UNIVERSIDAD RURAL DE GUATEMALA

FACULTAD DE INGENIERÍA
Ingeniería en Sistemas con especialización en IA y Ciberseguridad

Práctica 4: todo lo visto en el curso, en un solo programa (unidades 1 a 6)

Proyecto integrador: un sistema de notas

Algoritmos • FE729 • Sesión 4 de 4

Ing. Mynor A. Cifuentes

Universidad Rural de Guatemala • Laboratorio presencial • 1 de octubre de 2026

Agenda de hoy

Última clase: casi toda la jornada es un solo proyecto

APERTURA

- 10 min — Repaso relámpago
- 20 min — **Etapa 0.** Análisis en papel

Hoy no hay tema nuevo: se juntan las sesiones 1, 2 y 3 en un solo programa.

CIERRE

- 25 min — Evaluación en Socrative
- 20 min — Cierre del curso

PRÁCTICA FINAL EN PSEINT

- 35 min — **Etapa 1.** El menú
- 35 min — **Etapa 2.** Cargar y mostrar
- 15 min — Receso
- 40 min — **Etapa 3.** Estadísticas
- 25 min — **Etapa 4.** Puntos extra y pruebas
- 15 min — **Etapa 5.** Toques finales

NOTA

Cada etapa termina en un **punto de control**: el instructor revisa que funcione y usted sigue.

01 PARTE
ALGORITMOS

Repaso relámpago

¿Lo recuerdan?

Una pregunta por cada idea que hoy va a necesitar

Ses.	Pregunta	Respuesta
1	¿Qué significan las letras EPS?	
1	¿Cómo se declara una variable real `suma`?	
2	¿Qué estructura elige entre varias opciones de un número?	
2	¿Qué ciclo conviene si se sabe cuántas vueltas dar?	
3	¿Qué tiene una `Funcion` que no tiene un `SubProceso`?	
3	¿Qué hace `Por Referencia`?	

LO QUE SIGUE

En tres días aprendió a analizar, decidir, repetir, guardar listas y dividir el trabajo en módulos.

Hoy lo usa **todo a la vez**, como en un programa real.

¿Lo recuerdan?

Una pregunta por cada idea que hoy va a necesitar

Ses.	Pregunta	Respuesta
1	¿Qué significan las letras EPS?	Entrada, Proceso, Salida
1	¿Cómo se declara una variable real `suma`?	
2	¿Qué estructura elige entre varias opciones de un número?	
2	¿Qué ciclo conviene si se sabe cuántas vueltas dar?	
3	¿Qué tiene una `Funcion` que no tiene un `SubProceso`?	
3	¿Qué hace `Por Referencia`?	

LO QUE SIGUE

En tres días aprendió a analizar, decidir, repetir, guardar listas y dividir el trabajo en módulos.

Hoy lo usa **todo a la vez**, como en un programa real.

¿Lo recuerdan?

Una pregunta por cada idea que hoy va a necesitar

Ses.	Pregunta	Respuesta
1	¿Qué significan las letras EPS?	Entrada, Proceso, Salida
1	¿Cómo se declara una variable real `suma`?	`Definir suma Como Real`
2	¿Qué estructura elige entre varias opciones de un número?	
2	¿Qué ciclo conviene si se sabe cuántas vueltas dar?	
3	¿Qué tiene una `Funcion` que no tiene un `SubProceso`?	
3	¿Qué hace `Por Referencia`?	

LO QUE SIGUE

En tres días aprendió a analizar, decidir, repetir, guardar listas y dividir el trabajo en módulos.

Hoy lo usa **todo a la vez**, como en un programa real.

¿Lo recuerdan?

Una pregunta por cada idea que hoy va a necesitar

Ses.	Pregunta	Respuesta
1	¿Qué significan las letras EPS?	Entrada, Proceso, Salida
1	¿Cómo se declara una variable real `suma`?	`Definir suma Como Real`
2	¿Qué estructura elige entre varias opciones de un número?	`Segun … FinSegun`
2	¿Qué ciclo conviene si se sabe cuántas vueltas dar?	
3	¿Qué tiene una `Funcion` que no tiene un `SubProceso`?	
3	¿Qué hace `Por Referencia`?	

