Proyecto integrador: sistema de notas

CURSO	MODALIDAD	TIEMPO	VALOR
FE729	Individual, por etapas	2 h 30 min en PSeInt	100 puntos

Nombre _____ **Carné** _____

Descripción general

Construirá, paso a paso, un programa en PSeInt que administra las notas de **cinco estudiantes**. Las notas van de 0 a 100 y se aprueba con **61** o más. El programa funciona con un menú que se repite hasta que el usuario elige *Salir*. Todo se escribe en un solo archivo: `P04_SistemaNotas_<su_carné>.psc`.

```
===== SISTEMA DE NOTAS FE729 =====
1. Ingresar notas
2. Mostrar notas y categorias
3. Estadisticas
4. Sumar puntos extra
5. Salir
Opcion:
> 3
Promedio: 68
Nota mas alta: 90
Aprobados: 3
```

REGLA DEL PROYECTO

No está permitido usar `Por Referencia` en ningún `SubProceso` ni `Funcion`.

- Si un módulo debe **devolver** información, use una `Funcion` y guarde el valor con una asignación en el principal: `hayDatos <- CargarNotas(notas, CANTIDAD)`.
- Los **arreglos** se pasan como parámetro normal y el módulo puede leerlos y modificarlos directamente.

Las cinco etapas

Etapa	Qué construye	Tiempo	Puntos
1	El menú principal y el aviso de “no hay notas” (plantilla)	15 min	15
2	Ingresar, validar, clasificar y mostrar las notas	50 min	25
–	*Receso*	15 min	–
3	Estadísticas con tres funciones	35 min	25
4	Puntos extra validados, con tope de 100	25 min	20
5	El menú como módulo, pruebas finales y orden	25 min	15

ADVERTENCIA

Cada etapa termina en un **punto de control**: el instructor la revisa antes de que usted siga. No avance si la etapa anterior no funciona: los errores se acumulan.

Etapa 1. El menú principal — 15 min

El instructor le entrega la plantilla `P04_SistemaNotas_plantilla.psc`: el menú ya funciona con `Repetir` y `Segun`, y estos datos ya están declarados.

Dato	Tipo	Para qué sirve
`CANTIDAD = 5`	Constante	Cuántos estudiantes hay
`NOTA_MINIMA = 61`	Constante	La nota para aprobar
`notas[5]`	Arreglo de `Real`	Las cinco notas
`hayDatos`	`Logico`	

1. Abra la plantilla y guárdela como `P04_SistemaNotas_<su_carné>.psc`.
2. En los comentarios del inicio, escriba su nombre, su carné y la fecha.
3. Ejecute (`F9`) y pruebe las opciones 2, 9 y 5.
4. Las opciones 2, 3 y 4 no deben trabajar sin notas. Protéjalas con `hayDatos`:

</> OPCIÓN 2 DEL MENÚ: COMPLETE LOS ESPACIOS

```
      2:
        Si _________________ Entonces
          Escribir "(pendiente: etapa 2)"
        SiNo
          ____________________________________
        FinSi
```

Haga lo mismo en las opciones 3 y 4. Luego pruebe:

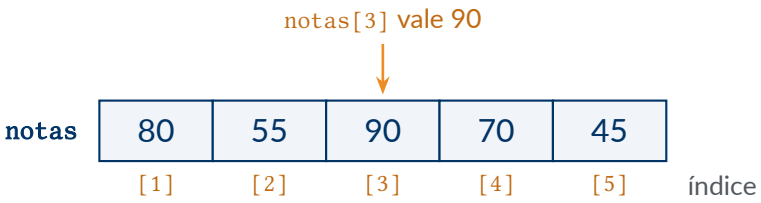
Escriba	Qué pasa
2	
9	
5	

RECOMENDACIÓN

Punto de control. Las opciones 2, 3 y 4 avisan que no hay notas. Levante la mano para que el instructor lo revise y guarde el archivo (`Ctrl+S`).

