```
mat1 = array[1..8,1,..4]of integer
lista_str = ARRAY [1..8] of string[7]
lista_chr = ARRAY [1..8] of char
mat2 = ARRAY [1..3,1..2] of string;
```

camiones:mat1

	bruto	tara	neto	producto(1,2 ó 3)
1				
2				
3				
4				
5				
6				
7				
8				

cupos:lista_str

	patentes	estados:lista_chr

camm:mat1

	cont	acum	min	max
1				
2				
3				

patente_mm:mat2

	min	max

prod:lista_str

	productos

TRABAJO PRÁCTICO Nº 2

PROGRAMA PRINCIPAL

```
type LISTA_STR = ARRAY[1..8] OF STRING[7];
     LISTA_CHR = ARRAY[1..8] OF CHAR;
     MAT1 = ARRAY[1..8,1..4] OF INTEGER;
     MAT2 = ARRAY[1..3,1..2] OF STRING;

VAR CAMIONES,CAMM: MAT1;
    CUPOS,PROD: LISTA_STR;
    ESTADO: LISTA_CHR;
    PAT_MM: MAT2;
    [redacted]: INTEGER;
```

```
Inicio()
MENU_PRINCIPAL()
```

```
Inicio()
VAR I,J: INTEGER;
FOR I ← 1 TO 8
    ESTADO[I] ← ' '
FOR I ← 1 TO 3
    FOR J ← 1 TO 2
        PAT_MM[I,J] ← ' '
InicioSTR(CUPOS,8)
InicioSTR(PROM,3)
InicioINT(CAMIONES,8,4)
InicioINT(CAMM,3,4)
```

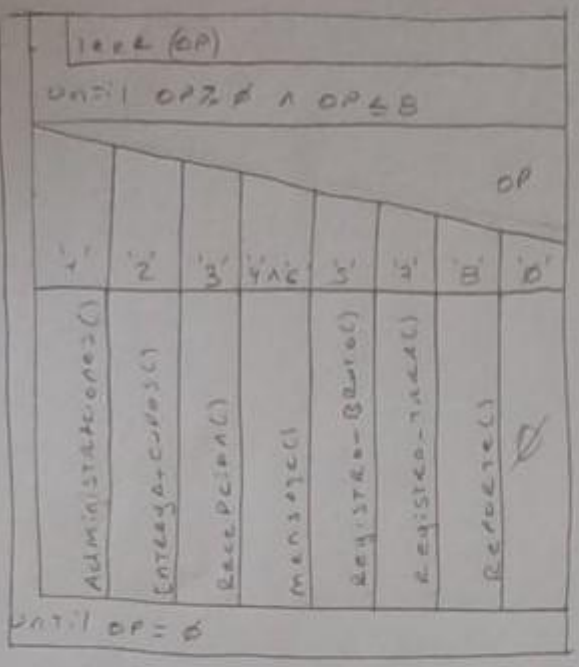
```
InicioSTR(VAR X: LISTA_STR, N: INTEGER)
VAR I: INTEGER;
FOR I ← 1 TO N
    X[I] ← ' '
```

```
InicioINT(VAR X: MAT1, N,M: INTEGER)
VAR I,J: INTEGER;
FOR I ← 1 TO N
    FOR J ← 1 TO M
        X[I,J] ← 0
```

```
MENSAJE()
MOSTRAR('EN CONSTRUCCIÓN')
```

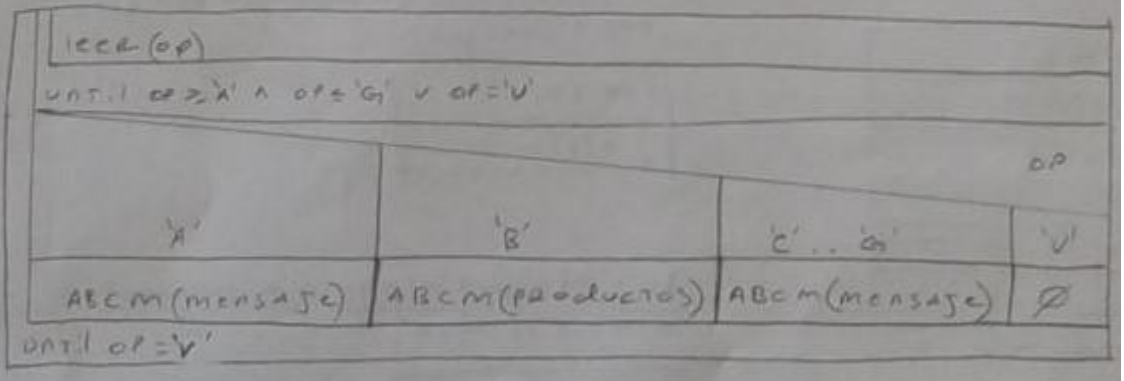
MENU_PRINCIPAL()

