UNIVERSIDADE
DE VIGO

ESCOLA SUPERIOR DE ENXEÑERÍA INFORMÁTICA

PRÁCTICA 7: PLSQL

GRUPO: BDII5_3

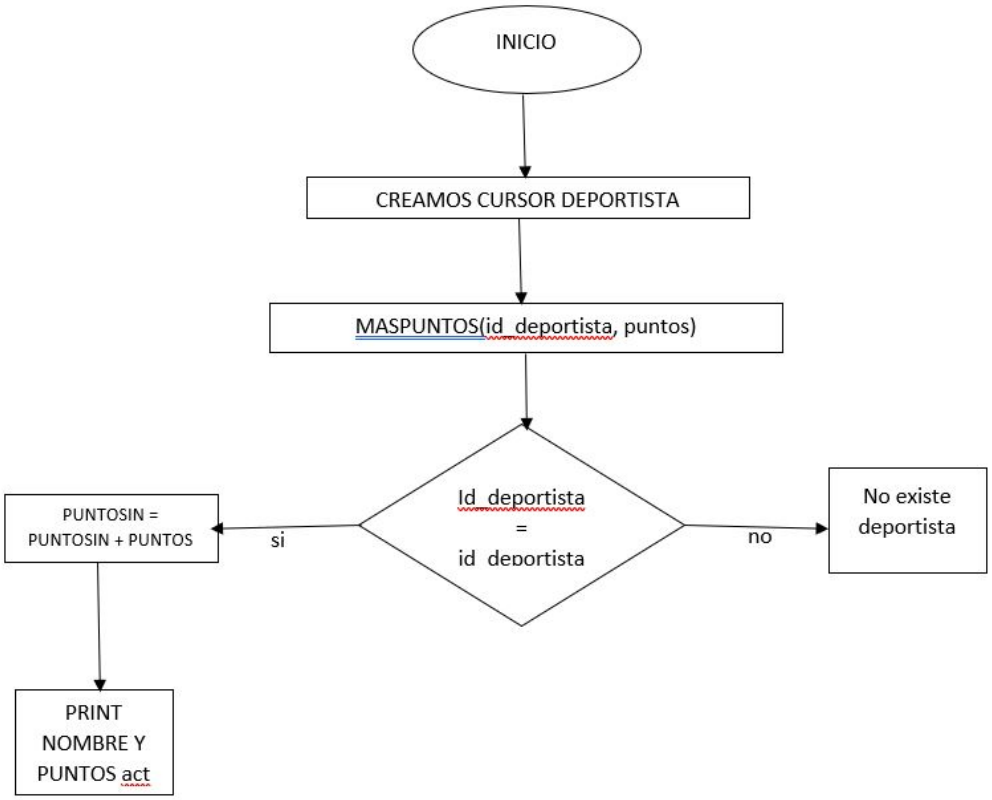
TEMÁTICA: ATP

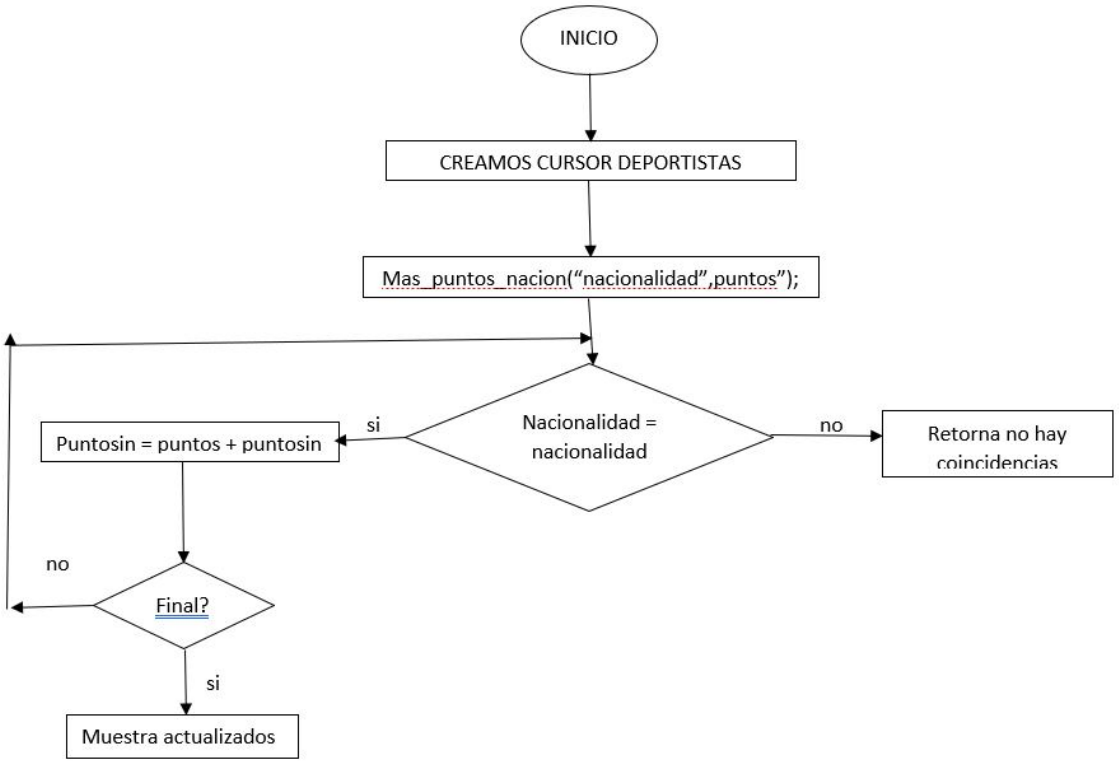
DNI: 77544994B NOMBRE: PONTE BAQUERO, OSCAR
DNI:44499556F NOMBRE: SABUCEDO GONZÁLEZ, ALBETO MATEO
DNI: 77009702K NOMBRE: TENORIO COSTA, JUAN
DNI: DNI NOMBRE: APELLIDOS, NOMBRE

