	Carátula para entrega de prácticas	
Facultad de Ingeniería	Laboratorio de docencia	

Laboratorios de computación salas A y B

Profesor: Karina García Morales

Asignatura: Fundamentos de programación

Grupo:

No. de Práctica: 5

Integrantes: Ever Ivan Rosales Gómez

No de Lista o brigade: 35

Semestre: 2023-2

Fecha de entrega: 29 de Marzo de 2023

Observaciones:

Calificación

PSEUDOCÓDIGO

OBJETIVO:

Elaborar pseudocódigos para la resolución de problemas con ayuda de sintaxis adecuada

DESARROLLO:

Para entender mejor esta práctica es importante saber qué es un pseudocódigo. Un pseudocódigo es una serie de pasos escritos para la resolución de problemas. cabe mencionar que debe cumplir un orden además de algunas reglas que se mencionan a continuación:

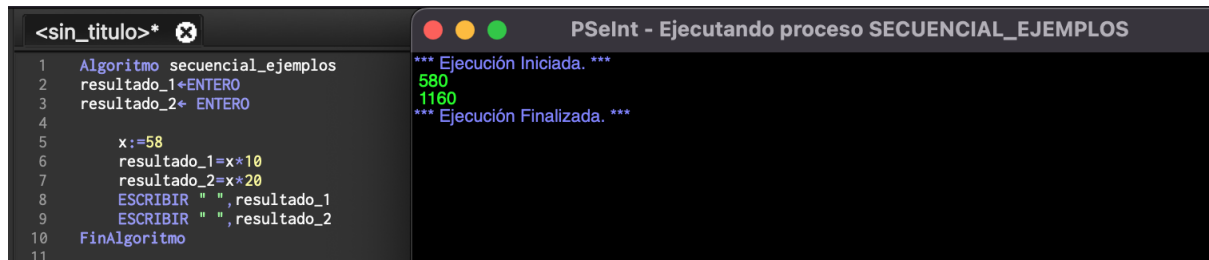
INICIO-FIN	Es importante que al iniciar un pseudocódigo empiece con un INICIO y al finalizarlo con un FIN.
PALABRAS RESERVADAS	Son instrucciones que le damos al código para hacer algo, por ejemplo al hacer un ciclo cuando se comienza con PARA.
SANGRÍA Y TABULACIÓN	Esto sirve para identificar bloques del pseudocódigo.
LECTURA Y ESCRITURA	Al querer que un escrito aparezca en pantalla no ayudamos de ESCRIBIR y el texto entre comillas, cuando queremos que algo se guarde en una variable hacemos uso de LEER
DECLARACIÓN DE VARIABLES	Sirve para conocer qué tipo de dato va ser el dato (entero, real, carácter, booleano, cadena)
OPERADORES ARITMÉTICOS	<p>Son operados, símbolos que nos ayudan a realizar las operaciones básicas que conocemos; suma (+), resta (-), multiplicación (*), potencia (^) y asignación (:=)</p> <p>Por otro lado, tenemos los OPERADORES LÓGICOS, si es verdadero o falso (1 o 0). Basta con recordar las compuertas lógica:</p> <p>-AND (&). Si dos expresiones son verdaderas el resultado es verdadero, de lo contrario es falso</p> <p>-OR (Disyunción, si por lo menos existe un valor verdadero el resultado es verdadero.</p> <p>-NOT (!) Es el lo contrario del resultado</p>
NOTACIÓN DE CAMELLO	Al momento de declarar variables y estas sean dos palabras, empleo Número de alumnos, las juntamos y la notación de camello se basa es que la letra "n" de número y la letra "a" de alumnos van en mayúscula, ejemplo: NumAlumnos o también puede ir numAlumnos

ESTRUCTURA DE CONTROL DE FLUJO

Una estructura de control de flujo ayuda a la repetición de instrucciones y existen 3 tipos:

SECUENCIAL

Se caracteriza por tener declaraciones, es decir una letra puede tomar un valor numérico para usarlo a lo largo de todo el pseudocódigo.



