UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE

Nombre: Criollo Solano Jahir **NRC:** 29583

Fecha: 03/01/2026

Tema: Ejercicios de Cap. 4

1. EJERCICIOS

Ejercicio 1 (Datos Población)

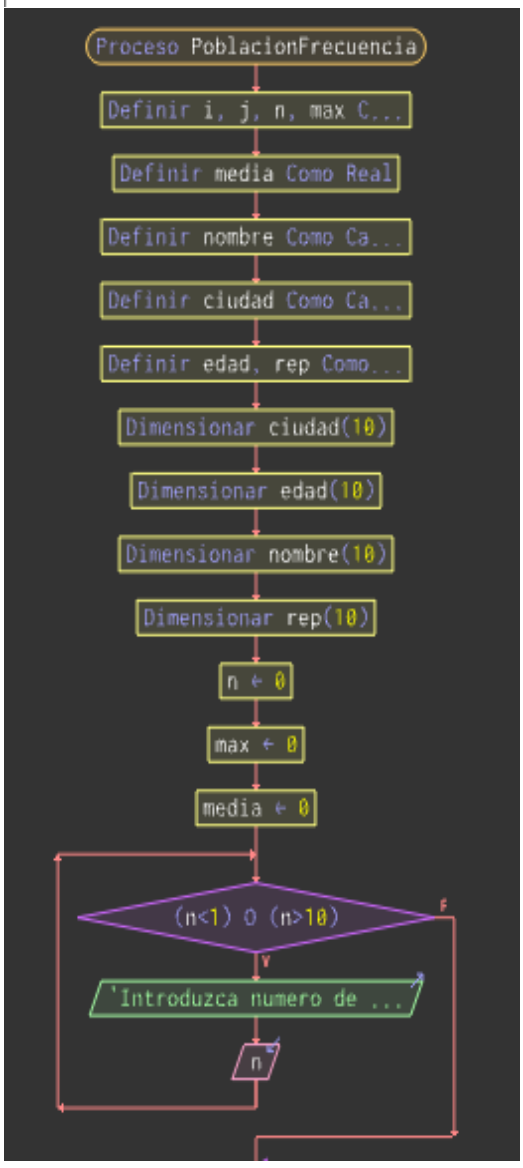
Objeto	Nombre	Valor	Tipo
Contador del ciclo	i	Variable	Entero
Contador del ciclo secundario	j	Variable	Entero
Número de personas (1–10)	n	Variable	Entero
Máximo número de repeticiones	máx	Variable	Entero
Edad media	media	Variable	Real
Edades de las personas	nombre	Arreglo	Cadena
Ciudad de nacimiento	edad	Arreglo	Entero
Condición de recursividad	ciudad	Arreglo	Cadena
Repeticiones del nombre	rep	Arreglo	Entero
Límite máximo de personas	10	Constante	Entero
Límite mínimo de personas	1	Constante	Entero