VAR OP: CHAR;



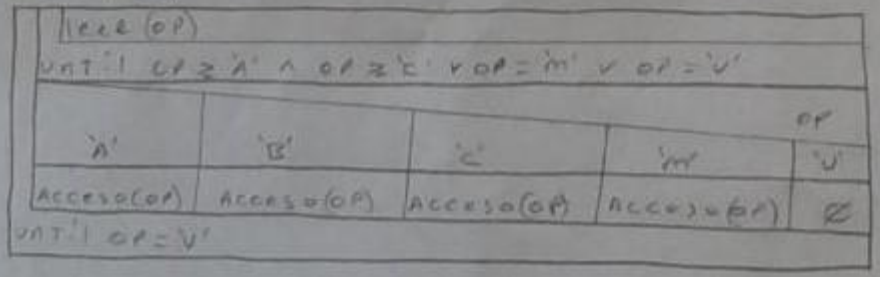
ADMINISTRACIONES()

VAR OP: CHAR;



ABCM(ACCESO)

VAR OP: CHAR;



PRODUCTOS (OP: CHAR)

VAR OPC: INTEGER;

'A'	'B'	'C'	'M'
FOR I ← 1 TO 3	MUESTRASTR (PROD, 3)	MUESTRASTR (PROD, 3)	MUESTRASTR (PROD, 3)
PROD[I] = ''	OPC ← VALID_INT (1, 3)		OPC ← VALID_INT (1, 3)
LEER PROD[I] / Ø	PROD[OPC] ← ''		LEER (PROD[OPC])

OP

VALID_INT (MIN, MAX: INTEGER): INTEGER

VAR NRO: INTEGER

LEER (NRO)
UNTIL NRO ≥ MIN ∧ NRO ≤ MAX
VALID_INT ← NRO

MUESTRASTR (X: ARREGLO_STR, N: INTEGER)

VAR I: INTEGER;

FOR I ← 1 TO N
  MOSTRAR (X[I])

CUPOS()

VAR PATENTE: STRING[7];
  I: INTEGER

I ← Ø
LEER (PATENTE)
WHILE PATENTE ≠ '*' ∧ I < 8

BUSQUEDA (CUPOS, PATENTE, 8) = -1

CUPOS[I] = ''

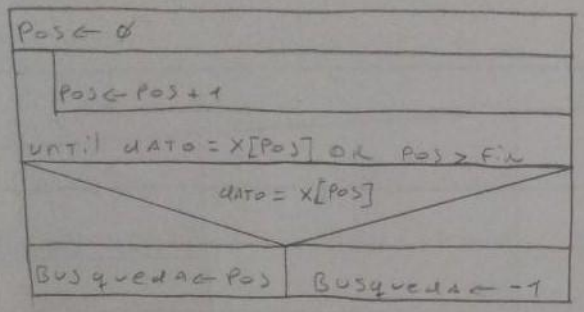
CUPOS[I] ← PATENTE
ESTADO[I] ← 'P'
C ← C + 1
I ← I + 1

MOSTRAR ('LA PATENTE YA FUE REGISTRADA')

UNTIL CUPOS[I] = PATENTE ∨ I = 8
LEER (PATENTE)