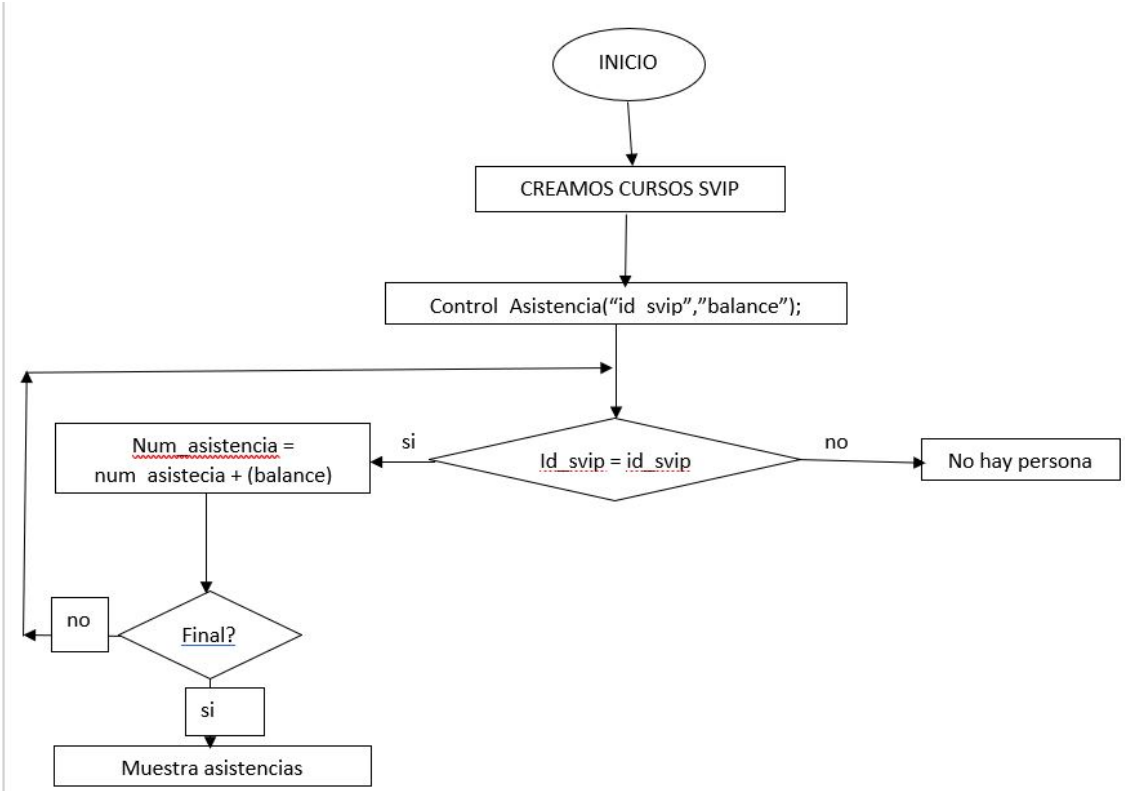
BASES DE DATOS II 2019-2020

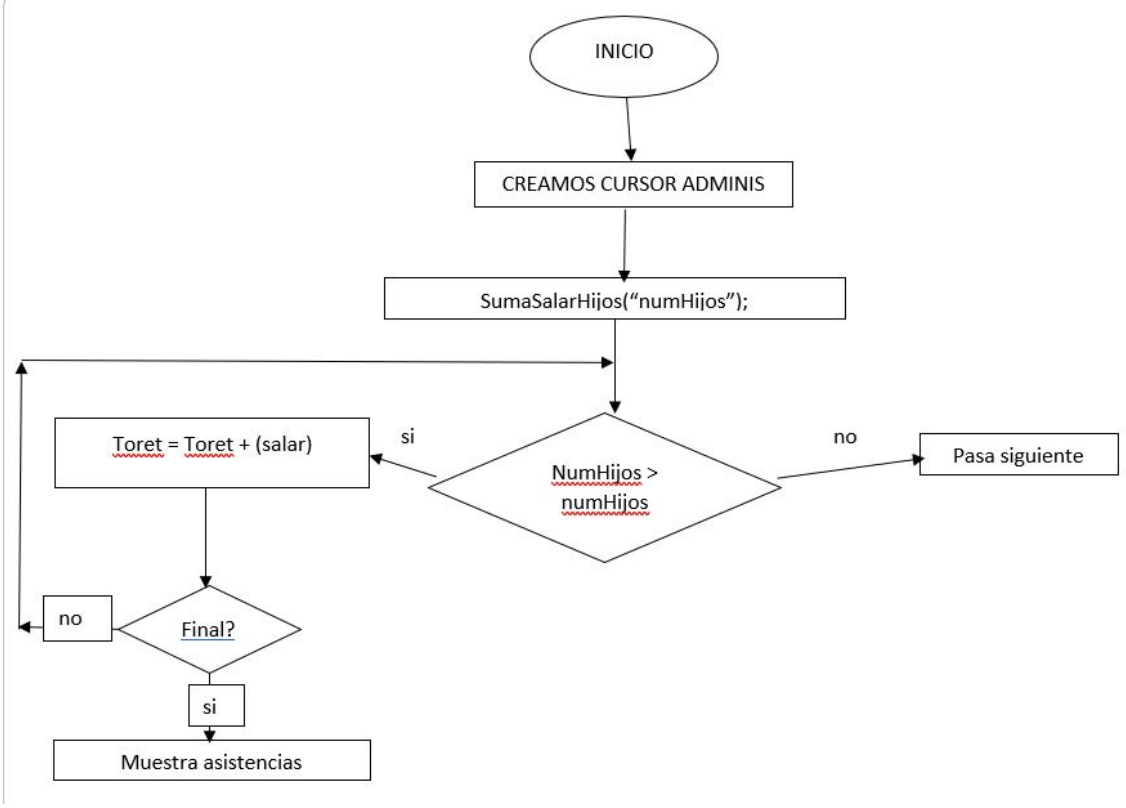