LO QUE SIGUE

En tres días aprendió a analizar, decidir, repetir, guardar listas y dividir el trabajo en módulos.

Hoy lo usa **todo a la vez**, como en un programa real.

¿Lo recuerdan?

Una pregunta por cada idea que hoy va a necesitar

Ses.	Pregunta	Respuesta
1	¿Qué significan las letras EPS?	Entrada, Proceso, Salida
1	¿Cómo se declara una variable real `suma`?	`Definir suma Como Real`
2	¿Qué estructura elige entre varias opciones de un número?	`Segun … FinSegun`
2	¿Qué ciclo conviene si se sabe cuántas vueltas dar?	`Para … FinPara`
3	¿Qué tiene una `Funcion` que no tiene un `SubProceso`?	
3	¿Qué hace `Por Referencia`?	

LO QUE SIGUE

En tres días aprendió a analizar, decidir, repetir, guardar listas y dividir el trabajo en módulos.

Hoy lo usa **todo a la vez**, como en un programa real.

¿Lo recuerdan?

Una pregunta por cada idea que hoy va a necesitar

Ses.	Pregunta	Respuesta
1	¿Qué significan las letras EPS?	Entrada, Proceso, Salida
1	¿Cómo se declara una variable real `suma`?	`Definir suma Como Real`
2	¿Qué estructura elige entre varias opciones de un número?	`Segun … FinSegun`
2	¿Qué ciclo conviene si se sabe cuántas vueltas dar?	`Para … FinPara`
3	¿Qué tiene una `Funcion` que no tiene un `SubProceso`?	Devuelve un valor
3	¿Qué hace `Por Referencia`?	

LO QUE SIGUE

En tres días aprendió a analizar, decidir, repetir, guardar listas y dividir el trabajo en módulos.

Hoy lo usa **todo a la vez**, como en un programa real.

¿Lo recuerdan?

Una pregunta por cada idea que hoy va a necesitar

Ses.	Pregunta	Respuesta
1	¿Qué significan las letras EPS?	Entrada, Proceso, Salida
1	¿Cómo se declara una variable real `suma`?	`Definir suma Como Real`
2	¿Qué estructura elige entre varias opciones de un número?	`Segun … FinSegun`
2	¿Qué ciclo conviene si se sabe cuántas vueltas dar?	`Para … FinPara`
3	¿Qué tiene una `Funcion` que no tiene un `SubProceso`?	Devuelve un valor
3	¿Qué hace `Por Referencia`?	El módulo cambia la variable original

LO QUE SIGUE

En tres días aprendió a analizar, decidir, repetir, guardar listas y dividir el trabajo en módulos.

Hoy lo usa **todo a la vez**, como en un programa real.

Todo lo que ya saben, en un solo programa

Cada sesión aporta una pieza del proyecto

Sesión	Lo que aprendió	Dónde aparece hoy
1	Análisis EPS, constantes, variables, expresiones y operadores lógicos	La tabla de la etapa 0; `CANTIDAD` y `NOTA_MINIMA`; las validaciones con `O`, `Y` y `NO`
2	`Si`, `Segun`, los tres ciclos, contador, acumulador y arreglos	El menú (`Repetir` + `Segun`), las categorías, la suma del promedio, el conteo de aprobados y `notas[5]`
3	Diseño descendente, `SubProceso`, `Funcion` y parámetros	Nueve módulos; `MostrarMenu` sin parámetros; `CargarNotas` recibe `hayDatos Por Referencia`; `SumarPuntos` cambia el arreglo
4	Integrar, probar y documentar	Un plan de siete pruebas y la entrega en GitHub

RECOMENDACIÓN

Tenga abiertos sus archivos `P02_*.psc` y `P03_*.psc`: varias piezas de hoy ya las escribió. El menú de tres notas del manual se amplía a cinco estudiantes, con validación, estadísticas y puntos extra.

02 PARTE
ALGORITMOS

Proyecto integrador

El reto

Un sistema de notas para cinco estudiantes

Situación. Un catedrático necesita un programa para manejar las notas finales (de 0 a 100) de sus cinco estudiantes. Se aprueba con 61 o más.