Ayuda: cómo se usa un arreglo en PSeInt

Lea esta ayuda antes de la etapa 2 y vuelva a ella cada vez que dude. Un **arreglo** guarda varios datos del mismo tipo con **un solo nombre**. Cada casilla tiene un número, el **índice**, que se escribe entre corchetes. En PSeInt (perfil Flexible) los índices empiezan en **1**.



Crear el arreglo y usar una casilla

</> CREAR (YA ESTÁ EN LA PLANTILLA)

```
Definir notas Como Real
Dimension notas[5]
```

Primero el tipo con `Definir`; después el tamaño con `Dimension`.

Instrucción	Qué hace
`notas[1] <- 80`	Guarda 80 en la casilla 1
`Leer notas[2]`	Lee del teclado y lo guarda en la casilla 2
`Escribir notas[3]`	Muestra lo que hay en la casilla 3
`notas[4] <- notas[4] + 5`	Le suma 5 a la casilla 4

Recorrer el arreglo con `Para`: la idea clave

Con un `Para`, la variable `i` vale 1, 2, 3… en cada vuelta. Si dentro del ciclo escribe `notas[i]`, en cada vuelta trabaja con **otra casilla**. Así, con dos o tres líneas se recorren todas. Este ejemplo usa otro arreglo, `precios`, para que vea el molde:

</> EJEMPLO: LLENAR Y MOSTRAR TRES PRECIOS

```
Algoritmo EjemploArreglo
  Definir precios Como Real
  Definir i Como Entero
  Dimension precios[3]
  // Llenar: una casilla por vuelta
  Para i <- 1 Hasta 3 Con Paso 1 Hacer
    Escribir "Precio ", i, ":"
    Leer precios[i]
  FinPara
  // Mostrar: una casilla por vuelta
  Para i <- 1 Hasta 3 Con Paso 1 Hacer
    Escribir i, ". Q", precios[i]
  FinPara
FinAlgoritmo
```

Vuelta	`i`	Casilla que usa
1	1	`precios[1]`
2	2	`precios[2]`
3	3	`precios[3]`

RECOMENDACIÓN

Cópielo en un archivo aparte y ejecútelo **paso a paso**: verá cómo cambia `i` y qué casilla se llena.

Pasar el arreglo a un módulo

- En el encabezado y en la llamada se escribe **solo el nombre**, sin corchetes.
  Encabezado: `SubProceso MostrarNotas(notas, n)`.
  Llamada: `MostrarNotas(notas, CANTIDAD)`.
- Se envía también `n`, la cantidad de casillas, para que el módulo sepa hasta dónde llega el `Para`.
- Dentro del módulo **no** se vuelve a escribir `Dimension`: se usa `notas[i]` igual que en el principal.

Errores frecuentes

Lo que escribió	Qué pasa	Corrección
`notas[0]` o `notas[6]`	PSeInt avisa que el índice está fuera de rango	Los índices van de 1 a 5
`Escribir notas`	No muestra las cinco notas	Use un `Para` con `notas[i]`
`notas[5]` dentro del `Para`	Siempre usa la misma casilla	Use `notas[i]`
`MostrarNotas(notas[5], n)`	Envía una sola nota, no el arreglo	Sin corchetes: `MostrarNotas(notas, n)`
Olvidar `Dimension`	PSeInt no sabe el tamaño del arreglo	Primero `Definir`, luego `Dimension`

Compruebe que entendió. Con las notas del dibujo (80, 55, 90, 70 y 45):

Pregunta	Respuesta
¿Cuánto vale `notas[2]`?	
¿Cuánto vale `notas[1] + notas[5]`?	
Después de `notas[4] <- notas[4] + 10`, ¿cuánto vale `notas[4]`?	
Si `i` vale 3, ¿qué casilla es `notas[i]` y cuánto vale?	