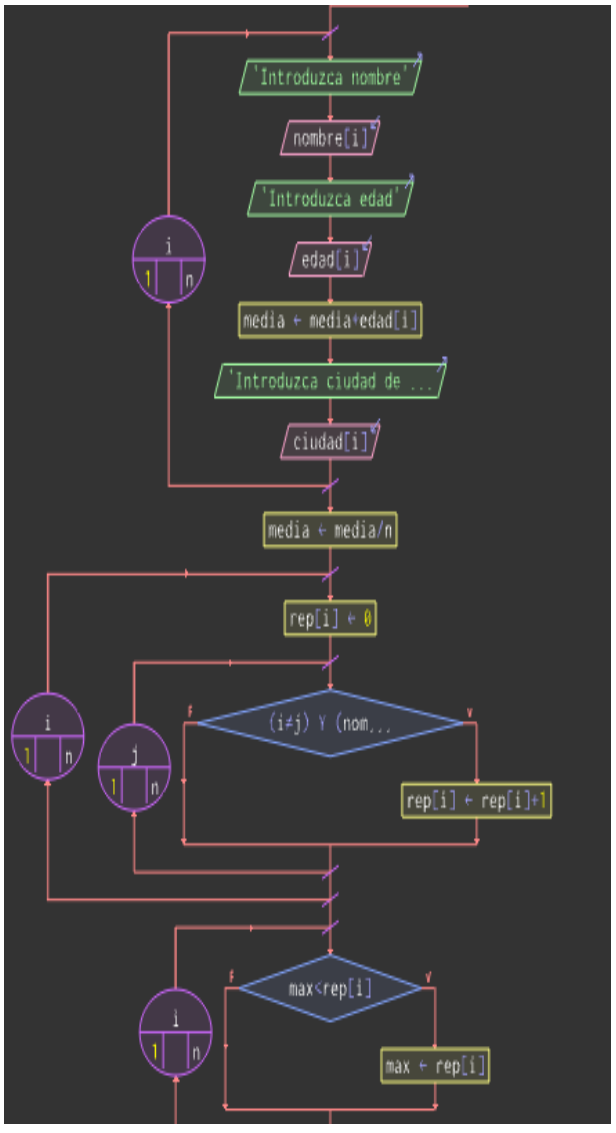
PSEUDOCÓDIGO

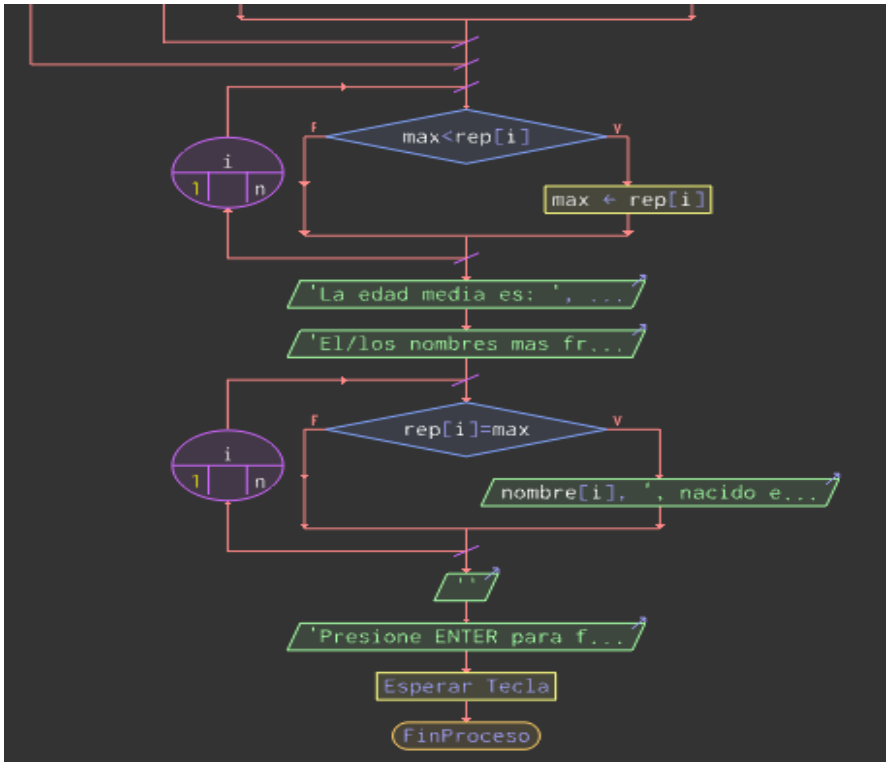
```
Proceso PoblacionFrecuencia
   Definir i, j, n, max Como Entero;
   Definir media Como Real;

   Definir nombre Como Cadena;
   Definir ciudad Como Cadena;
   Definir edad, rep Como Entero;

   Dimension nombre[10];
   Dimension ciudad[10];
   Dimension edad[10];
   Dimension rep[10];

   n <- 0;
   max <- 0;
```

```
    media <- 0;

    Mientras (n < 1) O (n > 10) Hacer;
       Escribir "Introduzca numero de personas (1-10)";
       Leer n;
    FinMientras;

    Para i <- 1 Hasta n Hacer;
       Escribir "Introduzca nombre";
       Leer nombre[i];

       Escribir "Introduzca edad";
       Leer edad[i];
       media <- media + edad[i];

       Escribir "Introduzca ciudad de nacimiento";
       Leer ciudad[i];
    FinPara;

    media <- media / n;

    Para i <- 1 Hasta n Hacer;
       rep[i] <- 0;
       Para j <- 1 Hasta n Hacer;
          Si (i <> j) Y (nombre[i] = nombre[j]) Entonces
             rep[i] <- rep[i] + 1;
          FinSi;
       FinPara;
    FinPara;

    Para i <- 1 Hasta n Hacer;
       Si max < rep[i] Entonces
          max <- rep[i];
       FinSi;
    FinPara;

    Escribir "La edad media es: ", media;
    Escribir "El/los nombres mas frecuentes son:";

    Para i <- 1 Hasta n Hacer;
       Si rep[i] = max Entonces
          Escribir nombre[i], ", nacido en ", ciudad[i];
       FinSi;
    FinPara;

    Escribir "";
    Escribir "Presione ENTER para finalizar...";
    Esperar Tecla;
FinProceso
```

```
> Brenda
Introduzca edad
> 21
Introduzca ciudad de nacimiento
> Quito
La edad media es: 19.3333333333
El/los nombres mas frecuentes son:
Jahir, nacido en Quito
Jhoel, nacido en Quito
Brenda, nacido en Quito

Presione ENTER para finalizar...
*** Ejecución Finalizada. ***
```







GDB Online: https://onlinegdb.com/egSp0NEd4_

Ejercicio 2 (Reconocimiento de carácteres)

Se pretende escribir un programa para reconocer caracteres a partir de un mapa de puntos. El mapa de puntos describe la forma de un carácter como una matriz de unos y ceros de 8x8 celdas (véase figura 4.1). Se dispone además de una tabla de estructuras de tipo **"struc letras",** que puede suponer convenientemente creada e inicializada, y que contiene la descripción de las 27 letras del alfabeto, tal como se describe en el ejemplo.

