COMPUTACIÓN 1
Instituto de Computación

Examen – 14 de febrero de 2017

- Duración de esta prueba: 3 Hs.
- No se podrá utilizar ningún tipo de material (apuntes, libro, calculadora, etc). Apague su celular.
- **Sólo** se contestarán preguntas sobre interpretación de la letra hasta 30 minutos antes de la finalización del mismo.
- Las partes no legibles del parcial se considerarán no escritas.
- En la primer hoja escriba con LETRA CLARA, en el ángulo superior derecho, su nombre, número de cédula de identidad y cantidad de hojas -en ese orden-; las demás hojas es suficiente con nombre, número de cédula y número de página.

Para la resolución de los diferentes ejercicios **solamente** podrá utilizar las siguientes funciones brindadas por **Octave**:

- length() y size()
- mod() y rem()
- floor(), ceil() y round()
- zeros() y ones()

Problema 1	15 ptos (1,2,3,5,4)	

a) Calcule la expresión decimal del siguiente número binario puro: 10001.
b) Cuantos bits necesita para poder representar correctamente el número 60 en complemento a 2.
c) Represente en octal el siguiente número F1239 representado en hexadecimal.
d) Determine la expresión en el sistema complemento a 1 con 10 bits de la tira
0 10000100 01000000000000000000000 en punto flotante simple precisión.
e) Determine la representación en el sistema de punto flotante simple precisión de 7×2^{-4}.

Nota: Justificar todas las respuestas.

Problema 2	23 ptos	

Implemente una función **iterativa** en Octave *encontrar* que reciba una matriz y un número y devuelva dos variables con la fila y columna de la primera ocurrencia del número en la matriz. En caso de no encontrar el número debe devolver 0 como valores de fila y columna. La matriz debe recorrerse por filas.

Problema 3	36 ptos (7,8,11,10)	

Se dispone de una matriz cuadrada y simétrica **MD** que almacena las distancias (valores positivos) entre distintos puntos de una ciudad. Los puntos están numerados del 1 a la dimensión de la matriz y la distancia del punto i al j se almacena en **MD(i,j)**. Si no existe camino directo entre i y j, **MD(i,j)=-1**.
También representaremos una ruta como un vector donde cada coordenada es un punto de la ciudad a visitar.

Ejemplo:

MD=

0	1	2	3
1	0	5	6
2	5	0	-1
3	6	-1	0

ruta1 =

3	2	4	1	2	4

ruta2 =

3	2	4	3	2

En este ejemplo la distancia entre el punto 1 y el 3 es igual a 2, la distancia entre el 2 y el 3 es igual a 5, y no hay camino que una los puntos 3 y 4, por lo que la distancia es -1.
La ruta1 comienza en el punto 3, pasando por 2, 4, 1, volviendo a 2 y terminando en el punto 4.
La ruta2 no se puede realizar dado que no hay camino entre 4 y 3.

a) Implementar en Octave una función **iterativa** *distIt* que reciba la matriz de distancias y un vector de ruta y devuelva la distancia del camino. Si no existe ruta entre los dos puntos del camino, debe devolver -1.

b) Implementar en Octave una función **recursiva** *distRec* que reciba la matriz de distancias y un vector de ruta y devuelva la distancia del camino. Si no existe ruta entre los dos puntos del camino, debe devolver -1.

c) Se define una matriz de rutas como una matriz donde cada fila almacena un vector de ruta de la siguiente manera: en la primera posición se encuentra el largo del vector y desde la posición 2 en adelante el vector de ruta propiamente dicho. Las posiciones más allá del largo de cada vector pueden contener cualquier valor.

Ejemplo:

MR =

4	1	2	4	3	x	x	x
2	1	3	x	x	x	x	x
7	3	1	4	2	3	2	4
3	2	3	1	x	x	x	x

En este ejemplo la matriz de rutas MR contiene 4 vectores de ruta:
[1,2,4,3] (largo 4)
[1,3] (largo 2)
[3,1,4,2,3,2,4] (largo 7)
[2,3,1] (largo 3)

Implementar en Octave una función **iterativa** *esElMenor* que reciba la matriz de distancias **MD**, un vector de una ruta **v** y una matriz de rutas **MR**, y devuelva 1 si el vector de ruta ingresado es de menor distancia que todos los almacenados en la matriz y 0 en caso contrario.

d) Implementar en Octave una función **recursiva** *obtenerMayor* que reciba una matriz de distancias **MD** y una matriz de rutas **MR** como la de la parte anterior y devuelva el vector de ruta de mayor distancia. En caso que ninguno de los vectores de la matriz contenga una ruta válida, la función puede devolver cualquiera de los vectores.

Nota: Es válido utilizar las funciones de partes anteriores si lo cree conveniente.

Problema 4	26 (8, 8, 10) ptos	

Teniendo en cuenta lo planteado en el Problema 3, imagine ahora que los puntos de la matriz de distancias representan esquinas de una ciudad y el elemento M(i,j) representa la longitud de la cuadra que conecta la esquina i con la esquina j. Como cada esquina está conectada directamente con otras 4 o 5 esquinas y la cantidad de esquinas de una ciudad es muy grande, representaremos la matriz de distancia como una matriz dispersa donde M(i,j)=0 indica que las esquinas i y j no son adyacentes.

a) Implementar en Octave la función **iterativa** *longCuadraIt* que recibe dos esquinas i y j y la matriz de distancias en formato disperso elemental y devuelve la longitud de la cuadra que conecta i y j.

b) Implementar en Octave la función **recursiva** *longCuadraRec* que recibe dos esquinas i y j, y la matriz de distancias en formato disperso elemental, y devuelve la longitud de la cuadra que conecta i y j.

c) Implementar en Octave la función **recursiva** *actualizarCuadraRec* que recibe dos esquinas i y j, una nueva distancia d, y la matriz de distancias en formato disperso elemental, y devuelve la matriz de distancia actualizada con la nueva distancia. Notar que si ya se tenía la distancia entre los puntos i y j se debe cambiar la distancia y si no existía se debe agregar.