COMPUTACIÓN 1
Instituto de Computación

# Primer Parcial – Modalidad Presencial - 27 de setiembre de 2021

- Duración del parcial: 2:00 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.
- 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 las demás hojas pongan 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()`

**Notas: - En todos los ejercicios se deben usar las estructuras de control adecuadas para cada caso. Por ejemplo: se controlará el uso correcto de for y while.**
**- No se deben realizar más iteraciones que las necesarias para resolver el problema**

| **Problema 1** | 4 (2, 2) ptos | |
|---|---|---|

a) Reescribir la siguiente expresión lógica sin utilizar el operador & (o &&):

```
r = (x && ~y) || (~x && y)
```

b) Reescribir el siguiente código utilizando combinaciones de if, elseif, else, end y sin utilizar operadores lógicos como & (o &&), | (o ||) ni ~ :

```
function res = expr(x,y)

  if (x && ~y) || (~x && y)
     res = 1;
  else
     res = 0;
  end
end
```

| **Problema 2** | 4 ptos | |
|---|---|---|

Determine el valor de las variables `a,b` y `c` luego de ejecutar `miscript.m` desde la línea de comandos de octave.

```
% funcAux.m

function c = funcAux(b,a)
  c = 1;
  n = a;
  for a = 1:n
    c = c + b;
  end
end
```

```
% miscript.m

c = 1;
a = 2;
b = funcAux(a,6);
a = a + 1;
```

| **Problema 3** | 7 ptos | |
|---|---|---|

Implementar en *Octave* la función ***iterativa*** *cantidadValores* que, dado un número entero *n* mayor que cero y un vector *v* que contiene valores entre 1 y *n*, retorne un vector *v2* de *n* elementos tal que *v2(pos)* contenga la cantidad de ocurrencias de *pos* en *v.*

Ejemplos de ejecución (los ejemplos siguen el formato cantidadValores(n, v) → v2)

cantidadValores(4, []) → [0,0,0,0]
cantidadValores(4, [1,3,3,3,1,3,1,2]) → [3,1,4,0]
cantidadValores(10, [1,3,3,3,1,3,1,2]) → [3,1,4,0,0,0,0,0,0,0]
cantidadValores(10, [4,2,3,1,9,8,7,6,5]) → [1,1,1,1,1,1,1,1,1,0]

| **Problema 4** | 7 ptos | |
|---|---|---|

Implementar en *Octave* la función ***iterativa*** *dosIgualesConsecutivos* que, dado un vector *x*, devuelva 1 si el vector tiene **al menos** dos elementos iguales en forma consecutiva. En caso contrario, la función deberá devolver 0.

dosIgualesConsecutivos([]) → 0
dosIgualesConsecutivos ([0,1,0,1,0]) → 0
dosIgualesConsecutivos ([0,1,1,0,3,4]) → 1
dosIgualesConsecutivos ([0,1,1,1,0,3,4]) → 1
dosIgualesConsecutivos ([0,1,0,3,4,4]) → 1

| **Problema 5** | 8 ptos | |
|---|---|---|

Implementar en *Octave* la función ***iterativa*** *cantidadParesMatriz* que, dada una matriz *Mx* que contiene elementos enteros mayores a 0, calcule la cantidad de los elementos de la matriz que son pares.

```
Mx = [ 2, 4, 1 ,3 , 1;
       3, 1, 5, 1, 7;
       3, 3, 6, 4, 6];
cantidadParesMatriz(Mx) → 5
```