The screenshot shows the PSeInt IDE with a file named 'sin_titulo.pse'. The code is as follows:

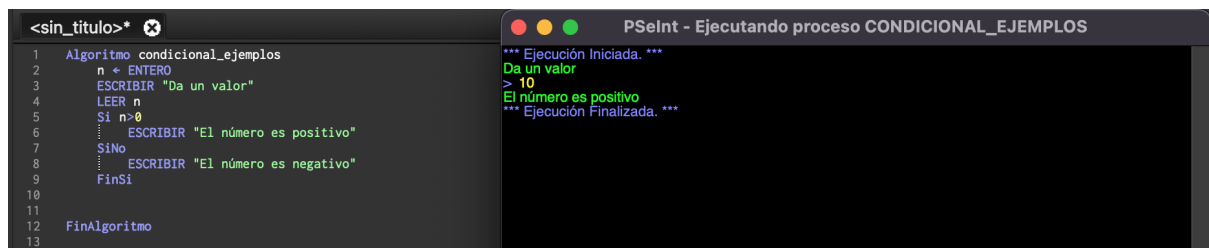
```
1 Algoritmo secuencial_ejemplos
2 resultado_1←ENTERO
3 resultado_2← ENTERO
4
5     x:=58
6     resultado_1=x*10
7     resultado_2=x*20
8     ESCRIBIR " ",resultado_1
9     ESCRIBIR " ",resultado_2
10 FinAlgoritmo
11
```

The execution output on the right shows:

```
*** Ejecución Iniciada. ***
580
1160
*** Ejecución Finalizada. ***
```

CONDICIONAL

Al hablar de esta estructura es importante recordar que una condición es algo que no solo tiene una respuesta, en este caso son dos respuesta (verdadera o falsa). El ejemplo más claro es que un número sea mayor que 0. Si el número cumple (verdadero) el número será positivo, por otro lado, si no cumple (falso) el número es negativo.

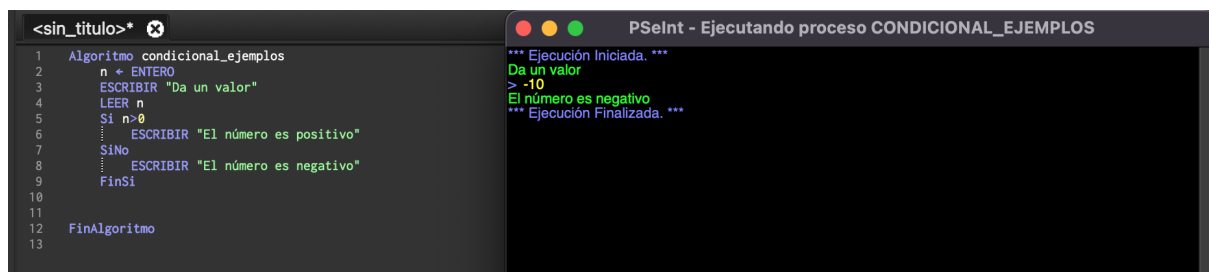


The screenshot shows the PSeInt IDE with a file named 'sin_titulo.pse'. The code is as follows:

```
1 Algoritmo condicional_ejemplos
2 n ← ENTERO
3 ESCRIBIR "Da un valor"
4 LEER n
5 Si n>0
6     ESCRIBIR "El número es positivo"
7 SiNo
8     ESCRIBIR "El número es negativo"
9 FinSi
10
11
12 FinAlgoritmo
13
```

The execution output on the right shows:

```
*** Ejecución Iniciada. ***
Da un valor
> 10
El número es positivo
*** Ejecución Finalizada. ***
```



The screenshot shows the PSeInt IDE with a file named 'sin_titulo.pse'. The code is as follows:

```
1 Algoritmo condicional_ejemplos
2 n ← ENTERO
3 ESCRIBIR "Da un valor"
4 LEER n
5 Si n>0
6     ESCRIBIR "El número es positivo"
7 SiNo
8     ESCRIBIR "El número es negativo"
9 FinSi
10
11
12 FinAlgoritmo
13
```

The execution output on the right shows:

```
*** Ejecución Iniciada. ***
Da un valor
> -10
El número es negativo
*** Ejecución Finalizada. ***
```