Struc letras
{
Char cod_ASCII; /* Letra a la que corresponde la matriz de puntos*/
Int mptosd[8][8]; /*Matriz de puntos del carácter*/
};
Struct letras tab_let[27];

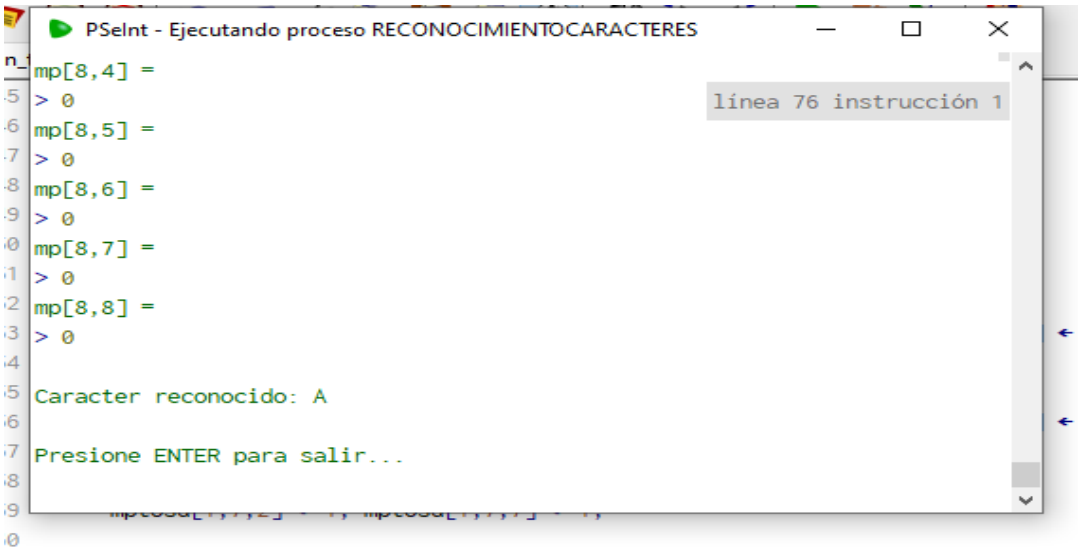
Objeto	Nombre	Valor	Tipo
Contador del ciclo principal	i	Variable	Entero
Contador del ciclo secundario	j	Variable	Entero
Número de personas (1–10)	n	Variable	Entero
Máximo número de repeticiones	max	Variable	Entero
Edad media	media	Variable	Real
Nombres de las personas	nombre	Arreglo	Cadena
Edades de las personas	edad	Arreglo	Entero

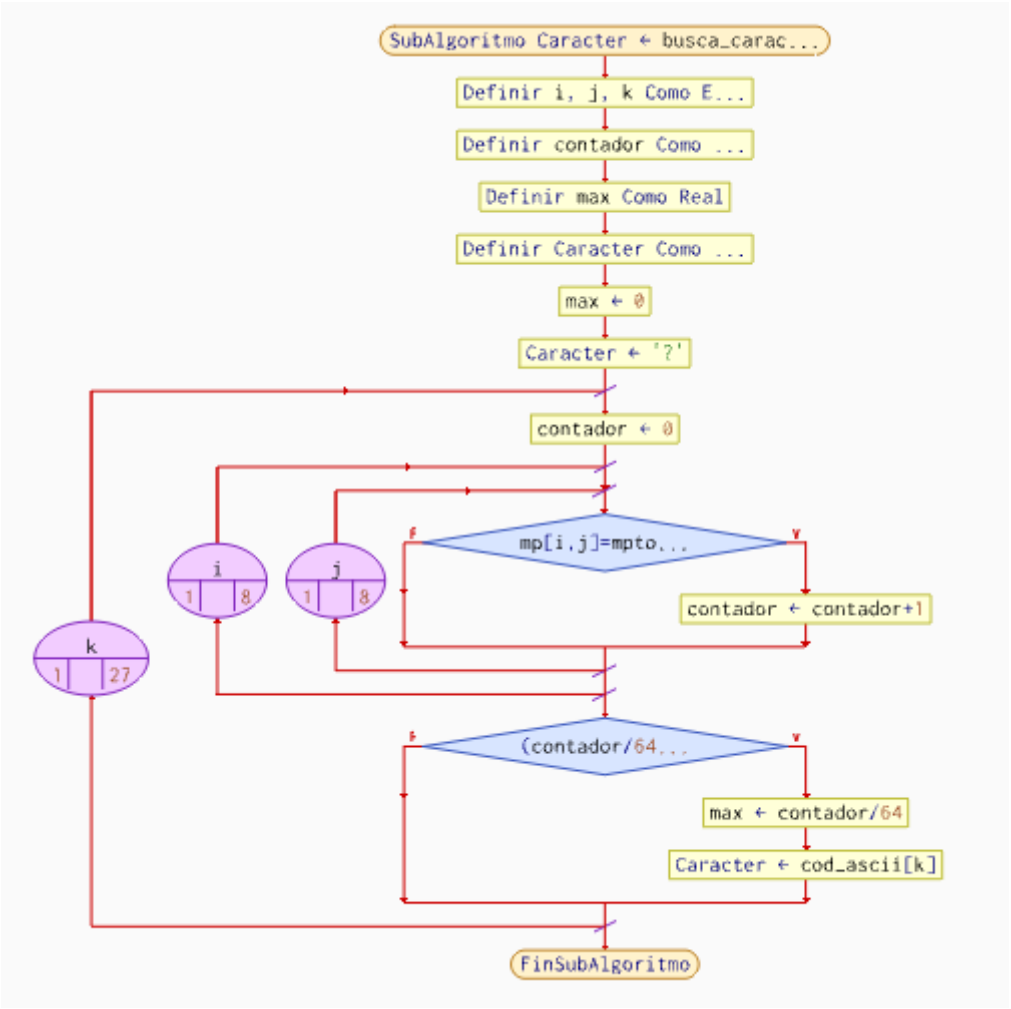
Ciudad de nacimiento	ciudad	Arreglo	Cadena
Repeticiones del nombre	rep	Arreglo	Entero
Límite máximo de personas	10	Constante	Entero
Límite mínimo de personas	1	Constante	Entero

