COMPUTACIÓN 1
Instituto de Computación

1er Parcial - 6 de octubre de 2018

- Duración del parcial: 3:00 Hs.
- No se podrá utilizar ningún tipo de material (apuntes, libro, calculadora, etc). Apague su teléfono celular.
- **Sólo** se contestarán preguntas sobre interpretación de la letra hasta 30 minutos antes de la finalización del mismo.
- Escriba las hojas de un solo lado. Las partes no legibles del examen se considerarán no escritas
- En la primera hoja a entregar ponga con letra clara, en el ángulo superior derecho, su **nombre**, número de **cédula de identidad** y **cantidad de hojas** -en ese orden-; en las demás hojas alcanza con poner 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	2 (1, 1) ptos	

a) Escriba en octal el número 10101_2 que está representado en binario.

b) Escriba en binario el número 11011_{16} que está representado en hexadecimal.

Problema 2	3 ptos	

Dado mi_script y func como:

```
% mi_script.m
x = x + 1;
z = x + 1;
```

```
% func.m
function z = func (x)
x = x + 1;
z = x + 1;
```

En la consola se escribe lo siguiente:

```
>> clear all;% borra todas las variables del espacio de trabajo
>> x = 1;
>> y = 0;
>> mi_script;
```

a) ¿Qué valor tienen *x, y* y *z* después de la ejecución de mi_script? Justifique su respuesta.

En la consola se escribe lo siguiente:

```
>> clear all; % borra todas las variables del espacio de trabajo
>> x = 1;
>> y = 0;
>> y = func(x);
```

b) ¿Qué valor tienen *x, y* y *z* después de la ejecución de func? Justifique su respuesta.

Nota: En todos los ejercicios se deben usar las estructuras de control correctas para cada caso.

Problema 3	14 (6,8) ptos	

a) Implementar en *Octave* la función **partir_en_dos** que dado un vector *v*, devuelva dos vectores, uno con los 10 primeros elementos múltiplos de 3 y otro con el resto de los elementos de *v*.

Ejemplos:

```
>> [a,b] = partir_en_dos([])
a = []
b = []

>> [a,b] = partir_en_dos([3,3,2,7,9,2,8])
a = [3,3,9]
b = [2,7,2,8]

>> [a,b] = partir_en_dos([3,3,2,7,9,2,8,6,12,15,4,3,9,2,21,12,10,6,33])
a = [3,3,9,6,12,15,3,9,21,12]
b = [2,7,2,8,4,2,10,6,33]
```

b) Implementar en *Octave* la función **suma_celda** que reciba una matriz *M* y dos índices *i* y *j*. La función devuelve 1 si la suma de los valores de la fila *i* y la columna *j*, sin contar a *M(i,j)*, es mayor que el valor de *M(i,j)*, y 0 en caso contrario. La matriz *M* puede tener valores postivos y negativos.

Problema 4	8 ptos	

Se define el producto cartesiano de dos vectores como la matriz con todos los pares que pueden formarse de forma que el primer elemento pertenezca al primer vector y el segundo elemento pertenezca al segundo vector.

Implementar en *Octave* la función **producto_cartesiano**, que reciba dos vectores, *v* y *w*, y devuelva una matriz de dos columnas con el resultado de realizar el producto cartesiano de ambos.

Ejemplos:

```
>> producto_cartesiano([], [])
ans = []

>> producto_cartesiano([], [2, 5])
ans = []

>> producto_cartesiano([4], [2, 5])
ans =
   4   2
   4   5
```

```
>> producto_cartesiano([7, 3, 7], [2, 5])
ans =
   7   2
   7   5
   3   2
   3   5
   7   2
   7   5
```

Nota: En todos los ejercicios se deben usar las estructuras de control correctas para cada caso.

Problema 5	13 (6, 7) ptos	

Una tienda vende repuestos de autos de distinto tipo. Cada tipo de repuesto tiene un número único ***i*** que lo identifica. Cada vez que llega un pedido, se controla la existencia de los repuestos solicitados y se le informa al cliente si es posible satisfacer su pedido. El vector *D* contiene cuántos elementos de cada repuesto hay en existencia, es decir que *D(i)* contiene la cantidad disponible del repuesto *i* en existencia. A su vez, para cada cliente se genera un vector *S*, de igual largo que *D*, que contiene la cantidad de cada repuesto que quiere comprar. Si *S(i)* vale 8, esto significa que quiere comprar 8 repuestos *i*. Cuando un tipo de repuesto *i* no es parte del pedido, el elemento *S(i)* correspondiente tiene valor 0 (cero).

a) Implementar en *Octave* la función **sePuedeRealizarPedido**, que reciba un vector *S* con la cantidad a solicitar de cada producto y un vector *D* con la cantidad disponible de cada producto, y devuelva 1 si es posible realizar el pedido y 0 en caso contrario (basta con que no haya suficiente disponibilidad de un producto para que no se pueda realizar el pedido). El largo de ambos vectores es el mismo y está dado por la cantidad de productos distintos pasibles de ser inventariados y vendidos.

Ejemplos:

```
>> sePuedeRealizarPedido([2,3,5], [5,2,15])
ans = 0

>> sePuedeRealizarPedido([2,3,5], [5,3,15])
ans = 1
```

Cuando no es posible realizar un pedido, es posible informar al cliente la cantidad de días necesarios para contar con los repuestos en la tienda y poder satisfacer su pedido. El vector *R*, de igual largo que *S* y *D*, contiene cuántos días se demora en reponer cada repuesto, es decir que *R(i)* contiene la cantidad de días en reponer la existencia del repuesto *i*.

b) Implementar en *Octave* la función **diasParaRealizarPedido**, que reciba un vector *S*, un vector *D*, y un vector *R* con la cantidad de días que faltan para reponer cada producto, y devuelva la cantidad de días necesarios para poder realizar el pedido. En caso de que existan en inventario la cantidad de repuestos requeridos, la función retornará 0 (cero).

Ejemplo:

```
>> diasParaRealizarPedido([2,3,5], [5,2,1], [1,5,15])
ans = 15

>> diasParaRealizarPedido([2,3,5], [5,3,5], [1,5,15])
ans = 0
```