CALIFICACIÓN

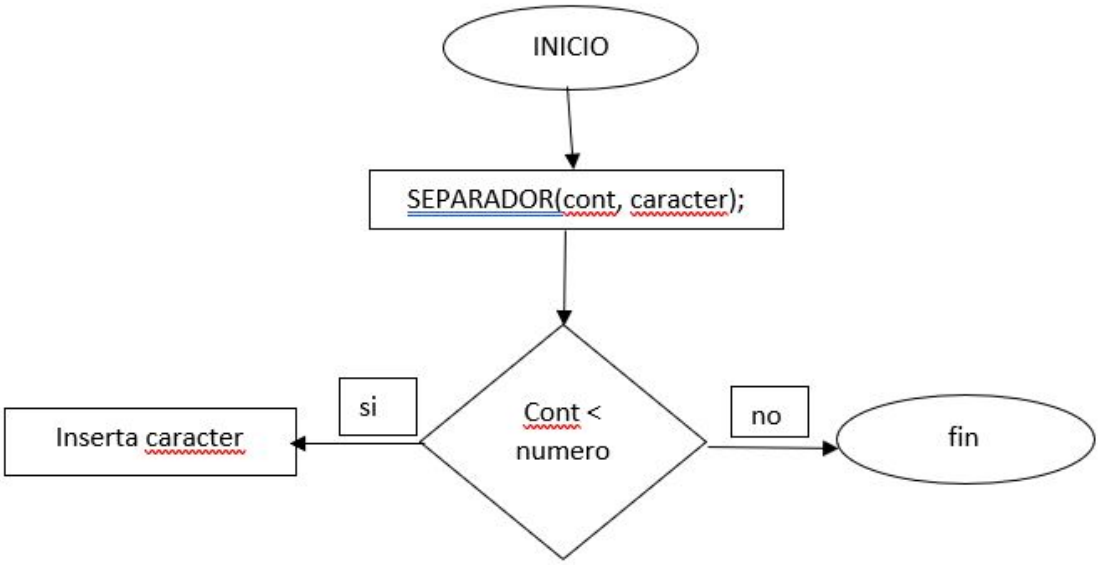
DIAGRAMAS DE FLUJO DE PROCEDIMIENTOS Y FUNCIONES