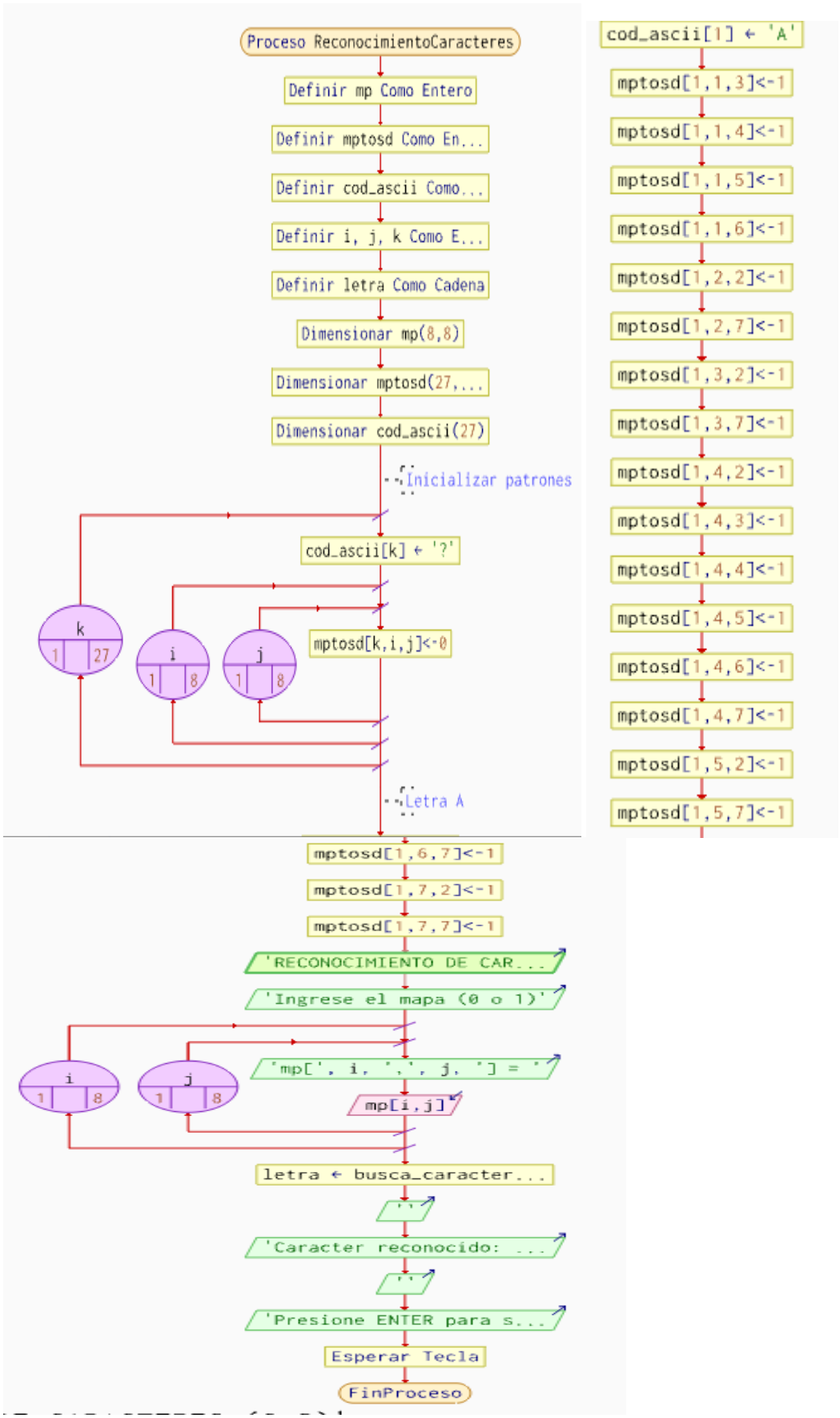
PSEUDOCÓDIGO

```
Funcion caracter <- busca_caracter(mp, mptosd, cod_ascii)
   Definir i, j, k Como Entero;
   Definir contador Como Entero;
   Definir max Como Real;
   Definir caracter Como Caracter;

   max <- 0;
   caracter <- "?";

   Para k <- 1 Hasta 27 Hacer
      contador <- 0;

      Para i <- 1 Hasta 8 Hacer
         Para j <- 1 Hasta 8 Hacer
            Si mp[i,j] = mptosd[k,i,j] Entonces
               contador <- contador + 1;
            FinSi;
         FinPara;
      FinPara;

      Si (contador / 64) > max Entonces
         max <- contador / 64;
         caracter <- cod_ascii[k];
      FinSi;
   FinPara;
FinFuncion


Proceso ReconocimientoCaracteres
   Definir mp Como Entero;
   Definir mptosd Como Entero;
   Definir cod_ascii Como Caracter;
   Definir i, j, k Como Entero;
   Definir letra Como Caracter;

   Dimension mp[8,8];
   Dimension mptosd[27,8,8];
   Dimension cod_ascii[27];

   // Inicializar patrones
   Para k <- 1 Hasta 27 Hacer
      cod_ascii[k] <- "?";
      Para i <- 1 Hasta 8 Hacer
         Para j <- 1 Hasta 8 Hacer
            mptosd[k,i,j] <- 0;
```

```
            FinPara;
        FinPara;
    FinPara;

    // Letra A
    cod_ascii[1] <- "A";

    mptosd[1,1,3] <- 1; mptosd[1,1,4] <- 1; mptosd[1,1,5] <- 1; mptosd[1,1,6] <- 1;
    mptosd[1,2,2] <- 1; mptosd[1,2,7] <- 1;
    mptosd[1,3,2] <- 1; mptosd[1,3,7] <- 1;
    mptosd[1,4,2] <- 1; mptosd[1,4,3] <- 1; mptosd[1,4,4] <- 1; mptosd[1,4,5] <- 1;
mptosd[1,4,6] <- 1; mptosd[1,4,7] <- 1;
    mptosd[1,5,2] <- 1; mptosd[1,5,7] <- 1;
    mptosd[1,6,2] <- 1; mptosd[1,6,7] <- 1;
    mptosd[1,7,2] <- 1; mptosd[1,7,7] <- 1;

    Escribir "RECONOCIMIENTO DE CARACTERES (8x8)";
    Escribir "Ingrese el mapa (0 o 1)";

    Para i <- 1 Hasta 8 Hacer
        Para j <- 1 Hasta 8 Hacer
            Escribir "mp[", i, ",", j, "] = ";
            Leer mp[i,j];
        FinPara;
    FinPara;

    letra <- busca_caracter(mp, mptosd, cod_ascii);

    Escribir "";
    Escribir "Caracter reconocido: ", letra;
    Escribir "";
    Escribir "Presione ENTER para salir...";
    Esperar Tecla;
FinProceso
```