EL PROGRAMA DEBE

1. **Ingresar** las cinco notas. Si una está fuera de 0 a 100, se vuelve a pedir.
2. **Mostrar** cada nota con su categoría.
3. Calcular **estadísticas**: promedio, nota más alta y cuántos aprobaron.
4. Sumar **puntos extra** a todos, sin pasar de 100.
5. **Salir** solo cuando el usuario lo pida.

Así se verá en PSeInt

```
===== 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
```

NOTA

Si elige 2, 3 o 4 antes de ingresar notas, el programa avisa: `Todavia no hay notas.`

Práctica final: instrucciones

Léalas antes de empezar

2 h 30 min en PSeInt

CÓMO SE TRABAJA

1. **Individual** y siempre en **PSeInt** (perfil Flexible).
2. Abra la plantilla y guárdela como `P04_SistemaNotas.psc` en `Practica_04`.
3. Ponga su **nombre** y la **fecha** en los comentarios.
4. Cada etapa termina en un **punto de control**: levante la mano y el instructor la revisa.
5. Guarde (`Ctrl+S`) al terminar cada etapa.

NOTA

Se revisa: que pase los siete casos de prueba, los módulos, la validación, el orden y los comentarios.

Etapa	Qué construye	Tiempo
0	Análisis EPS (cuaderno)	20 min
1	El menú que se repite	35 min
2	Cargar, validar y mostrar	35 min
–	*Receso*	15 min
3	Estadísticas	40 min
4	Puntos extra y pruebas	25 min
5	Toques finales	15 min

REGLAS

- Puede consultar a su vecino, pero no se copian archivos.
- **No avance** si la etapa anterior no funciona.
- ¿Terminó antes? Pase a los **retos extra**.

Lo que su programa debe usar

Todo el curso, en un solo archivo

Concepto	Sesión	Dónde aparece en su programa
Constantes; tipos `Entero`, `Real`, `Logico`, `Cadena`	1	`CANTIDAD`, `NOTA_MINIMA`, `hayDatos`, la categoría
Operadores relacionales y lógicos `O`, `Y`, `NO`	1	Validar las notas (etapa 2) y los puntos (etapa 5)
`Si` anidado y `Segun`	2	La función `Categoria` y el menú
Los tres ciclos: `Repetir`, `Mientras`, `Para`	2	El menú, las validaciones y los recorridos
Contador, acumulador y búsqueda del mayor	2	`ContarAprobados`, `Promedio`, `NotaMayor`
Arreglo	2	`notas[5]`
Cinco `Funcion` que devuelven un valor	3	`LeerNotaValida`, `Categoria` y las tres de la etapa 3
Cuatro `SubProceso`, uno sin parámetros	3	`CargarNotas`, `MostrarNotas`, `SumarPuntos`, `MostrarMenu`
Por valor y `Por Referencia`	3	`hayDatos Por Referencia`; el arreglo, siempre
Análisis, diseño descendente y pruebas	4	La etapa 0 y los siete casos de la etapa 4

NOTA

`DIV`, `MOD` y el centinela no encajan en este problema: están en el Socrative y en los retos extra.

Etapa 0. Análisis EPS

En el cuaderno, antes de tocar el teclado

20 min

Parte	En este problema
Entrada	
Proceso	
Salida	
Constantes	
Variables	

PIENSE

¿Para qué sirve `hayDatos`? ¿Qué pasaría si el usuario pide el promedio antes de ingresar notas?

Etapa 0. Análisis EPS

En el cuaderno, antes de tocar el teclado

20 min

Parte	En este problema
Entrada	Las 5 notas, la opción del menú y los puntos extra
Proceso	
Salida	
Constantes	
Variables	

PIENSE

¿Para qué sirve `hayDatos`? ¿Qué pasaría si el usuario pide el promedio antes de ingresar notas?

Etapa 0. Análisis EPS

En el cuaderno, antes de tocar el teclado

20 min

Parte	En este problema
Entrada	Las 5 notas, la opción del menú y los puntos extra
Proceso	Validar cada nota, clasificarla, sumar, buscar la mayor, contar aprobados y sumar puntos
Salida	
Constantes	
Variables	

PIENSE

¿Para qué sirve `hayDatos`? ¿Qué pasaría si el usuario pide el promedio antes de ingresar notas?