REPETITIVA

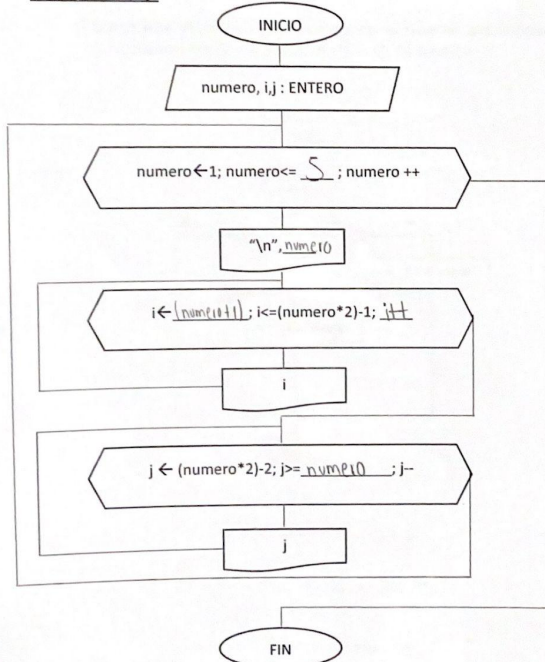
También llamada ciclos permite hacer una serie de instrucciones mientras se cumpla la expresión y en este caso se encuentran dos tipos:

HACER	HACER-MIENTRAS
Primero observa la condición y ejecuta, posteriormente regresa a la condición para ver si aún sigue la condición por lo que termina hasta que ésta se cumpla o sea falsa	Primero realiza las instrucciones y después observa la condición, el código regresa a la instrucción

Nombre: ROSALES GÓMEZ EVER JUAN

Completa lo faltante en cada etapa de solución del problema

Análisis: (Colocar en la parte de atrás de la hoja)
Diagrama de flujo



Pseudocódigo

INICIO
numero, i, j: ENTERO
Para numero = 1 Hasta 5 Con incremento numero++ Hacer
 Escribir "\n", numero;
 Para i = numero+1 Hasta (numero*2)-1 Con incremento i++ Hacer
 Escribir "i"
 Fin Para
 Para j = (numero*2)-2 Hasta numero Con decremento j-- Hacer
 Escribir "j"
 Fin Para
Fin Para
FIN

Programa

```

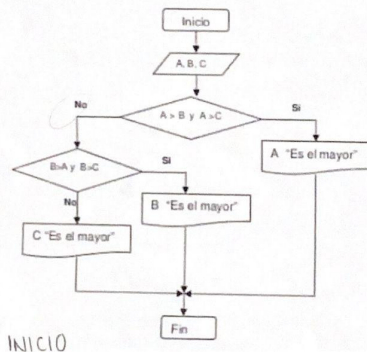
#include <stdio.h>
int main()
{
    int numero, i, a, j;
    for(numero=1; numero<=5; numero++)
    {
        printf("\n%d", numero);
        for(i=numero+1; i<=(numero*2)-1; i++)
        {
            printf("i");
        }
        for(j=(numero*2)-2; j>=numero; j--)
        {
            printf("j");
        }
    }
    return 0;
}
  
```

Ejecución

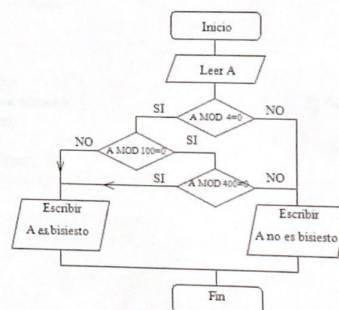
(Dibuja la salida)