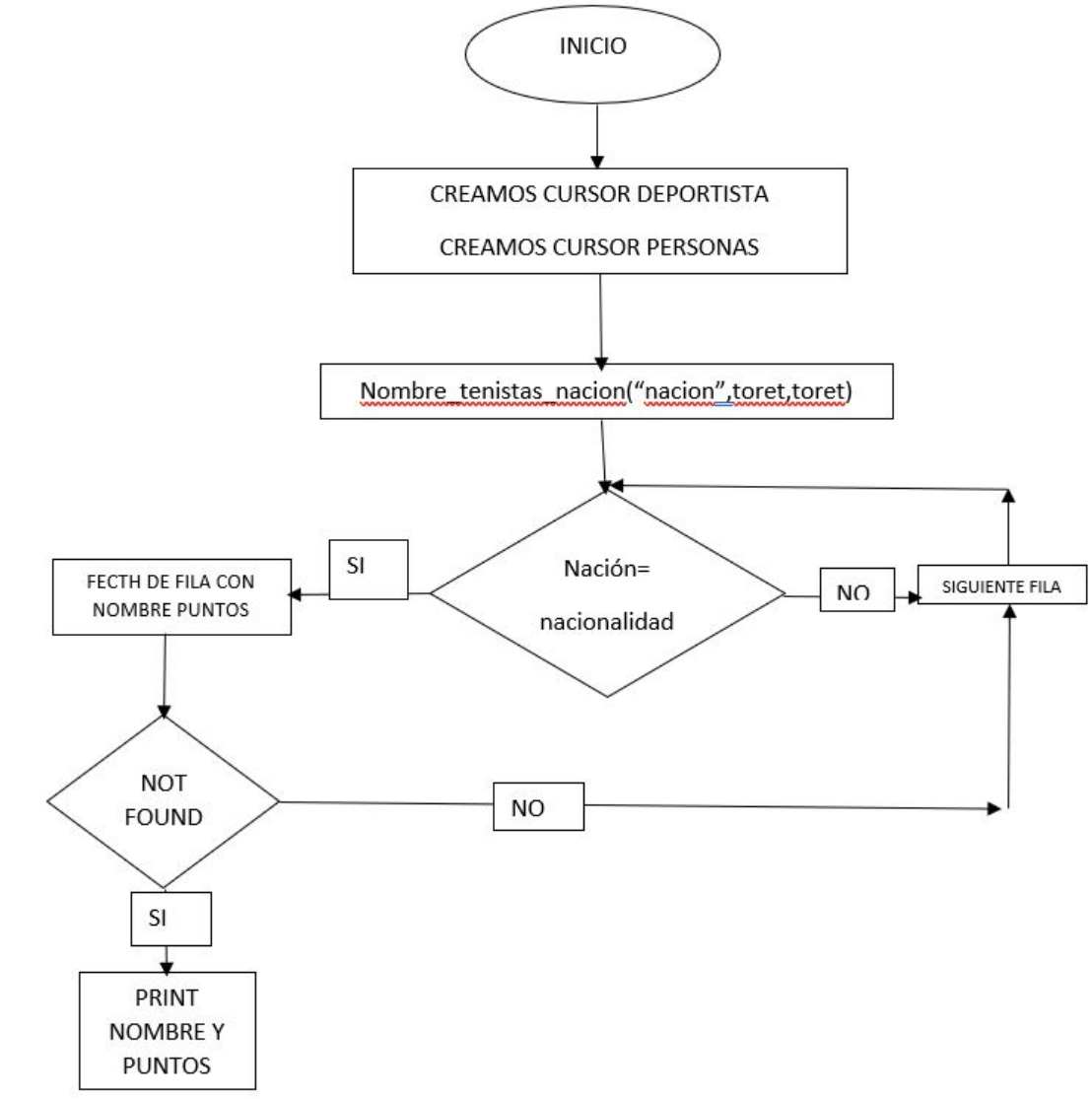





INICIO
CREAMOS CURSOS SVIP
Control Asistencia("id svip","balance");
Num_asistencia =
num_asistecia + (balance)
si
Id_svip = id_svip
no
No hay persona
no
Final?
si
Muestra asistencias


INICIO
CREAMOS CURSOR ADMINIS
SumaSalarHijos("numHijos");
Toret = Toret + (salar)
si
NumHijos >
numHijos
no
Pasa siguiente
no
Final?
si
Muestra asistencias


INICIO
SEPARADOR(cont, caracter);
Cont < numero
si
Inserta caracter
no
fin


INICIO
CREAMOS CURSOR DEPORTISTA
CREAMOS CURSOR PERSONAS
Nombre_tenistas_nacion("nacion",toret,toret)
Nación= nacionalidad
SI
FECTH DE FILA CON NOMBRE PUNTOS
NO
SIGUIENTE FILA
NOT FOUND
NO
SI
PRINT NOMBRE Y PUNTOS

