

## Solución Problema 1)

Teórico

## Solución Problema 2)

### Parte a)

```
function s = es_salida(M, pos)
[m,n] = size(M);
x = pos(1);
y = pos(2);
s = x==1 | x==m | y==1 | y==n;
```

### Parte b)

```
function [exito, destino] = movimiento(M, pos, mov)
[m,n] = size(M);
fila = pos(1);
col = pos(2);
exito = 0;
destino = [0,0];
switch mov
case 8 % ARRIBA
    if fila > 1 && fila <= m
        if M(fila-1, col) == 0
            exito = 1;
            destino = [fila-1, col];
        end
    end
case 2 % ABAJO
    if fila >= 1 && fila < m
        if M(fila+1,col) == 0
            exito = 1;
            destino = [fila+1,col];
        end
    end
case 4 % IZQUIERDA
    if col > 1 && col <= n
        if M(fila,col-1) == 0
            exito = 1;
            destino = [fila,col-1];
        end
    end
case 6 % DERECHA
    if col >= 1 && col < n
        if M(fila,col+1) == 0
            exito = 1;
            destino = [fila,col+1];
        end
    end
end
```

**Parte b) (solución alternativa)**

```
function [exito, destino] = movimiento(M, pos, mov)
[m,n] = size(M);
fila = pos(1);
col = pos(2);
exito = 0;
destino = [0,0];
if mov==8 % ARRIBA
    if fila > 1 && fila <= m
        if M(fila-1, col) == 0
            exito = 1;
            destino = [fila-1, col];
        end
    end
elseif mov==2 % ABAJO
    if fila >= 1 && fila < m
        if M(fila+1,col) == 0
            exito = 1;
            destino = [fila+1,col];
        end
    end
elseif mov==4 % IZQUIERDA
    if col > 1 && col <= n
        if M(fila,col-1) == 0
            exito = 1;
            destino = [fila,col-1];
        end
    end
elseif mov==6 % DERECHA
    if col >= 1 && col < n
        if M(fila,col+1) == 0
            exito = 1;
            destino = [fila,col+1];
        end
    end
end
end
```

**Parte c)**

```
function [exito, destino] = verificar_camino(M, pos, movs)
l = length(movs);
if l==0 % NO HAY MOVIMIENTOS
    exito = 1;
    destino = pos;
else
    [exito,destino] = movimiento(M, pos, movs(1));
    if exito
        [exito, destino] = verificar_camino(M, destino, movs(2:l));
    end
end
```

### Solución Problema 3)

#### Parte a)

```
Pi = [1,2,3];  
Pj = [3,1,2];  
Pv = [1,1,1];
```

#### Parte b)

```
function M = permutadc(Pi, Pj, Pv, A)  
LP = length(Pi); % LP es también la cantidad de filas de A  
for i = 1:LP  
    h = Pi(i);  
    k = Pj(i);  
    % hay un 1 en el lugar (h,k) de la matriz P, esto significa que  
    % tengo que permutar la fila k por la fila h en el resultado  
    M(h,:) = A(k,:);  
end
```

#### Parte c)

```
function [Mi, Mj, Mv] = permutardd(Pi, Pj, Pv, Ai, Aj, Av)  
LP = length(Pi);  
LA = length(Ai);  
Mi = [];  
Mj = [];  
Mv = [];  
for x = 1:LP  
    h = Pi(x);  
    k = Pj(x);  
    % un 1 en el lugar (h,k) de la matriz P indica que los elementos  
    % de la fila k deben pasar a la fila h en el resultado  
    for y = 1:LA  
        if k == Ai(y) % es un elemento que esta en la fila k?  
            Mi = [Mi h]; % lo pasamos para la fila h  
            Mj = [Mj Aj(y)]; % y lo dejamos en la misma columna  
            Mv = [Mv Av(y)];  
        end  
    end  
end
```

### Solución Problema 4)

#### Parte a)

```
function v = separardigitos(n)  
v = [];  
while n >= 10  
    v = [mod(n,10), v];  
    n = floor(n/10);  
end  
v = [n, v];
```

#### Parte b)

```
function s = secuencia(n)  
v = separardigitos(n);  
s = sum(v.*v);
```

**Parte c)**

```
function [f,s] = esfeliz(n, tope)
if n == 1
    f = 1;
    s = [1];
elseif tope == 0
    f = 0;
    s = [n];
else
    siguiente = secuencia(n);
    [f,s] = esfeliz(siguiente, tope - 1);
    s = [n, s];
end
```