombo. Los valores internos son aquellas posiciones que forman parte del rombo o diamante de la matriz. Puede asumir la existencia de una función validaMatriz(matriz) que retorna **True** si la entrada es una matriz válida. No puede descomponer la matriz de entrada.
Dado la siguiente matriz de entrada:

$$\text{matriz} = \begin{bmatrix} 9 & 2 & \mathbf{0} & 3 & 4 \\ 3 & \mathbf{0} & \mathbf{0} & \mathbf{0} & 9 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 4 & \mathbf{0} & \mathbf{0} & \mathbf{0} & 3 \\ 1 & 3 & \mathbf{0} & 9 & 1 \end{bmatrix}$$

El resultado sería:

True

En Python:

>>> **matriz_ diamante (matriz)**
True

$$\text{matriz} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

4. Máxima Profundidad de Lista (25 puntos)

Escriba una solución con sintaxis Python cuya función principal se llame **max_profundidad_lista(lista)**, el parámetro de entrada es una **lista**, **no vacía**, cuyos elementos deben ser **listas, con profundidades variadas** (contienen listas sucesivamente). La máxima profundidad de las sublistas puede tener elementos heterogéneos. La función debe indicar el camino de la máxima profundidad de sublistas, "de arriba hasta abajo", en forma de lista.
Ejemplos del comportamiento de la función:

```
>>> max_profundidad_lista ([[["H"], [["M"],[[12]]], [13]], [[45]], []])
[[['H'], [['M'], [[12]]], [13]], [['M'], [[12]]], [[12]], [12]]
>>> max_profundidad_lista ([[["Hola"]], "Mundo"])
[[["Hola"]], ["Hola"]]
>>> max_profundidad_lista ([["Distanciamiento"], ["Social"]]) #retorna cualquiera de las dos
[["Distanciamiento"]]
```

Para reflexionar:

"Siempre estoy haciendo cosas que no puedo hacer. Así es como consigo hacerlas"
Pablo Picasso.