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

Problema 1	8 (1, 1, 1, 1, 1, 3) ptos	

a) Exceso a M

b) 00000000 y 11111111

c) El número 64 no se puede representar en Complemento a 2 de 7 bits. Es mayor que el máximo número representable.

d) y e)

Todos los dígitos son representables con 4 bits. Para calcular el Complemento a 1 del dígito de la cédula, simplemente se expresa el número en binario con 5 bits, sustituyendo 0's por 1's y viceversa. Para realizar la resta, como ambos números son negativos, simplemente se realiza la suma en binario de sus expresiones en Complemento a 1. En caso de finalizar la operación con acarreo, se lo suma al digito de mas a la derecha.

x	Ca1(-x)	11101-x=11101+Ca1(-x)
0	00000/11111	11101+00000=11101 / 11101+11111=11100+1=11101
1	11110	11101+11110=11011+1=11100
2	11101	11101+11101=11010+1=11011
3	11100	11101+11100=11001+1=11010
4	11011	11101+11011=11000+1=11001
5	11010	11101+11010=10111+1=11000
6	11001	11101+11001=10110+1=10111
7	11000	11101+11000=10101+1=10110
8	10111	11101+10111=10100+1=10101
9	10110	11101+10110=10011+1=10100

f) **b** debe ser un numero muy pequeño de forma que, al igualar los exponentes durante la suma, la coma se desplace más lugares que los disponibles para almacenar la mantisa.

Por ejemplo, podemos tomar **b** = 000010000001. En este caso, la diferencia entre los exponentes de **a** y **b** es 1110_2-0001_2=1101_2=13_{10}. Para sumar **a+b**, debemos desplazar la mantisa del número más pequeño (**b**) 13 lugares hacia la derecha y luego sumar las mantisas. El exponente del resultado será igual al de **a.** Luego de desplazada, la mantisa de **b** queda: 0.00000000000010000001. Al sumar ambas mantisas obtenemos 1.11111110000010000001. Luego, al truncar la mantisa para almacenarla con 7 bits, se obtiene 1.1111111, que es igual a la mantisa de **a**. Por lo tanto, en esta representación, **a+b=a**.

Problema 2	7 (1.5,5.5) ptos	

Parte a.1) []
Parte a.2) []
Parte a.3) []

Parte b)

```
function v = min2vect(x,y)
      lx = length(x);
      ly = length(y);
      if (lx == 0)
            v = []
      else
            v = min2vect(x(2:lx), y(2:ly));
            if (x(1) < y(1))
                  v = [x(1) v];
            else
                  v = [y(1) v];
            endif
      endif
end
```

Problema 3	7 ptos	

```
c= 10
b= 7
a= 8
```

Parte 2: Iteración – 42 puntos

Problema 4	14 ptos	

```
function res = primeraFilaQueSumaX(M, X)
   [lf,lc] = size(M);
   fila = 1;
   res = -1;
   while (fila <= lf && res == -1)
      suma = 0;
      for col=1:lc
            suma = suma + M(fila,col);
      end
      if (suma == X)
            res = fila;
      else
            fila = fila + 1;
      end
   end
end
```

Problema 5	14 ptos	

```
function saltos = saltaHastaRepetir(p, posiciones)
   lp = length(posiciones);
   repes = zeros(lp, 1);
   saltos = 0;
   while (repes(p) == 0)
      repes(p) = 1;
      p = posiciones(p);
      saltos = saltos + 1;
   endwhile
end
```

Problema 6	14 ptos	

```
function [fd, cd, dd] = multiplicacionElementoaElementoFiltrada(A, B,
                                                                fs, cs, ds)
   l = length(fs);
   fd = [];
   cd = [];
   dd = [];
   for x=1:l
      prod = A(fs(x),cs(x)) * B(fs(x),cs(x));
      if (prod ~= 0)
            fd = [fd fs(x)];
            cd = [cd cd(x)];
            dd = [dd prod];
      endif
   endfor
end
```

Parte 3: Recursión – 36 puntos

Problema 7	12 ptos	

```
function rec = recortarMascara(v, mascara)
      lv = length(v);
      if (lv == 0)
            rec = [];
      else
            rec = recortarMascara(v(2:lv), mascara(2:lv));
            if (mascara(1)~=1)
                  rec = [v(1) rec];
            endif
      endif
end
```

Problema 8	12 ptos	

```
function suma = sumarPosiciones(v, posiciones)
      lp = length(posiciones);
      if (lp == 0)
            suma = 0;
      else
            suma = sumarPosiciones(v, posiciones(2:lp));
            suma = suma + v(posiciones(1));
      endif
end
```

Problema 9	12 ptos	

```
function [res0, res1, res2] = triptico(N)
   if (N == 1)
      res0 = [];
      res1 = [1];
      res2 = [];
   else
      [res0, res1, res2] = triptico(N-1);
      resto = mod(N, 3);
      if (resto == 0)
            res0 = [res0, N];
      elseif (resto == 1)
            res1 = [res1, N];
      else
            res2 = [res2, N];
      endif
   endif
end
```