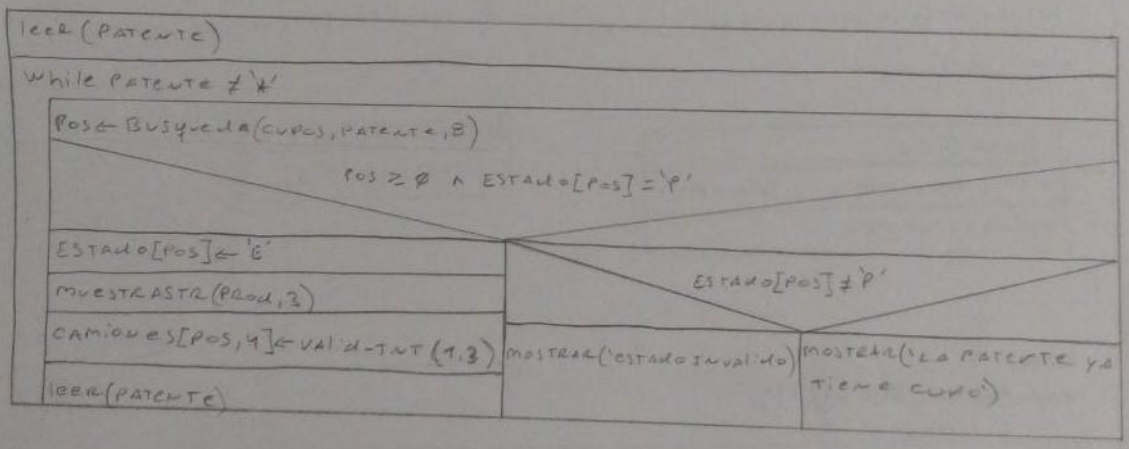
Busqueda(x:LISTA_STR, dato:STRING, fin:Integer):Integer

VAR pos:Integer;



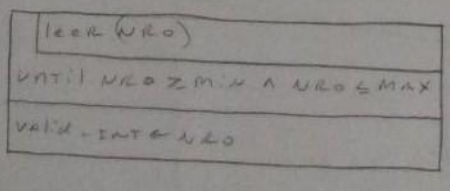
Recepcion()

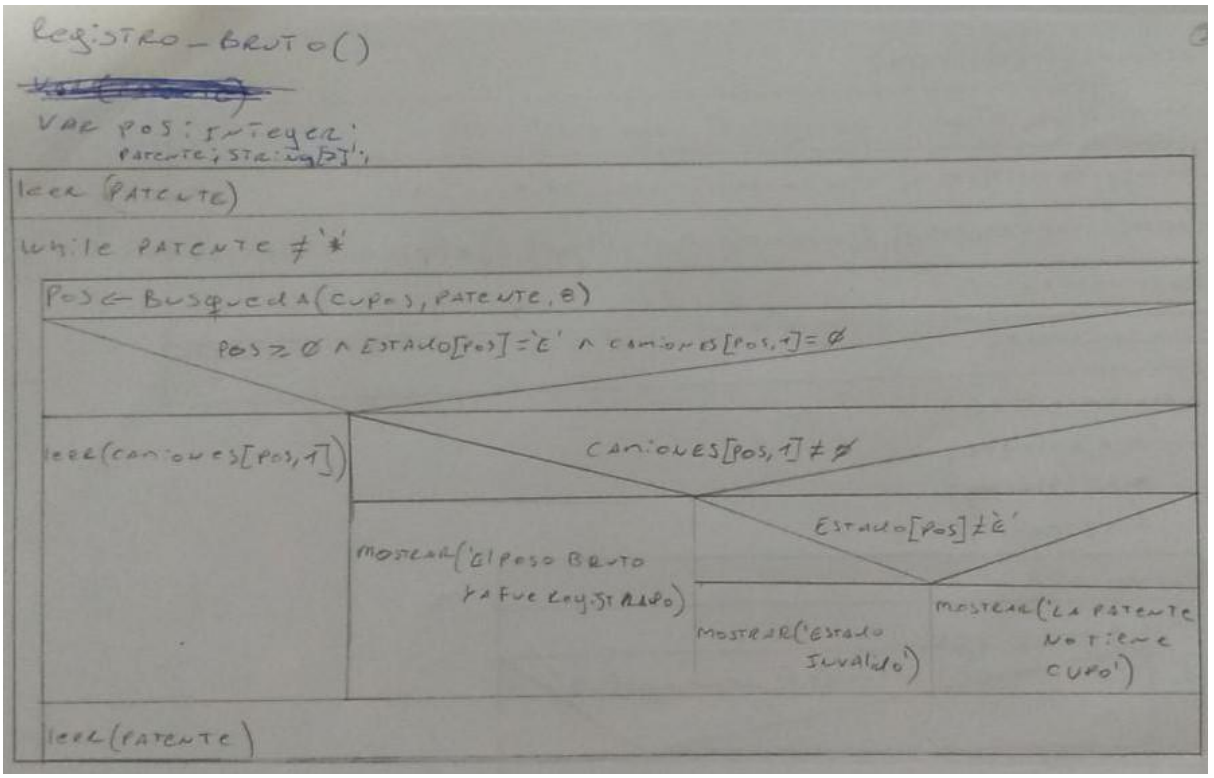
VAR pos:Integer;
PATENTE:STRING;