Etapa 2. Ingreso y clasificación de notas

50 min

Todos los módulos se escriben **arriba** de la línea `Algoritmo SistemaDeNotas`. Si duda con el arreglo, revise la ayuda anterior.

a) La función `LeerNotaValida`

Pide la nota del estudiante `numero` y la valida con un `Mientras`: mientras la nota sea menor que 0 o mayor que 100, la vuelve a pedir. Al final **devuelve** la nota válida.

</> FUNCIÓN LEERNOTAVALIDA: COMPLETE LOS ESPACIOS

```
Funcion nota <- LeerNotaValida(numero)
  Definir nota Como Real
  Escribir "Nota del estudiante ", numero, " (0 a 100):"
  Leer nota
  Mientras _____________________________ Hacer
    Escribir "Nota fuera de rango. Escribala de nuevo:"
    ____________________
  FinMientras
FinFuncion
```

RECOMENDACIÓN

Para la condición, piense: ¿puede una nota ser menor que 0 *y* mayor que 100 al mismo tiempo? Elija entre `Y` y `O`. Y no olvide el `Leer` dentro del ciclo: sin él, el ciclo nunca termina.

b) La función `CargarNotas`

Llena el arreglo con un `Para`, llamando a `LeerNotaValida(i)` en cada posición, y al terminar **devuelve** `Verdadero`.

</> FUNCIÓN CARGARNOTAS: COMPLETE LOS ESPACIOS

```
Funcion listo <- CargarNotas(notas, n)
  Definir listo Como Logico
  Definir i Como Entero
  Para i <- 1 Hasta __________ Con Paso 1 Hacer
    notas[i] <- ______________________________
  FinPara
  listo <- ___________________
FinFuncion
```

En la opción 1 del menú, borre el `Escribir "(pendiente…)"` y escriba:

```
      1:
        hayDatos <- CargarNotas(notas, CANTIDAD)
```

> **Así se actualiza la bandera sin `Por Referencia`.** El arreglo sí se llena dentro de la función, porque los arreglos se modifican directamente. Pero `hayDatos` es una variable simple: por eso la función *devuelve* `Verdadero` y el principal lo guarda en la bandera.

Prueba. Elija 1. Cuando pida la nota del estudiante 1, escriba `105`, luego `-5` y luego `80`. ¿Qué hizo el programa?

c) La función `Categoria`

Recibe una nota y **devuelve** el texto de su categoría.

Rango	Categoría
90 a 100	`Sobresaliente`
76 a 89	`Notable`
61 a 75	`Satisfactorio`
0 a 60	`Insuficiente`

</> ENCABEZADO QUE DEBE USAR

```
Funcion cat <- Categoria(nota)
```

```
  Definir cat Como Cadena
  // ... escriba aqui los Si anidados ...
FinFuncion
```

RECOMENDACIÓN

Es la parte B de la práctica 2, ahora como función. Empiece por la categoría más alta: si la nota no es 90 o más, pregunte si es 76 o más, y así. Con tres comparaciones basta.

d) El subproceso `MostrarNotas`

Escriba el `SubProceso MostrarNotas(notas, n)`. Con un `Para` de 1 a `n`, escriba una línea por estudiante con este formato: `1. 85 - Notable`. La categoría de cada nota es `Categoria(notas[i])`, y puede ir dentro del mismo `Escribir`.

En la opción 2, cambie el `Escribir "(pendiente…)"` por la llamada:

</> OPCIÓN 2 DEL MENÚ: COMPLETE EL ESPACIO

```
      2:
        Si hayDatos Entonces
          ____________________________________
        SiNo
          Escribir "Todavia no hay notas."
        FinSi
```

Prueba. Cargue `80`, `55`, `90`, `70` y `45`, y elija 2.

Estudiante	Nota	Categoría esperada	¿Coincide?
1	80		
2	55		
3	90		
4	70		
5	45		

RECOMENDACIÓN

Punto de control. Las notas fuera de 0 a 100 se vuelven a pedir y las cinco categorías son correctas. Levante la mano para que el instructor lo revise y guarde el archivo (`Ctrl+S`).