Etapa 0. Análisis EPS

En el cuaderno, antes de tocar el teclado

20 min

Parte	En este problema
Entrada	Las 5 notas, la opción del menú y los puntos extra
Proceso	Validar cada nota, clasificarla, sumar, buscar la mayor, contar aprobados y sumar puntos
Salida	El menú, la lista con categorías, las estadísticas y los avisos
Constantes	
Variables	

PIENSE

¿Para qué sirve `hayDatos`? ¿Qué pasaría si el usuario pide el promedio antes de ingresar notas?

Etapa 0. Análisis EPS

En el cuaderno, antes de tocar el teclado

20 min

Parte	En este problema
Entrada	Las 5 notas, la opción del menú y los puntos extra
Proceso	Validar cada nota, clasificarla, sumar, buscar la mayor, contar aprobados y sumar puntos
Salida	El menú, la lista con categorías, las estadísticas y los avisos
Constantes	`CANTIDAD = 5`, `NOTA_MINIMA = 61`
Variables	`notas[5]`, `opcion`, `puntos`, `hayDatos`

PIENSE

¿Para qué sirve `hayDatos`? ¿Qué pasaría si el usuario pide el promedio antes de ingresar notas?

Etapa 0. Análisis EPS

En el cuaderno, antes de tocar el teclado

20 min

Parte	En este problema
Entrada	Las 5 notas, la opción del menú y los puntos extra
Proceso	Validar cada nota, clasificarla, sumar, buscar la mayor, contar aprobados y sumar puntos
Salida	El menú, la lista con categorías, las estadísticas y los avisos
Constantes	`CANTIDAD = 5`, `NOTA_MINIMA = 61`
Variables	`notas[5]`, `opcion`, `puntos`, `hayDatos`

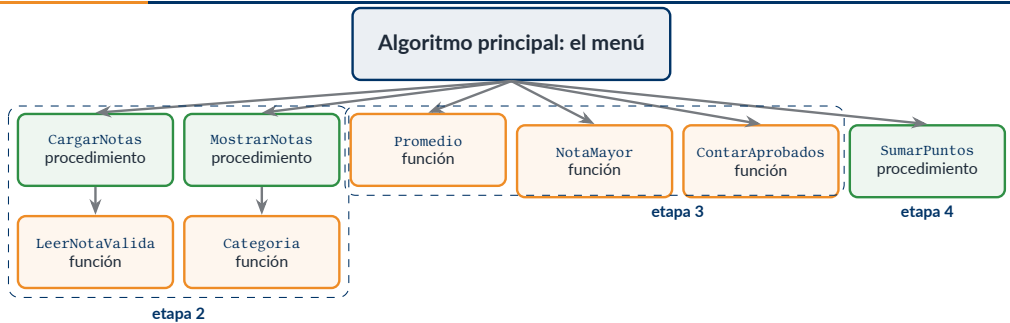
PIENSE

¿Para qué sirve `hayDatos`? ¿Qué pasaría si el usuario pide el promedio antes de ingresar notas?

`hayDatos` es de tipo `Logico`: empieza en `Falso` y pasa a `Verdadero` al cargar las notas. Así el programa no calcula con un arreglo vacío.

Etapa 0. Diseño descendente

Ocho módulos, cada uno hace una sola cosa; el noveno, el menú, llega en la etapa 5



NOTA

Verde: procedimientos. **Naranja:** funciones. Un módulo puede llamar a otro.

RECOMENDACIÓN

Punto de control 0: muestre su tabla EPS y su esquema.