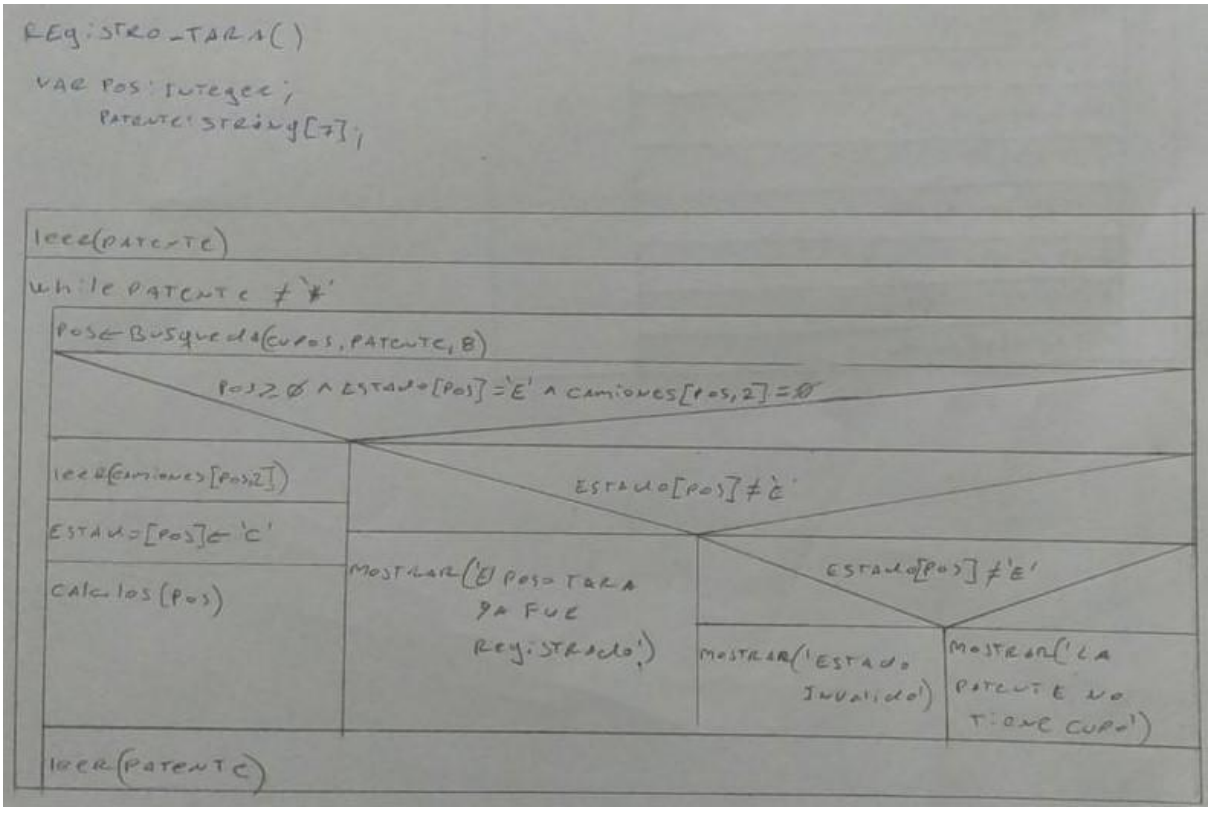


VALID-INT(min, max:Integer):Integer

VAR NRO:Integer




REGISTRO_BRUTO()
VAR POS: INTEGER;
PATENTE: STRING[7];
leer (PATENTE)
while PATENTE ≠ '*'
POS ← Busqueda(CUPOS, PATENTE, 8)
POS ≥ ∅ ∧ ESTADO[POS] = 'E' ∧ CAMIONES[POS,1] = ∅
leer(camiones[POS,1])
CAMIONES[POS,1] ≠ ∅
ESTADO[POS] ≠ 'E'
MOSTRAR('El Peso BRUTO ya fue Registrado')
MOSTRAR('Estado Invalido')
MOSTRAR('LA PATENTE NO TIENE CUPO')
leer(PATENTE)