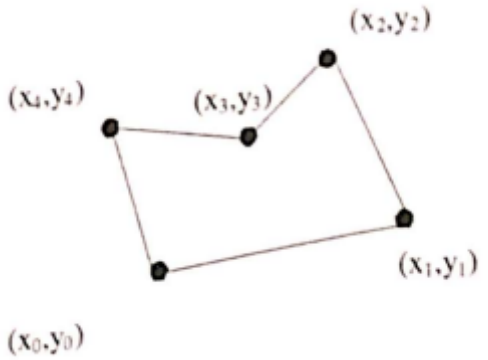

SubAlgoritmo Caracter ← busca_carac...
Definir i, j, k Como E...
Definir contador Como ...
Definir max Como Real
Definir Caracter Como ...
max ← 0
Caracter ← '?'
contador ← 0
mp[i,j]=mpto...
F
V
contador ← contador+1
i
1
8
j
1
8
k
1
27
(contador/64...
F
V
max ← contador/64
Caracter ← cod_ascii[k]
FinSubAlgoritmo



GDB Online: https://onlinegdb.com/lunz-SoFb

Ejercicio 3 (Polígono)

Un polígono es una figura geométrica cerrada delimitada por segmentos rectos (aristas). En este ejercicio se usa la estructura tipo para almacenar la información de un polígono, donde en vez de almacena sus aristas se almacenan sus vértices, como se describe a continuación:

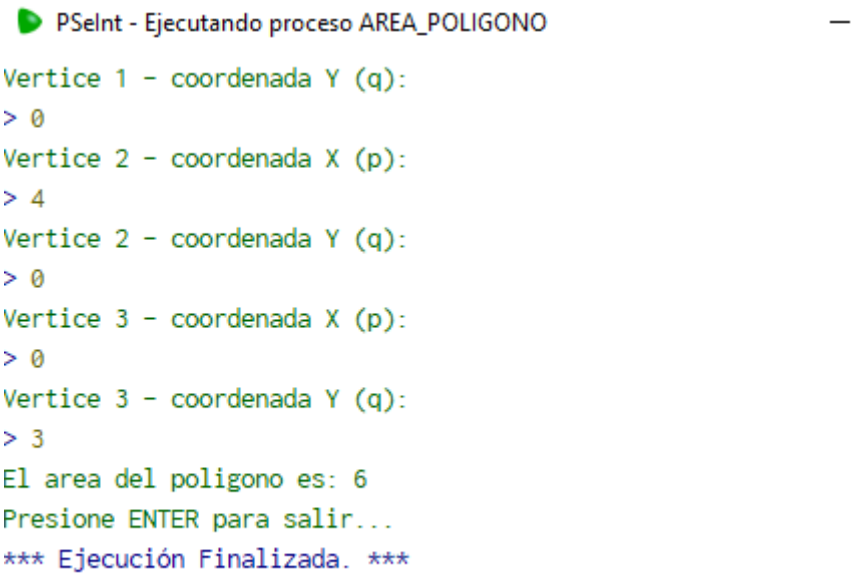


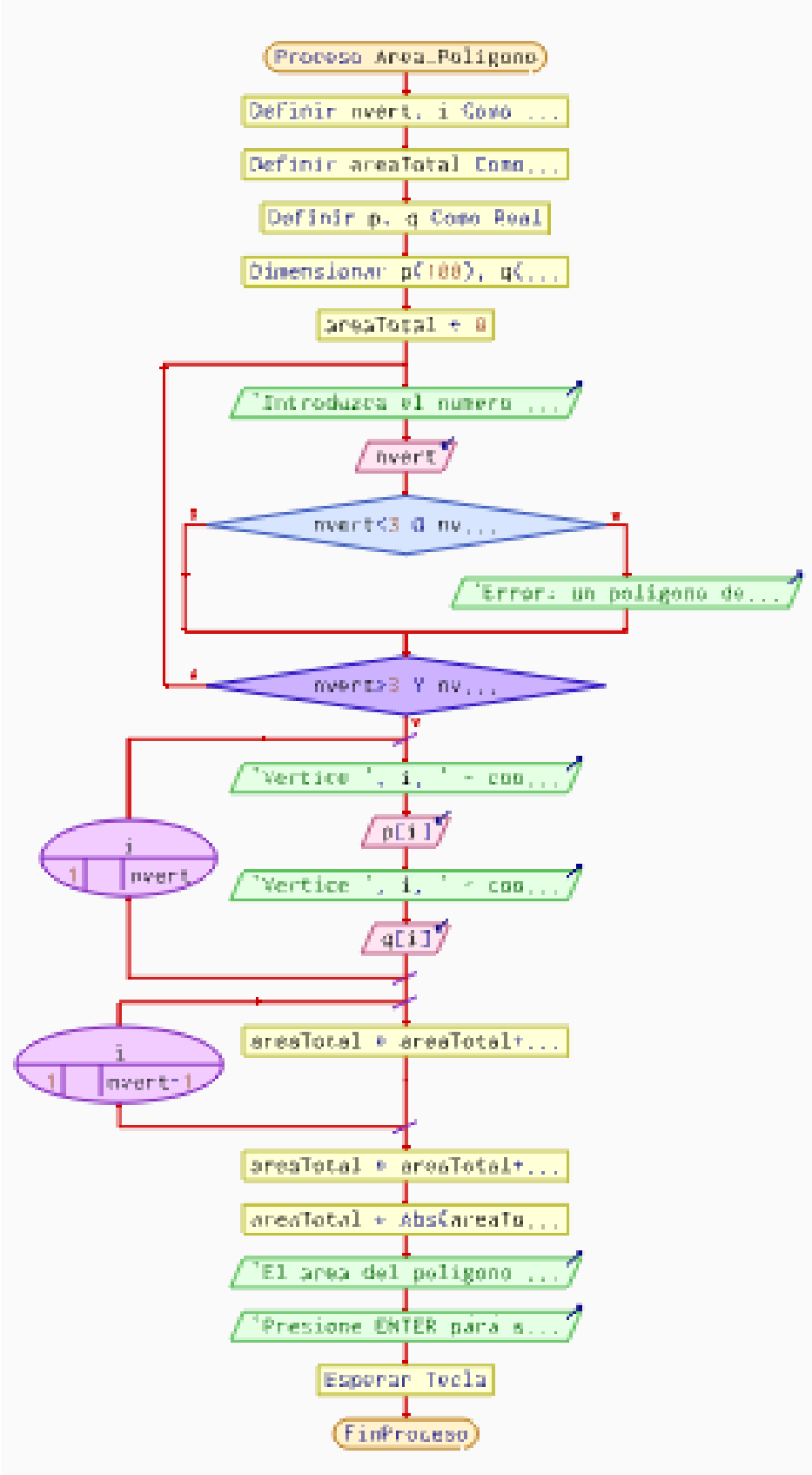
Objeto	Nombre	Valor	Tipo
Número de vértices	nvert	Variable	Entero
Contador de ciclos	i	Variable	Entero
acumulador del cálculo del área	areaTotal	Variable	Real
coordenadas X de los vértices	p	Arreglo	Real
coordenadas Y de los vértices	q	Arreglo	Real