Etapa 1. El menú que se repite

Abra la plantilla o cópielo

35 min

```
Algoritmo SistemaDeNotas
  Definir CANTIDAD, NOTA_MINIMA, opcion Como Entero
  Definir puntos Como Real
  Definir hayDatos Como Logico
  Definir notas Como Real
  Dimension notas[5]

  CANTIDAD <- 5
  NOTA_MINIMA <- 61
  hayDatos <- Falso

  Repetir
    Escribir " "
    Escribir "===== SISTEMA DE NOTAS FE729 ====="
    Escribir "1. Ingresar notas"
    Escribir "2. Mostrar notas y categorias"
    Escribir "3. Estadisticas"
    Escribir "4. Sumar puntos extra"
    Escribir "5. Salir"
    Escribir "Opcion:"
    Leer opcion

    Segun opcion Hacer
      1:
        Escribir "(pendiente: etapa 2)"
      2:
        Escribir "(pendiente: etapa 2)"
      3:
        Escribir "(pendiente: etapa 3)"
      4:
        Escribir "(pendiente: etapa 4)"
      5:
        Escribir "Gracias, hasta pronto."
      De Otro Modo:
        Escribir "Opcion invalida"
    FinSegun
  Hasta Que opcion = 5
FinAlgoritmo
```

NOTA

Sesión 1: constantes y variables. **Sesión 2:** `Repetir` + `Segun` y el arreglo. Los módulos se escribirán **arriba** de la línea 1.

Etapa 1. Pruebe el esqueleto

Antes de agregar módulos, el menú debe funcionar solo

Escriba	Debe ver
2	
9	
5	

PIENSE

¿Por qué aquí conviene `Repetir` y no `Mientras`?

RECOMENDACIÓN

Punto de control 1: el menú responde a 2, 9 y 5. Guarde el archivo.

Etapa 1. Pruebe el esqueleto

Antes de agregar módulos, el menú debe funcionar solo

Escriba	Debe ver
2	`(pendiente: etapa 2)` y el menú otra vez
9	
5	

PIENSE

¿Por qué aquí conviene `Repetir` y no `Mientras`?

RECOMENDACIÓN

Punto de control 1: el menú responde a 2, 9 y 5. Guarde el archivo.

Etapa 1. Pruebe el esqueleto

Antes de agregar módulos, el menú debe funcionar solo

Escriba	Debe ver
2	`(pendiente: etapa 2)` y el menú otra vez
9	`Opcion invalida` y el menú otra vez
5	

PIENSE

¿Por qué aquí conviene `Repetir` y no `Mientras`?

RECOMENDACIÓN

Punto de control 1: el menú responde a 2, 9 y 5. Guarde el archivo.

Etapa 1. Pruebe el esqueleto

Antes de agregar módulos, el menú debe funcionar solo

Escriba	Debe ver
2	`(pendiente: etapa 2)` y el menú otra vez
9	`Opcion invalida` y el menú otra vez
5	`Gracias, hasta pronto.` y termina

El `Repetir` vuelve a mostrar el menú hasta que `opcion` vale 5. El `De Otro Modo` atrapa cualquier número que no esté en la lista.

PIENSE

¿Por qué aquí conviene `Repetir` y no `Mientras`?

RECOMENDACIÓN

Punto de control 1: el menú responde a 2, 9 y 5. Guarde el archivo.

Etapa 1. Pruebe el esqueleto

Antes de agregar módulos, el menú debe funcionar solo

Escriba	Debe ver
2	`(pendiente: etapa 2)` y el menú otra vez
9	`Opcion invalida` y el menú otra vez
5	`Gracias, hasta pronto.` y termina

El `Repetir` vuelve a mostrar el menú hasta que `opcion` vale 5. El `De Otro Modo` atrapa cualquier número que no esté en la lista.

PIENSE

¿Por qué aquí conviene `Repetir` y no `Mientras`?

Porque el menú debe mostrarse **al menos una vez**, antes de saber qué opción elige el usuario.

RECOMENDACIÓN

Punto de control 1: el menú responde a 2, 9 y 5. Guarde el archivo.