SENTENCIAS DE DEFINICIÓN DE PROCEDIMIENTOS Y FUNCIONES

```
/*****************************************/
/* 7.- Procedimientos y Funciones PL/SQL */
/*****************************************/
--Procedimiento NOMBRE_DEPORTISTA_NACION
CREATE OR REPLACE
PROCEDURE NombreTenistasNacion(nacionin IN VARCHAR2,
numdeponacion OUT NUMBER)
IS

    regPer PERSONA%ROWTYPE;
    regDep DEPORTISTA%ROWTYPE;
    E_MI_EXCEPCION EXCEPTION;

    CURSOR C_DEPORTISTA IS
        SELECT ID_PERSONA, NOMBRE, APELLIDOS, SEXO, FECHA_NAC
        FROM DEPORTISTA D, PERSONA P
        WHERE NACIONALIDAD=nacionin AND ID_DEPORTISTA=ID_PERSONA
        ORDER BY PUNTOSIN DESC;

    CURSOR D_DEPORTISTA IS
        SELECT REVES, MANOD, NACIONALIDAD, PUNTOSIN,
ID_DEPORTISTA
        FROM DEPORTISTA D, PERSONA P
        WHERE NACIONALIDAD=nacionin AND
D.ID_DEPORTISTA=ID_PERSONA
        ORDER BY PUNTOSIN DESC;

        BEGIN
            OPEN C_DEPORTISTA;
            OPEN D_DEPORTISTA;
            DBMS_OUTPUT.PUT_LINE('Deportistas de nacionalidad:
"' || nacionin || '"');
            LOOP
                FETCH C_DEPORTISTA INTO regPer;
                EXIT WHEN C_DEPORTISTA%NOTFOUND;
                FETCH D_DEPORTISTA INTO regDep;
                DBMS_OUTPUT.PUT(regPer.NOMBRE || ' ' ||
regDep.puntosin);
                DBMS_OUTPUT.PUT_LINE('');
            END LOOP;

            numdeponacion := C_DEPORTISTA%ROWCOUNT;
            IF(numDeponacion = 0) THEN
                RAISE E_MI_EXCEPCION;
            END IF;

            CLOSE C_DEPORTISTA;
            CLOSE D_DEPORTISTA;
```

```
            EXCEPTION
                WHEN E_MI_EXCEPCION  THEN
                    DBMS_OUTPUT.PUT_LINE('No hay deportistas
para la nacionalidad introducida');
                WHEN OTHERS THEN
                    DBMS_OUTPUT.PUT_LINE('Código: ' || SQLCODE
|| SUBSTR(SQLERRM, 11, 100));
            END NombreTenistasNacion;
            /
            show errors;

--Procedimiento DEPORTISTAS_NACION
CREATE OR REPLACE
PROCEDURE masPuntos(idDep IN VARCHAR2, puntos IN NUMBER)
IS
    aux NUMBER;
    regDep DEPORTISTA%ROWTYPE;
    E_MI_EXCEPCION EXCEPTION;

    CURSOR C_DEPORTISTA IS
        SELECT REVES, MANOD, NACIONALIDAD, PUNTOSIN,
ID_DEPORTISTA
        FROM DEPORTISTA
        WHERE id_deportista=idDep
         FOR UPDATE;

        BEGIN
            DBMS_OUTPUT.PUT_LINE('Deportista: ' || idDep || '
recibe '|| puntos||' puntos');
            FOR regDep IN C_DEPORTISTA LOOP
            aux := aux + 1;
            UPDATE DEPORTISTA SET puntosIn = puntosIn + puntos
            WHERE CURRENT OF C_DEPORTISTA;
            END LOOP;
            IF(aux = 0) THEN
                RAISE E_MI_EXCEPCION;
            END IF;

            EXCEPTION
                WHEN E_MI_EXCEPCION  THEN
                    DBMS_OUTPUT.PUT_LINE('La id introducida no
existe');
                WHEN OTHERS THEN
                    DBMS_OUTPUT.PUT_LINE('Código: ' || SQLCODE
|| SUBSTR(SQLERRM, 11, 100));
            END masPuntos;
            /
            show errors;
--Procedimiento puntosNacion
CREATE OR REPLACE
PROCEDURE puntosNacion(nacionin IN VARCHAR2, puntos IN NUMBER)
IS

    aux NUMBER := 0;