PSEUDOCÓDIGO

```
Algoritmo Area_Poligono

   Definir nvert, i Como Entero;
   Definir areaTotal Como Real;
   Definir p, q Como Real;
   Dimension p[100], q[100];

   areaTotal <- 0;

   Repetir
      Escribir "Introduzca el numero de vertices (3-100): ";
      Leer nvert;
      Si nvert < 3 O nvert > 100 Entonces
         Escribir "Error: un poligono debe tener entre 3 y 100 vertices.";
      FinSi
   Hasta Que nvert >= 3 Y nvert <= 100

   Para i <- 1 Hasta nvert Hacer
      Escribir "Vertice ", i, " - coordenada X (p): ";
      Leer p[i];
      Escribir "Vertice ", i, " - coordenada Y (q): ";
      Leer q[i];
   FinPara
```

```
    Para i <- 1 Hasta nvert - 1 Hacer
        areaTotal <- areaTotal + (p[i] * q[i + 1] - p[i + 1] * q[i]);
    FinPara

    areaTotal <- areaTotal + (p[nvert] * q[1] - p[1] * q[nvert]);

    areaTotal <- Abs(areaTotal) / 2;

    Escribir "El area del poligono es: ", areaTotal;

    Escribir "Presione ENTER para salir...";
    Esperar Tecla;
FinAlgoritmo
INICIO
//DECLARACIÓN DE VARIBLES
   N←Int;
   D←Int;
   R←Int;
//DESARROLLO
  Escribe “Ingrese el primer número”;
  Lee N;
  Escribe “Ingrese el segundo número”;
  Lee D;
//CÁLCULOS
  R← N%D;
  Si R←0, Entonces;
  Imprime “El número”, N, “es divisible por”, D;
  Si No
  Imprime “El número”, N, “No es divisible por”, D;
  Fin Si
FIN
```





GDB Online: https://onlinegdb.com/2AUWSBcnl

Ejercicio 4 (Game Over)

Se pretende hacer un pequeño juego de palabras por ordenador en lenguaje C que muestra en pantalla los caracteres que apareen en la figura 4.3 de fin de juego.
Considere la siguiente estructura:

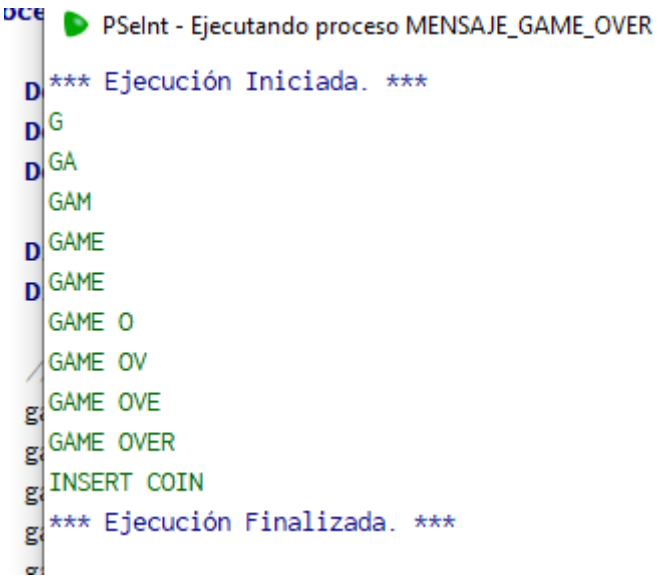
#define N 10
#define M 12
Struct mensaje
{
Char game_over[N];
Char se_acabo[M];
};
Typedef struct mensaje men;

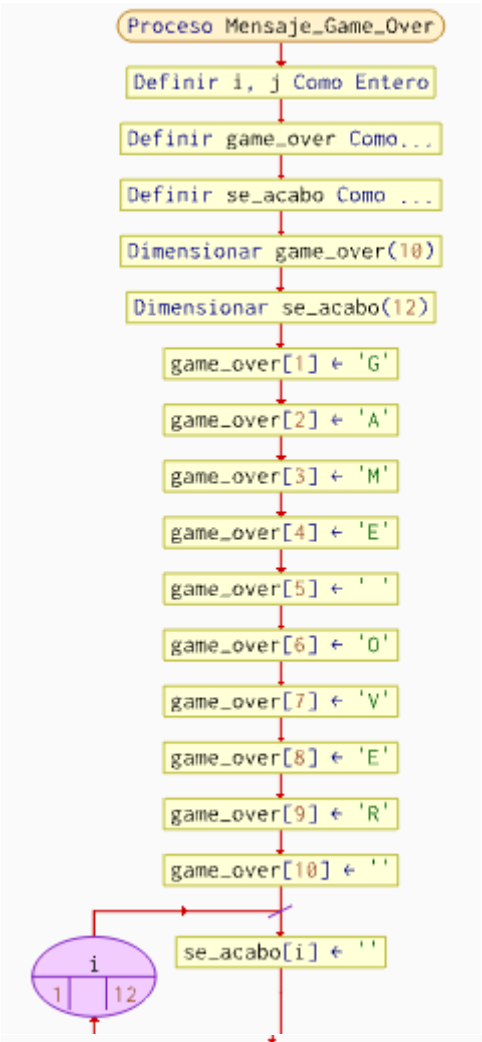
Objeto	Nombre	Valor	Tipo
Contador1	i	Variable	Entero
Contador2	j	Variable	Entero
Mensaje principal	Game_over	Variable	Carácter
Mensaje final	Se_acabo	Variable	Carácter
Tamaño mensaje G. O	10	Constante	Entero
Tamaño mensaje I. C	12	Constante	Entero
Letra individual G. O	Game_over[i]	Variable	Carácter
Letra individual I. C	Se_acabo[i]	Variable	Carácter
Texto “GAME OVER”	"GAME OVER"	Constante	Cadena
Texto “INSERT COIN”	"INSERT COIN"	Constante	Cadena