Etapa 2. Cargar y validar las notas

Los módulos van arriba de `Algoritmo`

35 min

```
1  Funcion nota <- LeerNotaValida(numero)
2    Definir nota Como Real
3    Escribir "Nota del estudiante ", numero, " (0 a 100):"
4    Leer nota
5    Mientras nota < 0 O nota > 100 Hacer
6      Escribir "Nota fuera de rango. Escribala de nuevo:"
7      Leer nota
8    FinMientras
9  FinFuncion
10
11 SubProceso CargarNotas(notas, n, hayDatos Por Referencia)
12   Definir i Como Entero
13   Para i <- 1 Hasta n Con Paso 1 Hacer
14     notas[i] <- LeerNotaValida(i)
15   FinPara
16   hayDatos <- Verdadero
17 FinSubProceso
```

FÍJESE EN

- Línea 5: una expresión lógica con `O` (sesión 1) dentro de un `Mientras` (sesión 2).
- Línea 14: un procedimiento que **llama a una función** y guarda su valor en el arreglo.
- Línea 11: `hayDatos` va `Por Referencia` porque el módulo debe cambiarla en el principal.

ADVERTENCIA

Con `Y` en lugar de `O`, la condición nunca se cumple: una nota no puede ser menor que 0 y mayor que 100 a la vez.

Etapa 2. Clasificar y mostrar

¿Le suena? Es la parte B de la sesión 2, ahora como función

```
Funcion cat <- Categoria(nota)
  Definir cat Como Cadena
  Si nota >= 90 Entonces
    cat <- "Sobresaliente"
  SiNo
    Si nota >= 76 Entonces
      cat <- "Notable"
    SiNo
      Si nota >= 61 Entonces
        cat <- "Satisfactorio"
      SiNo
        cat <- "Insuficiente"
      FinSi
    FinSi
  FinSi
FinFuncion

SubProceso MostrarNotas(notas, n)
  Definir i Como Entero
  Para i <- 1 Hasta n Con Paso 1 Hacer
    Escribir i, ". ", notas[i], " - ", Categoria(notas[i])
  FinPara
FinSubProceso
```

Conecte las opciones 1 y 2 del menú

```
1:
  CargarNotas(notas, CANTIDAD, hayDatos)
2:
  Si hayDatos Entonces
    MostrarNotas(notas, CANTIDAD)
  SiNo
    Escribir "Todavia no hay notas."
  FinSi
```

NOTA

Borre los `Escribir "(pendiente…)"` de las opciones 1 y 2 y ponga estas líneas en su lugar.

RECOMENDACIÓN

`Categoria` devuelve un texto: por eso su resultado es `Cadena`, no `Real`.

Etapa 2. Pruebe con los datos de la sesión 2

Notas: 80, 55, 90, 70 y 45

Prueba de validación (opción 1)

```
Nota del estudiante 1 (0 a 100):
> 105
Nota fuera de rango. Escribala de nuevo:
> -5
Nota fuera de rango. Escribala de nuevo:
> 80
Nota del estudiante 2 (0 a 100):
> 55   …
```

NOTA

Primero pruebe la opción 2 **sin** cargar notas: debe decir `Todavia no hay notas.`

Opción 2 después de cargar

RECOMENDACIÓN

Punto de control 2: validación, lista y categorías correctas. Guarde.

Etapa 2. Pruebe con los datos de la sesión 2

Notas: 80, 55, 90, 70 y 45

Prueba de validación (opción 1)

```
Nota del estudiante 1 (0 a 100):
> 105
Nota fuera de rango. Escribala de nuevo:
> -5
Nota fuera de rango. Escribala de nuevo:
> 80
Nota del estudiante 2 (0 a 100):
> 55   …
```

NOTA

Primero pruebe la opción 2 **sin** cargar notas: debe decir `Todavia no hay notas.`

Opción 2 después de cargar

```
1. 80 - Notable
2. 55 - Insuficiente
3. 90 - Sobresaliente
4. 70 - Satisfactorio
5. 45 - Insuficiente
```

RECOMENDACIÓN

Punto de control 2: validación, lista y categorías correctas. Guarde.