    regDep DEPORTISTA%ROWTYPE;
    E_MI_EXCEPCION EXCEPTION;
```

```
    CURSOR C_DEPORTISTA IS
        SELECT REVES, MANOD, NACIONALIDAD, PUNTOSIN,
ID_DEPORTISTA
        FROM DEPORTISTA
        WHERE NACIONALIDAD=nacionin
        FOR UPDATE;

        BEGIN
            DBMS_OUTPUT.PUT_LINE('Deportistas de nacionalidad:
"' || nacionin || '" reciben ' || puntos || ' puntos');
            FOR regDep IN C_DEPORTISTA LOOP
            aux := aux + 1;
            UPDATE DEPORTISTA SET puntosIn = puntosIn + puntos
            WHERE CURRENT OF C_DEPORTISTA;
            END LOOP;
            IF(aux = 0) THEN
                RAISE E_MI_EXCEPCION;
            END IF;

            EXCEPTION
                WHEN E_MI_EXCEPCION  THEN
                    DBMS_OUTPUT.PUT_LINE('No hay deportistas
para la nacionalidad introducida');
                WHEN OTHERS THEN
                    DBMS_OUTPUT.PUT_LINE('Código: ' || SQLCODE
|| SUBSTR(SQLERRM, 11, 100));
            END puntosNacion;
            /
            show errors;

--Procedimiento controlAsistencia
CREATE OR REPLACE
PROCEDURE controlAsistencia(codigo IN VARCHAR2, balance IN
NUMBER)
IS

   aux NUMBER;
    regSvip SVIP%ROWTYPE;
    E_MI_EXCEPCION EXCEPTION;

    CURSOR C_SVIP IS
        SELECT num_asistencia, codigo_svip
        FROM SVIP
        WHERE codigo_svip=codigo
         FOR UPDATE;

        BEGIN
            DBMS_OUTPUT.PUT_LINE('Empleado: ' || codigo || '
recibe '|| balance ||' numeros de asistencia');
            FOR regSvip IN C_SVIP LOOP
            DBMS_OUTPUT.PUT_LINE(regSvip.codigo_svip||' contaba
con '||regSvip.num_asistencia);
            aux := aux + 1;
            UPDATE SVIP SET num_asistencia = num_asistencia +
(balance)
```

```
            WHERE CURRENT OF C_SVIP;
            END LOOP;

            OPEN C_SVIP;
            FETCH C_SVIP INTO regSvip;
            DBMS_OUTPUT.PUT_LINE(regSvip.codigo_svip||' ahora
tiene '||regSvip.num_asistencia);
            CLOSE C_SVIP;

            IF(aux = 0) THEN
                RAISE E_MI_EXCEPCION;
            END IF;

            EXCEPTION
                WHEN E_MI_EXCEPCION  THEN
                    DBMS_OUTPUT.PUT_LINE('La id introducida no
existe');
                WHEN OTHERS THEN
                    DBMS_OUTPUT.PUT_LINE('Código: ' || SQLCODE
|| SUBSTR(SQLERRM, 11, 100));
            END controlAsistencia;
            /
            show errors;

-- FUNCION:SumaSalariosHijos
CREATE OR REPLACE
FUNCTION SumaSalariosHijos(hijos IN NUMBER)
RETURN NUMBER
IS
   sumaSalarios NUMBER;
BEGIN
   SELECT SUM(SALAR) INTO sumaSalarios
      FROM ADMINISTRACION
      WHERE num_Hijos>=hijos;
   RETURN sumaSalarios;
END SumaSalariosHijos;
/
show errors

CREATE OR REPLACE
PROCEDURE separador(contador IN NUMBER,caracter IN CHAR)
IS
    aux NUMBER;
        BEGIN
            aux := contador;
            DBMS_OUTPUT.NEW_LINE;
            WHILE aux > 0 LOOP
                DBMS_OUTPUT.PUT(caracter);
                aux := aux- 1;
            END LOOP;
            DBMS_OUTPUT.NEW_LINE;
            DBMS_OUTPUT.NEW_LINE;
            DBMS_OUTPUT.NEW_LINE;
        END separador;
            /
            show errors;
```