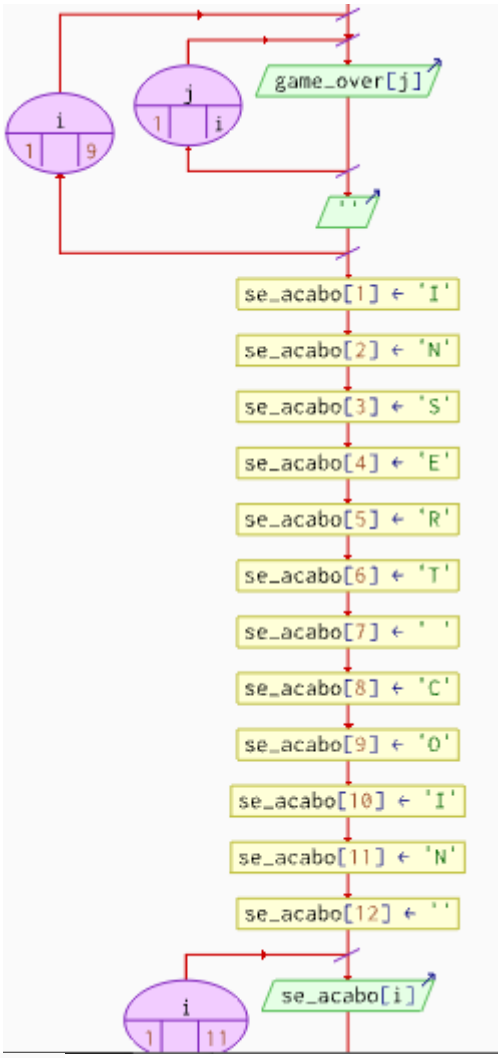
PSEUDOCÓDIGO

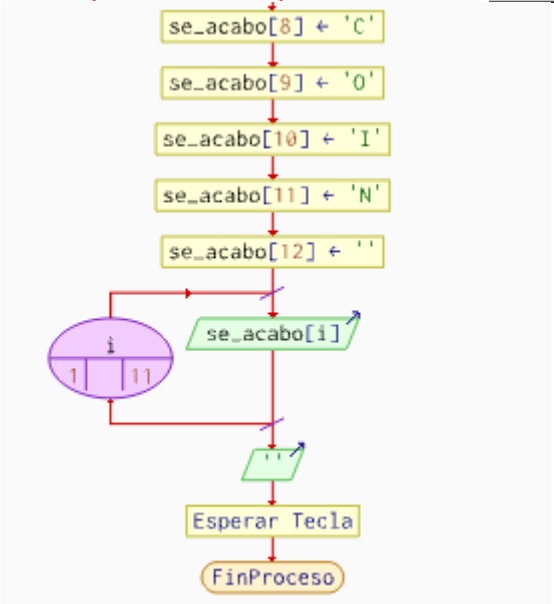
```
Proceso Mensaje_Game_Over

  Definir i, j Como Entero;
  Definir game_over Como Caracter;
  Definir se_acabo Como Caracter;

  Dimension game_over[10];
  Dimension se_acabo[12];

  game_over[1] <- "G";
  game_over[2] <- "A";
  game_over[3] <- "M";
  game_over[4] <- "E";
  game_over[5] <- " ";
  game_over[6] <- "O";
  game_over[7] <- "V";
  game_over[8] <- "E";
  game_over[9] <- "R";
  game_over[10] <- "";

  Para i <- 1 Hasta 12 Hacer
     se_acabo[i] <- "";
  FinPara

  Para i <- 1 Hasta 9 Hacer
```

```
    Para j <- 1 Hasta i Hacer
       Escribir Sin Saltar game_over[j];
    FinPara
    Escribir "";
  FinPara

  se_acabo[1] <- "I";
  se_acabo[2] <- "N";
  se_acabo[3] <- "S";
  se_acabo[4] <- "E";
  se_acabo[5] <- "R";
  se_acabo[6] <- "T";
  se_acabo[7] <- " ";
  se_acabo[8] <- "C";
  se_acabo[9] <- "O";
  se_acabo[10] <- "I";
  se_acabo[11] <- "N";
  se_acabo[12] <- "";

  Para i <- 1 Hasta 11 Hacer
     Escribir Sin Saltar se_acabo[i];
  FinPara
  Escribir "";

  Esperar Tecla;

FinProceso
```









GDB Online: https://onlinegdb.com/mjeBc0QDW

2. CONCLUSIONES

1. La correcta declaración de variables y arreglos en PSeInt es fundamental para evitar errores de ejecución, especialmente al trabajar con datos de tipo carácter.
2. La conversión de código desde lenguaje C a PSeInt permitió comprender mejor la lógica del programa y las diferencias básicas entre ambos lenguajes de programación.

3. RECOMENDACIONES

1. Definir siempre el tipo de todas las variables antes de utilizarlas en PSeInt, respetando el orden **Definir → Dimensión**, para asegurar la correcta ejecución del algoritmo.
2. Verificar el funcionamiento del programa con datos simples y realizar pruebas paso a paso antes de ejecutar casos más complejos.

4. REFERENCIAS

Joyanes Aguilar, L. (2013). *Fundamentos de programación: algoritmos, estructuras de datos y objetos.*