Etapa 2. Pruebe con los datos de la sesión 2

Notas: 80, 55, 90, 70 y 45

Prueba de validación (opción 1)

```
Nota del estudiante 1 (0 a 100):
> 105
Nota fuera de rango. Escribala de nuevo:
> -5
Nota fuera de rango. Escribala de nuevo:
> 80
Nota del estudiante 2 (0 a 100):
> 55   …
```

NOTA

Primero pruebe la opción 2 **sin** cargar notas: debe decir `Todavia no hay notas.`

Opción 2 después de cargar

```
1. 80 - Notable
2. 55 - Insuficiente
3. 90 - Sobresaliente
4. 70 - Satisfactorio
5. 45 - Insuficiente
```

SI SIEMPRE DICE "TODAVÍA NO HAY NOTAS"

Revise que `CargarNotas` tenga `hayDatos Por Referencia`. Sin esa palabra, el cambio a `Verdadero` se queda en la copia (sesión 3).

RECOMENDACIÓN

Punto de control 2: validación, lista y categorías correctas. Guarde.

Etapa 3. Estadísticas con tres funciones

Contador y acumulador de la sesión 2, ahora dentro de funciones

40 min

```
Funcion prom <- Promedio(notas, n)
  Definir prom, suma Como Real
  Definir i Como Entero
  suma <- 0
  Para i <- 1 Hasta n Con Paso 1 Hacer
    suma <- suma + notas[i]
  FinPara
  prom <- suma / n
FinFuncion

Funcion mayor <- NotaMayor(notas, n)
  Definir mayor Como Real
  Definir i Como Entero
  mayor <- notas[1]
  Para i <- 2 Hasta n Con Paso 1 Hacer
    Si notas[i] > mayor Entonces
      mayor <- notas[i]
    FinSi
  FinPara
FinFuncion
```

```
Funcion cuantos <- ContarAprobados(notas, n, minima)
  Definir cuantos, i Como Entero
  cuantos <- 0
  Para i <- 1 Hasta n Con Paso 1 Hacer
    Si notas[i] >= minima Entonces
      cuantos <- cuantos + 1
    FinSi
  FinPara
FinFuncion
```

FÍJESE EN

- `suma` es un **acumulador** y `cuantos` un **contador**.
- `NotaMayor` empieza con la primera nota y compara desde la segunda.
- Las tres reciben el arreglo y `n`; ninguna escribe en pantalla: solo calculan.

Etapa 3. La opción 3 y su seguimiento

Una función se usa dentro de un `Escribir`

```
    3:
      Si hayDatos Entonces
        Escribir "Promedio: ", Promedio(notas, CANTIDAD)
        Escribir "Nota mas alta: ", NotaMayor(notas, CANTIDAD)
        Escribir "Aprobados: ",
          ContarAprobados(notas, CANTIDAD, NOTA_MINIMA)
      SiNo
        Escribir "Todavia no hay notas."
      FinSi
```

El `Escribir "Aprobados: "` y su segunda línea van en **una sola línea** en PSeInt; aquí se parten solo para que quepan.

Seguimiento de `NotaMayor`

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

NOTA

Promedio: $340/5 = 68$. Aprobados: 80, 90 y 70, es decir, **3**.

Etapa 3. La opción 3 y su seguimiento

Una función se usa dentro de un `Escribir`

```
    3:
      Si hayDatos Entonces
        Escribir "Promedio: ", Promedio(notas, CANTIDAD)
        Escribir "Nota mas alta: ", NotaMayor(notas, CANTIDAD)
        Escribir "Aprobados: ",
          ContarAprobados(notas, CANTIDAD, NOTA_MINIMA)
      SiNo
        Escribir "Todavia no hay notas."
      FinSi
```

El `Escribir "Aprobados: "` y su segunda línea van en **una sola línea** en PSeInt; aquí se parten solo para que quepan.

Seguimiento de `NotaMayor`

i	notas[i]	¿Mayor?	mayor
–	–	–	80
2	55	No	80
3	90		
4	70		
5	45		

NOTA

Promedio: $340/5 = 68$. Aprobados: 80, 90 y 70, es decir, **3**.

Etapa 3. La opción 3 y su seguimiento

Una función se usa dentro de un `Escribir`

```
    3:
      Si hayDatos Entonces
        Escribir "Promedio: ", Promedio(notas, CANTIDAD)
        Escribir "Nota mas alta: ", NotaMayor(notas, CANTIDAD)
        Escribir "Aprobados: ",
          ContarAprobados(notas, CANTIDAD, NOTA_MINIMA)
      SiNo
        Escribir "Todavia no hay notas."
      FinSi
```

El `Escribir "Aprobados: "` y su segunda línea van en **una sola línea** en PSeInt; aquí se parten solo para que quepan.