Etapa 3. Estadísticas

35 min

Escriba tres funciones que **devuelven** su resultado. Las tres reciben el arreglo y `n`, recorren las notas con un `Para` y no escriben nada en pantalla: solo calculan. Complete `Promedio`:

FUNCIÓN PROMEDIO: COMPLETE LOS ESPACIOS

```
Funcion prom <- Promedio(notas, n)
  Definir prom, suma Como Real
  Definir i Como Entero
  suma <- __________
  Para i <- 1 Hasta n Con Paso 1 Hacer
    suma <- ____________________________
  FinPara
  prom <- ____________________
FinFuncion
```

Con el mismo molde, escriba las otras dos:

Encabezado	Guía
`Funcion mayor <- NotaMayor(notas, n)`	Empiece con `mayor <- notas[1]` y recorra desde la segunda nota. Si una nota es mayor que `mayor`, guárdela.
`Funcion cuantos <- ContarAprobados(notas, n, minima)`	Es un **contador**: empieza en 0 y aumenta en 1 cada vez que una nota es mayor o igual que `minima`.

Seguimiento a mano. Antes de ejecutar, complete cómo cambia `mayor` con las notas 80, 55, 90, 70 y 45.

i	notas[i]	¿Es mayor?	mayor
—	—	—	80
2	55		
3	90		
4	70		
5	45		

En la opción 3, dentro del `Si hayDatos`, escriba tres líneas: `Promedio:`, `Nota mas alta:` y `Aprobados:`, cada una seguida de la llamada a su función. A `ContarAprobados` pásele la constante `NOTA_MINIMA`, no el número 61.

NOTA

Una función devuelve un valor que hay que **escribir o guardar**. Si la llama sola en una línea, el valor se pierde.

Calcule a mano con las mismas cinco notas: promedio nota más alta aprobados

RECOMENDACIÓN

Punto de control. La opción 3 muestra lo mismo que calculó a mano. Levante la mano para que el instructor lo revise y guarde el archivo (`Ctrl+S`).

Etapa 4. Puntos extra

25 min

a) Validar los puntos en la opción 4

Pida cuántos puntos sumar y valide con un `Mientras` que estén entre 0 y 20:

OPCIÓN 4 DEL MENÚ: COMPLETE LOS ESPACIOS

```
      4:
        Si hayDatos Entonces
          Escribir "Puntos a sumar (0 a 20):"
          Leer puntos
          Mientras _____________________________ Hacer
            Escribir "Solo de 0 a 20. Escribalos de nuevo:"
            ____________________
          FinMientras
          ____________________________________
          Escribir "Puntos aplicados."
        SiNo
          Escribir "Todavia no hay notas."
        FinSi
```

b) El subproceso `SumarPuntos`

Escriba el `SubProceso SumarPuntos(notas, n, puntos)`. Con un `Para`, sume `puntos` a cada nota; si una nota pasa de 100, déjela en 100. Este módulo **modifica el arreglo directamente**, así que no necesita devolver nada.

RECOMENDACIÓN

El tope es un `Si` dentro del `Para`, justo después de sumar: pregunte si `notas[i]` quedó mayor que 100.

Prueba. Con las notas 80, 55, 90, 70 y 45, elija 4 y escriba `25`; luego `15`. Después elija 2. ¿Cómo quedaron las notas?

RECOMENDACIÓN

Punto de control. Los puntos fuera de 0 a 20 se vuelven a pedir y ninguna nota pasa de 100. Levante la mano para que el instructor lo revise y guarde el archivo (`Ctrl+S`).

Etapa 5. El menú como módulo y pruebas finales

25 min

Saque la impresión del menú a un `SubProceso MostrarMenu` **sin parámetros**:

1. Arriba de `Algoritmo`, escriba el encabezado `SubProceso MostrarMenu` y su `FinSubProceso`.
2. **Mueva** adentro los ocho `Escribir` del menú (el título, las cinco opciones y `Opcion:`).
3. En el `Repetir`, donde estaban esos `Escribir`, llame al subproceso justo antes de `Leer opcion`.