SENTENCIAS DE DEFINICIÓN DE BLOQUES DE PRUEBAS

```
/*******************************************************/
/* 9.- Bloque para prueba de Procedimientos y Funciones */
/*******************************************************/
SET SERVEROUTPUT ON
DECLARE
    toret NUMBER;
    toret2 varchar2(10);
    contador NUMBER:=100;

BEGIN
    DBMS_OUTPUT.NEW_LINE;

    BEGIN
--Procedimiento
    separador(contador,'-');
    DBMS_OUTPUT.PUT_LINE('INICIO DE PROCEDIMIENTO:
NombreTenistasNacion');
    NombreTenistasNacion('ESPAÑOLA', toret);
    DBMS_OUTPUT.PUT_LINE('Tenemos a ' || toret || '
deportistas');
    DBMS_OUTPUT.NEW_LINE;
    EXCEPTION
        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE('[EXCEPCIÓN]');
    END;
    BEGIN
--Procedimiento
    separador(contador,'@');
    DBMS_OUTPUT.PUT_LINE('INICIO DE PROCEDIMIENTO: masPuntos');
    toret := 150;
    toret2 := '32456689K';
    masPuntos(toret2, toret);
    DBMS_OUTPUT.PUT_LINE('Se han otorgado ' || toret || ' puntos
a ' || toret2);
    DBMS_OUTPUT.NEW_LINE;
    EXCEPTION
        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE('[EXCEPCIÓN]');
    END;
    BEGIN
--Procedimiento
    separador(contador,'-');
    DBMS_OUTPUT.PUT_LINE('INICIO DE PROCEDIMIENTO:
NombreTenistasNacion');
    NombreTenistasNacion('ESPAÑOLA', toret);
    DBMS_OUTPUT.PUT_LINE('Tenemos a ' || toret || '
deportistas');
    DBMS_OUTPUT.NEW_LINE;
    EXCEPTION
        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE('[EXCEPCIÓN]');
    END;
```

```
    BEGIN
--Procedimiento
    separador(contador,'-');
    DBMS_OUTPUT.PUT_LINE('INICIO DE PROCEDIMIENTO:
puntosNacion');
    toret := 50;
    toret2 := 'ESPAÑOLA';
    puntosNacion(toret2, toret);
    DBMS_OUTPUT.PUT_LINE('Se han otorgado ' || toret || ' puntos
a los jugadores de la selección ' || toret2);
    DBMS_OUTPUT.NEW_LINE;
    EXCEPTION
        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE('[EXCEPCIÓN]');
    END;
    BEGIN
--Procedimiento
    separador(contador,'-');
    DBMS_OUTPUT.PUT_LINE('INICIO DE PROCEDIMIENTO:
NombreTenistasNacion');
    NombreTenistasNacion('ESPAÑOLA', toret);
    DBMS_OUTPUT.PUT_LINE('Tenemos a ' || toret || '
deportistas');
    DBMS_OUTPUT.NEW_LINE;
    EXCEPTION
        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE('[EXCEPCIÓN]');
    END;
    BEGIN
--Procedimiento
    separador(contador,'-');
    DBMS_OUTPUT.PUT_LINE('INICIO DE PROCEDIMIENTO: control
asistencia');
    toret := -1;
    toret2 := '0238978P';
    controlAsistencia(toret2, toret);
    DBMS_OUTPUT.PUT_LINE('Se ha modificado en ' || toret || ' al
numero de asistencia del empleado con codigo ' || toret2);
    DBMS_OUTPUT.NEW_LINE;
    EXCEPTION
        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE('[EXCEPCIÓN]');
    END;
       BEGIN
-- Funcion
      separador(contador,'-');
      DBMS_OUTPUT.PUT_LINE('INICIO FUNCIÓN: SumaSalariosHijos');
      toRet := SumaSalariosHijos(2);
      DBMS_OUTPUT.PUT_LINE('Suma de Salarios : ' || toRet || '
para personas con 2 o más hijos');
      DBMS_OUTPUT.PUT_LINE('FIN FUNCIÓN');
      DBMS_OUTPUT.NEW_LINE;
   EXCEPTION
      WHEN OTHERS THEN
         DBMS_OUTPUT.PUT_LINE('[EXCEPCIÓN]');
         DBMS_OUTPUT.PUT_LINE('[Código]: ' || SQLCODE);
```

```
         DBMS_OUTPUT.PUT_LINE('[Mensaje]: ' || SUBSTR(SQLERRM,
11, 100));
   END;
END;
```

HOJA DE FIRMAS

DNI: 77544994B NOMBRE: PONTE BAQUERO, OSCAR

DNI: 44499556F NOMBRE: SABUCEDO GONZÁLEZ, ALBETO MATEO

DNI: 77009702K NOMBRE: TENORIO COSTA, JUAN

DNI: DNI NOMBRE: APELLIDOS, NOMBRE