REGISTRO_TARA()
VAR POS: Integer;
PATENTE: String[7];
leer(PATENTE)
while PATENTE ≠ '*'
POS ← Busqueda(CUPOS, PATENTE, 8)
POS ≥ ∅ ∧ ESTADO[POS] = 'E' ∧ CAMIONES[POS,2] = ∅
leer(camiones[POS,2])
ESTADO[POS] ← 'C'
Calculos(POS)
ESTADO[POS] ≠ 'E'
MOSTRAR('El Peso TARA ya fue Registrado')
MOSTRAR('Estado Invalido')
MOSTRAR('LA PATENTE NO TIENE CUPO')
leer(PATENTE)

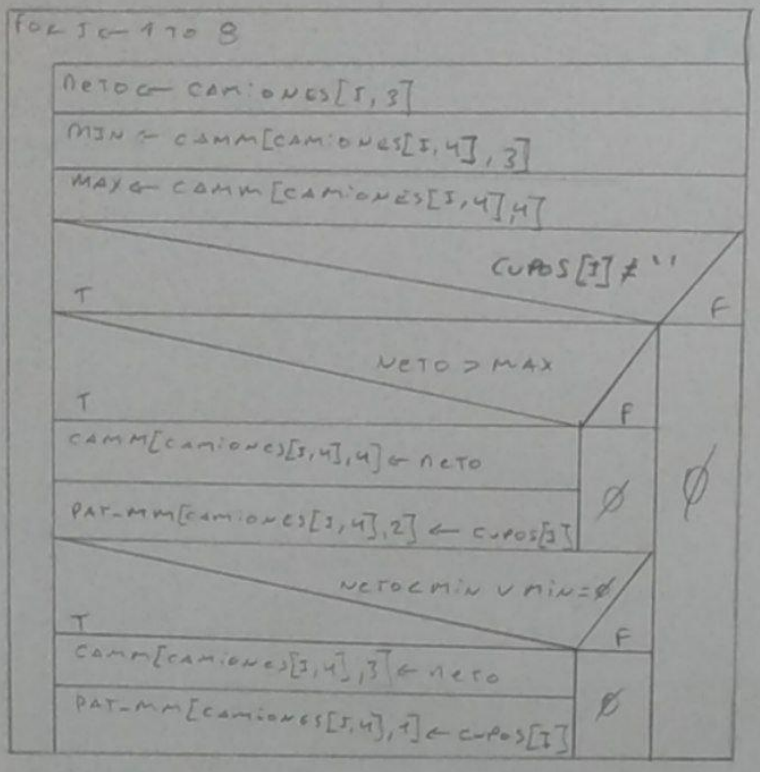
```
Calculos(Pos:Integer)
CAMIONES[Pos,3] ← CAMIONES[Pos,1] - CAMIONES[Pos,2]
CAMM[CAMIONES[Pos,4],1] ← CAMM[CAMIONES[Pos,4],1] + 1
CAMM[CAMIONES[Pos,4],2] ← CAMM[CAMIONES[Pos,4],2] + CAMIONES[Pos,3]
ORDENAMIENTO()
CALCULOMINMAX()
```

```
ORDENAMIENTO()
VAR AUX:INTEGER;
    AUX2:STRING;
    AUX3:CHAR;
FOR I ← 1 TO 7
  FOR J ← I+1 TO 8
    CAMIONES[I,3] < CAMIONES[J,3]
    FOR K ← 1 TO 4
      AUX ← CAMIONES[I,K]
      CAMIONES[I,K] ← CAMIONES[J,K]
      CAMIONES[J,K] ← AUX
    AUX2 ← CURSOS[I]
    CURSOS[I] ← CURSOS[J]
    CURSOS[J] ← AUX2
    AUX3 ← ESTADO[I]
    ESTADO[I] ← ESTADO[J]
    ESTADO[J] ← AUX3
```

∅

CalculoMinMax()

var neto, min, max: integer;



REPORTES()

MOSTRAR('CANTIDAD DE CUPOS OTORGADOS:', C)

MOSTRAR('CANTIDAD DE CAMIONES RECIBIDOS:', camm[1,1] + camm[2,1] + cam[3,1])

FOR I ← 1 TO 3

    MOSTRAR(PROV[I], camm[I,1], camm[I,2], camm[I,3]/cam[I,1], pat_mm[I,1], pat_mm[I,2])

FOR I ← 1 TO 8

    MOSTRAR(cupos[I], prod[camiones[I,4]], camiones[I,3])