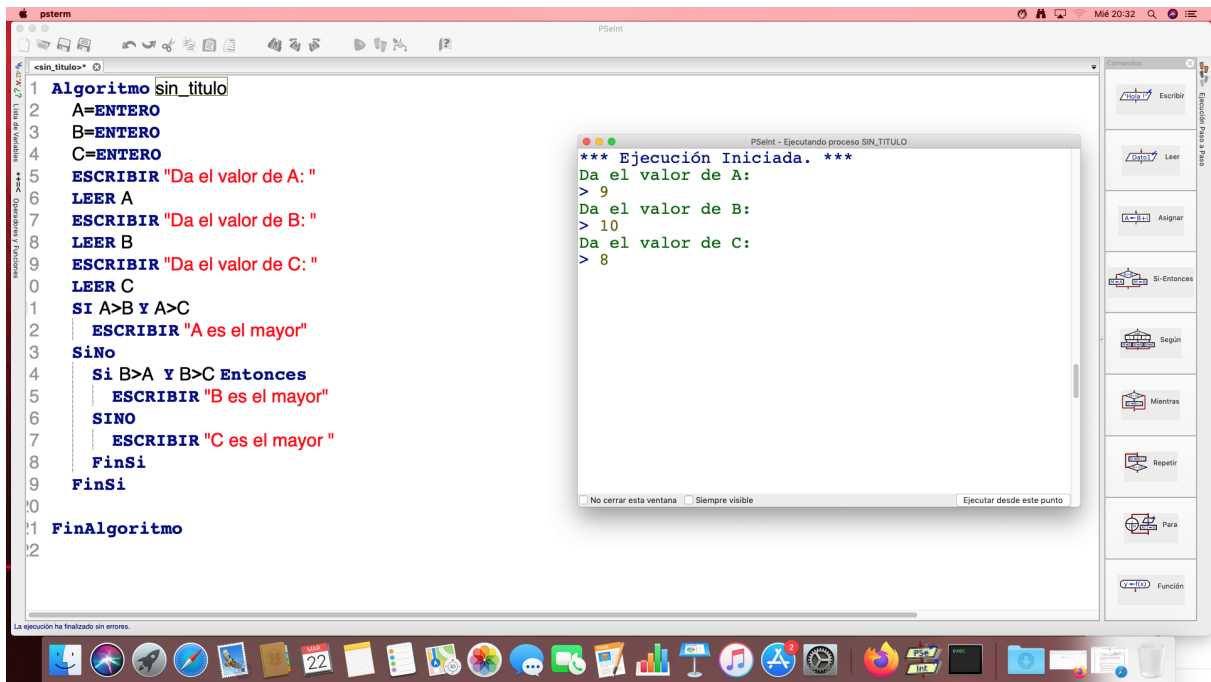
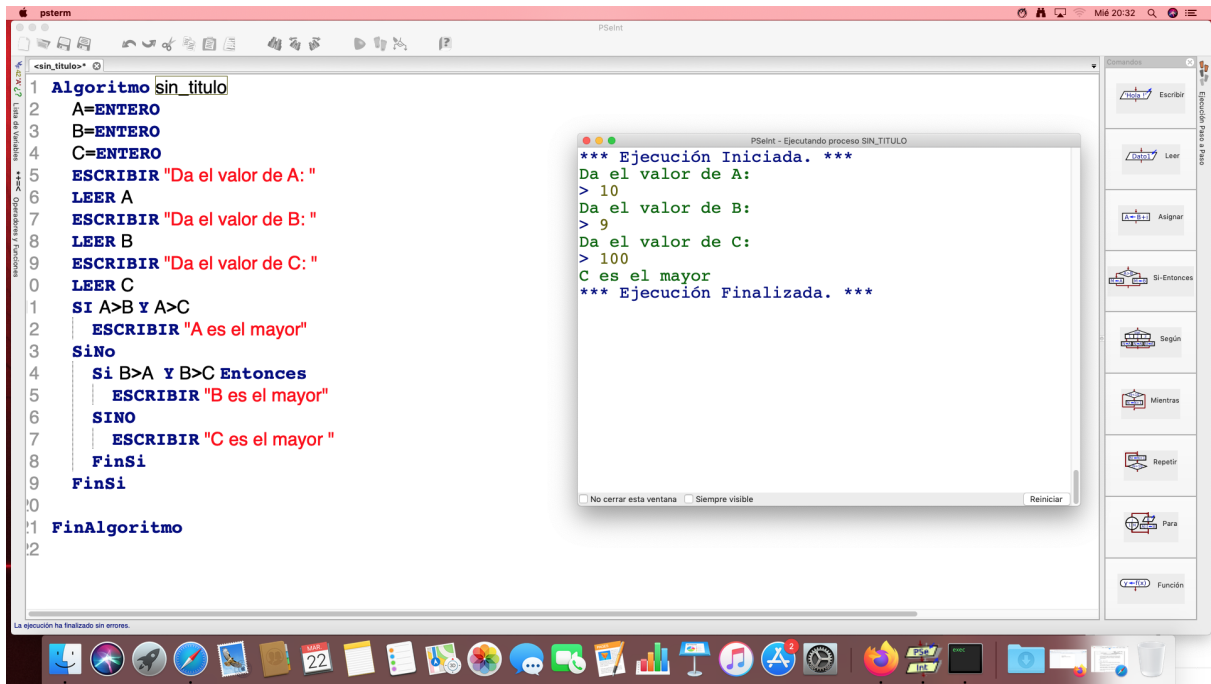
Genera el análisis y pseudocódigo de los siguientes diagramas:

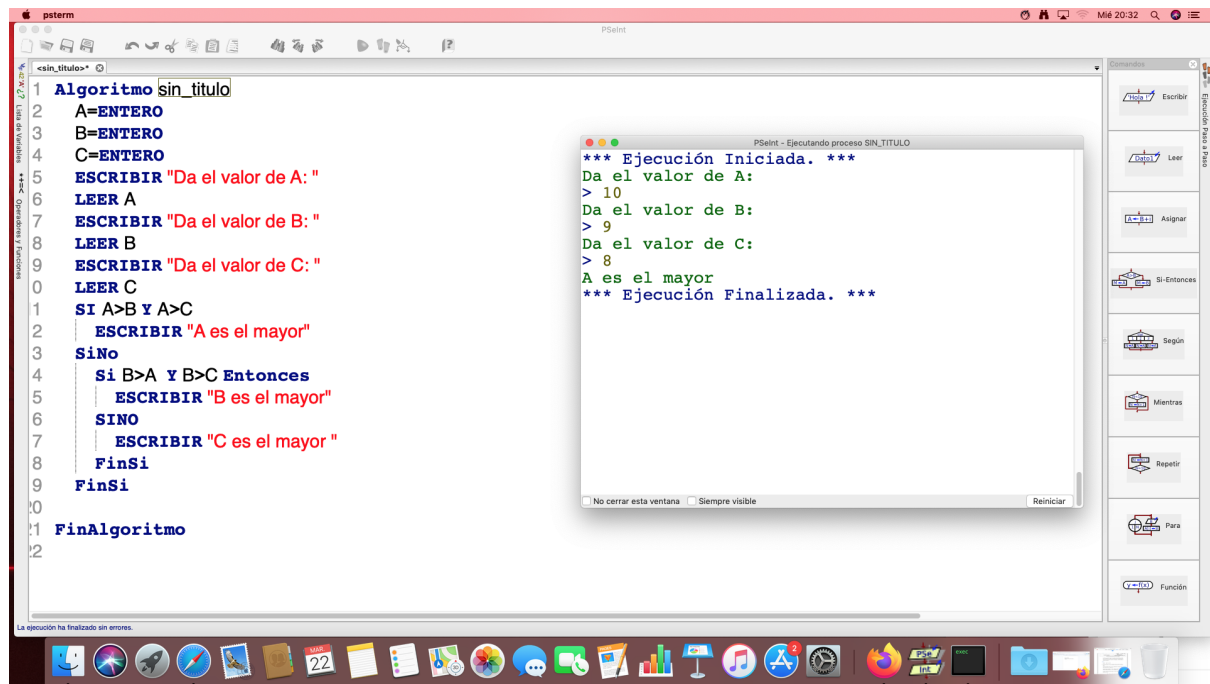
Diagrama de flujo de procesos en el que se almacenen 3 números en 3 variables A, B y C. El menor