</> EL INICIO DEL REPETIR: COMPLETE EL ESPACIO

```
Repetir
  ____________________
  Leer opcion
  Segun opcion Hacer
```

ADVERTENCIA

`MostrarMenu` no recibe parámetros: se escribe *sin paréntesis*, al declararlo y al llamarlo. Con `()` vacíos, PSeInt marca un error.

Plan de pruebas final

Cierre PSeInt, ábralo de nuevo y ejecute los ocho casos en orden. Escriba el resultado esperado *antes* de probar y marque si funcionó.

No.	Caso	Qué escribir	Resultado esperado	¿Funcionó?
1	Sin datos	Opción `2` al iniciar		
2	Opción inválida	`9`		
3	Validación	Opción `1`; luego `105`, `-5` y `80`		
4	Carga	`55`, `90`, `70`, `45`; opción `2`		
5	Estadísticas	Opción `3`		
6	Puntos fuera de rango	Opción `4` con `25`		
7	Puntos extra	Luego `15`; opciones `2` y `3`		
8	Salir	`5`		

RECOMENDACIÓN

Punto de control. El menú se ve igual que antes y los ocho casos dan lo esperado. Levante la mano para que el instructor lo revise y guarde el archivo (`Ctrl+S`).

Preguntas

1. ¿Por qué `CargarNotas` devuelve `Verdadero` y `SumarPuntos` no devuelve nada, si los dos cambian el arreglo?

2. ¿Por qué `Categoria` es una función y `MostrarNotas` es un subproceso?

3. ¿Por qué el menú usa `Repetir` y no `Mientras`?

4. Escriba la condición de los puntos extra de otra forma, usando `NO` y `Y`.

Restricciones y criterios de evaluación

- ✗ Prohibido usar `Por Referencia` en cualquier encabezado de módulo.
- ✓ Use `Funcion` y una asignación para devolver valores.
- ✓ Los arreglos se pasan como parámetro normal.
- ✓ Todas las entradas del usuario se validan con ciclos: las notas, los puntos y la opción del menú.
- ✓ Use las constantes `CANTIDAD` y `NOTA_MINIMA` en lugar de los números 5 y 61.

Criterio	Qué se revisa	Puntos
Etapa 1: menú	Se repite hasta 5, rechaza opciones inválidas y avisa si no hay notas	15
Etapa 2: ingreso y clasificación	`LeerNotaValida` con `Mientras`, `CargarNotas` devuelve `Verdadero`, categorías y formato correctos	25
Etapa 3: estadísticas	Tres funciones que devuelven su valor; 68, 90 y 3 con las notas de prueba	25
Etapa 4: puntos extra	Validación de 0 a 20 y tope de 100 en `SumarPuntos`	20
Etapa 5: menú y orden	`MostrarMenu` sin parámetros, sin `Por Referencia`, constantes, comentarios y los ocho casos	15
	Total	**100**

Entrega

- ☐ El archivo se llama `P04_SistemaNotas_<su_carné>.psc`.
- ☐ Tiene su nombre, su carné y la fecha en los comentarios del inicio.
- ☐ Todos los módulos están arriba de `Algoritmo` y ninguno usa `Por Referencia`.
- ☐ Los ocho casos de prueba dieron lo esperado.
- ☐ Está subido a la carpeta `Practica_04` de su repositorio `Algoritmos-FE729`, con un `README` de tres líneas.

Nivel extra (para quien termine antes)

1. Agregue la función `NotaMenor` y muéstrela en la opción 3.
2. Muestre también el promedio entero en decenas y unidades con `DIV` y `MOD`: 68 son 6 decenas y 8 unidades.
3. Opción 6: cuente cuántos estudiantes hay en cada categoría (mueva *Salir* a la opción 7).
4. Guarde también los nombres en un segundo arreglo `nombres[5]` de tipo `Cadena` y muéstrelos en la opción 2.