

Examen - 28 de diciembre de 2020
Parte 1 – 22 puntos

- Duración de esta parte de la prueba: 45 minutos.
- Duración total incluyendo descansos: 200 min (45' - 10' - 75' - 10' - 60') = 3h:20min
- Puntaje total de la prueba: 100 puntos.
- 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.

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.
- No se deben realizar más iteraciones o invocaciones recursivas que las necesarias para resolver el problema

| | | |
|-------------------|--------------------------|--|
| Problema 1 | 8 (1, 1, 1, 1, 1, 3) pts | |
|-------------------|--------------------------|--|

- ¿Cuáles de los sistemas de representación de enteros vistos en el curso (exceso a M, Complemento a 1, Complemento a 2, signo y valor absoluto) preservan el orden de los números representados?
- ¿De qué formas se puede representar el 0 en Complemento a 1 de 8 bits?
- Represente el número 64 en Complemento a 2 de 7 bits
- Convierta el negativo del dígito de verificación de su cédula, es decir $-x$ donde x es el dígito verificador de su cédula, a Complemento a 1 de 5 bits.
- Calcule $11101 - x$, donde x es el dígito verificador de su cédula, utilizando Complemento a 1 de 5 bits.
- Sea $a=011101111111$ un número en punto flotante con 1 bit de signo, 4 bits de exponente y 7 bits de mantisa, donde el exponente 1111 se reserva para Inf y NaN, y el 0000 se reserva para números desnormalizados.

Encuentre un número positivo b normalizado para el cual $a+b=a$ en la representación anterior.

Sugerencia: tenga en cuenta que, en esta representación, a es el número positivo normalizado más grande.

| | | |
|-------------------|-----------------|--|
| Problema 2 | 7 (1.5,5.5) pts | |
|-------------------|-----------------|--|

El siguiente código intenta calcular, a partir de dos vectores del mismo largo, un vector que contiene el mínimo entre los dos vectores posición a posición.

```
1 - function v = min2Vect(x,y)
2 -   lx=length(x);
3 -   ly=length(y);
4 -   if (lx == 0)
5 -     v = [];
6 -   else
7 -     v = min2Vect(x(2:lx),y(2:ly));
8 -   end
9 - end
```

Parte a) Si se ejecuta la función con los parámetros $x = [-3,9]$ e $y = [5,9]$, las invocaciones recursivas que se hacen son las siguientes:

Invocación 1: `min2Vect([-3,9],[5,9])`
-> Invocación 2: `min2Vect([9],[9])`
-> Invocación 3: `min2Vect([],[])`

- a.1) (0.5 puntos) ¿Cuál es el resultado que devuelve Invocación 3?
- a.2) (0.5 puntos) ¿Cuál es el resultado que devuelve Invocación 2?
- a.3) (0.5 puntos) ¿Cuál es el resultado que devuelve Invocación 1?

Parte b) Corrija el código para que la función recursiva cumpla su cometido.

| | | |
|-------------------|-------|--|
| Problema 3 | 7 pts | |
|-------------------|-------|--|

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

| | |
|--|--|
| <pre>% funcAux.m function c = funcAux(a,b) c = 1; for i=1:a c = c+b; end end</pre> | <pre>% miscript.m c = 10; b = funcAux(3,2); a = b + 1;</pre> |
|--|--|