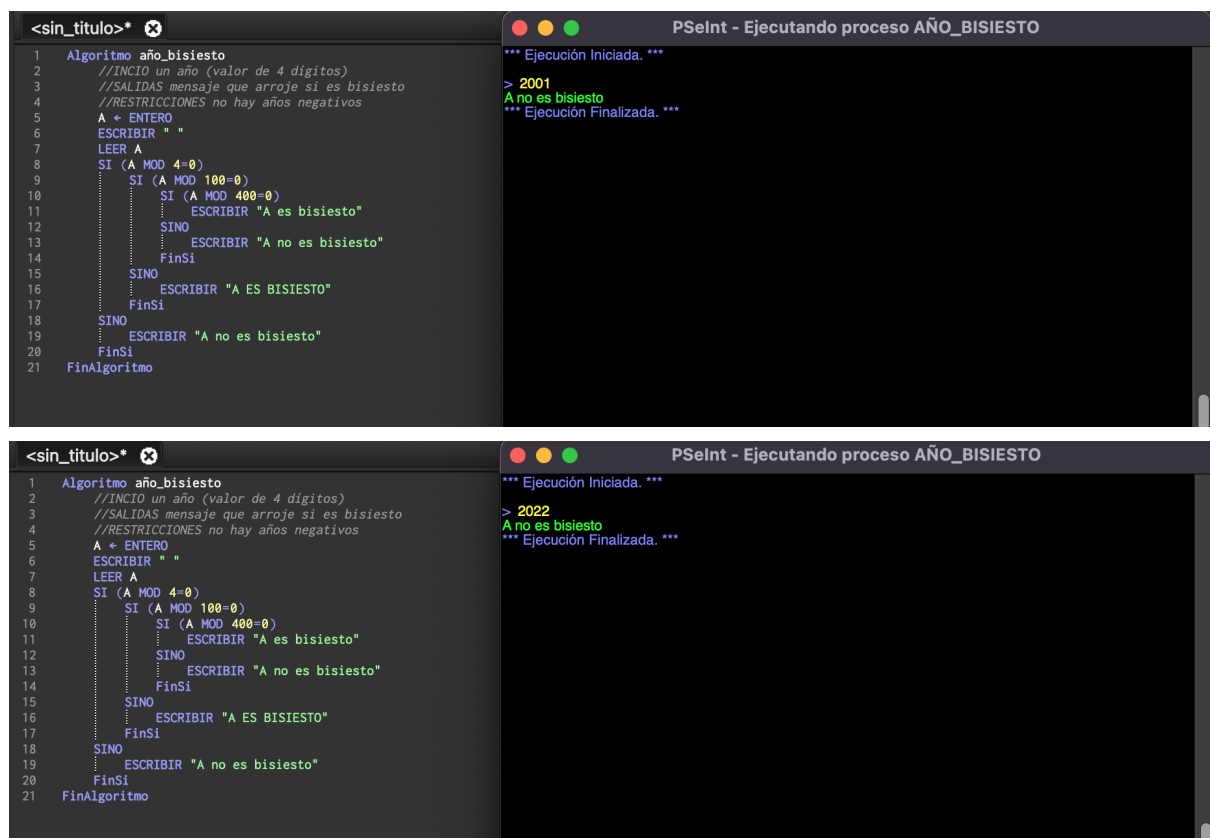
Un año es bisiesto si es múltiplo de 4, exceptuando los múltiplos de 100, que sólo son bisiestos cuando son múltiplos además de 400, por ejemplo, el año 1900 no fue bisiesto, pero el año 2000 si lo será. Diagrama que dado un año A nos diga si es o no bisiesto.







Pseudocódigo de un año bisiesto



<sin_titulo>*

```

1 Algoritmo año_bisiesto
2 //INCIO un año (valor de 4 dígitos)
3 //SALIDAS mensaje que arroje si es bisiesto
4 //RESTRICCIONES no hay años negativos
5 A ← ENTERO
6 ESCRIBIR " "
7 LEER A
8 SI (A MOD 4=0)
9     SI (A MOD 100=0)
10        SI (A MOD 400=0)
11            ESCRIBIR "A es bisiesto"
12        SINO
13            ESCRIBIR "A no es bisiesto"
14        FinSi
15    SINO
16        ESCRIBIR "A ES BISIESTO"
17    FinSi
18 SINO
19     ESCRIBIR "A no es bisiesto"
20 FinSi
21 FinAlgoritmo

```

PSeInt - Ejecutando proceso AÑO_BISIESTO

```

*** Ejecución Iniciada. ***
> 2004
A ES BISIESTO
*** Ejecución Finalizada. ***

```

EJERCICIOS DE TAREA

1. Algoritmo que valide la calificación que ingrese el usuario, esta debe estar entre 5 y 10. Indicar si ha aprobado con calificación mayor a 6; un letrero "Aprobado, felicidades", en caso de no aprobar "Reprobado, tienes una nueva oportunidad"

<sin_titulo>*

```

1 Algoritmo EJERCICIO_1
2 //ENTRADA: calificación
3 //SALIDA: Escrito si es aprobado o no
4 //RESTRICCIONES: no hay calificaciones negativas
5 calif ← ENTERO
6 ESCRIBIR "Ingresa el valor de tu calificación: "
7 LEER calif
8 SI (calif≤5)
9     ESCRIBIR "Reprobado, tienes una nueva oportunidad"
10 SiNo
11     ESCRIBIR "Aprobado, felicidades"
12 FinSi
13 FinAlgoritmo

```

PSeInt - Ejecutando proceso EJERCICIO_1

```

*** Ejecución Iniciada. ***
Ingresa el valor de tu calificación:
> 10
Aprobado, felicidades
*** Ejecución Finalizada. ***

```

<sin_titulo>*

```

1 Algoritmo EJERCICIO_1
2 //ENTRADA: calificación
3 //SALIDA: Escrito si es aprobado o no
4 //RESTRICCIONES: no hay calificaciones negativas
5 calif ← ENTERO
6 ESCRIBIR "Ingresa el valor de tu calificación: "
7 LEER calif
8 SI (calif≤5)
9     ESCRIBIR "Reprobado, tienes una nueva oportunidad"
10 SiNo
11     ESCRIBIR "Aprobado, felicidades"
12 FinSi
13 FinAlgoritmo

```

PSeInt - Ejecutando proceso EJERCICIO_1

```

*** Ejecución Iniciada. ***
Ingresa el valor de tu calificación:
> 5
Reprobado, tienes una nueva oportunidad
*** Ejecución Finalizada. ***

```

2. Del ejercicio de clase (el menú) modifica con el ciclo hacer mientras, en donde valide, si el usuario no elige las opciones 1 ó 2 para caracteres deben ser 'a', 'b' o 'c') imprimir un letrero de "error"(es el default) y volver a mostrarle el menú.
3. Algoritmo que muestre la numeración del 1 al 1000, emplea la estructura MIENTRAS, HACER MIENTRAS Y PARA


```
<sin_titulo>*
1  Algoritmo EJERCICIO_3
2  //ENTRADA: valores entre 1 y 1000
3  //SALIDA: Imprimir los valores
4  //RESTRICCIONES: no valores negativos
5  Para i=1 Hasta 1000 Hacer
6  |   ESCRIBIR " ",i
7  FinPara
8  FinAlgoritmo
9
```

Errores de Sintaxis

La sintaxis es correcta. Puede presionar F9 para ejecutar el algoritmo.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57

```
<sin_titulo>*
1  Algoritmo EJERCICIO_3
2  //ENTRADA: valores entre 1 y 1000
3  //SALIDA: Imprimir los valores
4  //RESTRICCIONES: no valores negativos
5  Para i=1 Hasta 1000 Hacer
6  |   ESCRIBIR " ",i
7  FinPara
8  FinAlgoritmo
9
```

Errores de Sintaxis

La sintaxis es correcta. Puede presionar F9 para ejecutar el algoritmo.

945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
*** Ejecución Finalizada. ***

4. Solicitar al usuario que ingrese la variable, si ingresa una variable diferente a la letra 'a' o 'A', volver a solicitarla, en caso de contrario, imprimir la letra 'a' o 'A' que ingresó el usuario


```

<sin_titulo>* ✕
1  Algoritmo EJERCICIO_4
2      //ENTRADA: funcion
3      //SALIDA: Llamarla si la funcion es igual a o A
4      //RESTRICCIONES: no hay letras negativas
5      variable ← CARACTER
6      A ← CARACTER
7      a ← CARACTER
8
9      ESCRIBIR "Ingresa una variable"
10     LEER variable
11     SI (variable=A y variable#a)Entonces
12     ..... ESCRIBIR "Ingresa otra variable"
13     SINO
14     ..... ESCRIBIR " ",variable
15     FinSi
16 FinAlgoritmo

```

NOTA: Al hacer las pruebas no funciona, desconozco el motivo, intenté hacerlo al revés, poniendo en la condición (var==a Y var==A) pero no arroja la letra a o A en caso de que el usuario pusieras esas letras

CONCLUSIÓN:

Se consiguieron los objetivos ya que al resolver problemas comunes que se prestan en nuestra vida diaria el uso de las estructuras son de gran utilidad.

Para conocer si el pseudocódigo funciona perfectamente es importante realizar las pruebas de escritorio. También es importante conocer el tipo de variable que vamos a declarar pues esto nos ayuda a identificar las salida que debe tener el problema y puede que se ahorre espacio o necesitemos, por ejemplo; si queremos sacar el promedio de calificaciones, es recomendable declarar la variable “promedio” como REAL, pues sabremos los decimales exactos y podemos poner una condición en este caso si el promedio es igual o mayor (\geq) a 6 sube la calificación, en caso contrario se mantiene la calificación.