Seguimiento de `NotaMayor`

i	notas[i]	¿Mayor?	mayor
–	–	–	80
2	55	No	80
3	90	Sí	90
4	70		
5	45		

NOTA

Promedio: $340/5 = 68$. Aprobados: 80, 90 y 70, es decir, **3**.

Etapa 3. La opción 3 y su seguimiento

Una función se usa dentro de un `Escribir`

```
    3:
      Si hayDatos Entonces
        Escribir "Promedio: ", Promedio(notas, CANTIDAD)
        Escribir "Nota mas alta: ", NotaMayor(notas, CANTIDAD)
        Escribir "Aprobados: ",
          ContarAprobados(notas, CANTIDAD, NOTA_MINIMA)
      SiNo
        Escribir "Todavia no hay notas."
      FinSi
```

El `Escribir "Aprobados: "` y su segunda línea van en **una sola línea** en PSeInt; aquí se parten solo para que quepan.

Seguimiento de `NotaMayor`

i	notas[i]	¿Mayor?	mayor
–	–	–	80
2	55	No	80
3	90	Sí	90
4	70	No	90
5	45	No	**90**

NOTA

Promedio: $340/5 = 68$. Aprobados: 80, 90 y 70, es decir, **3**.

Etapa 3. La opción 3 y su seguimiento

Una función se usa dentro de un `Escribir`

```
    3:
      Si hayDatos Entonces
        Escribir "Promedio: ", Promedio(notas, CANTIDAD)
        Escribir "Nota mas alta: ", NotaMayor(notas, CANTIDAD)
        Escribir "Aprobados: ",
          ContarAprobados(notas, CANTIDAD, NOTA_MINIMA)
      SiNo
        Escribir "Todavia no hay notas."
      FinSi
```

El `Escribir "Aprobados: "` y su segunda línea van en **una sola línea** en PSeInt; aquí se parten solo para que quepan.

```
Promedio: 68
Nota mas alta: 90
Aprobados: 3
```

RECOMENDACIÓN

Punto de control 3: 68, 90 y 3 con las notas de la etapa 2.

Seguimiento de `NotaMayor`

i	notas[i]	¿Mayor?	mayor
–	–	–	80
2	55	No	80
3	90	Sí	90
4	70	No	90
5	45	No	**90**

NOTA

Promedio: $340/5 = 68$. Aprobados: 80, 90 y 70, es decir, **3**.

Etapa 4. Puntos extra

Un procedimiento que cambia el arreglo

25 min

```
1  SubProceso SumarPuntos(notas, n, puntos)
2    Definir i Como Entero
3    Para i <- 1 Hasta n Con Paso 1 Hacer
4      notas[i] <- notas[i] + puntos
5      Si notas[i] > 100 Entonces
6        notas[i] <- 100
7      FinSi
8    FinPara
9  FinSubProceso
```

Opción 4 del menú

```
   4:
     Si hayDatos Entonces
       Escribir "Puntos a sumar:"
       Leer puntos
       SumarPuntos(notas, CANTIDAD, puntos)
       Escribir "Puntos aplicados."
     SiNo
       Escribir "Todavia no hay notas."
     FinSi
```

PIENSE

`SumarPuntos` no dice `Por Referencia`. ¿Cambiarán las notas del principal?

NOTA

Las líneas 5 a 7 ponen un **tope**: una nota de 95 más 15 puntos queda en 100, no en 110.

Etapa 4. Puntos extra

Un procedimiento que cambia el arreglo

25 min

```
1  SubProceso SumarPuntos(notas, n, puntos)
2    Definir i Como Entero
3    Para i <- 1 Hasta n Con Paso 1 Hacer
4      notas[i] <- notas[i] + puntos
5      Si notas[i] > 100 Entonces
6        notas[i] <- 100
7      FinSi
8    FinPara
9  FinSubProceso
```

Opción 4 del menú

```
    4:
      Si hayDatos Entonces
        Escribir "Puntos a sumar:"
        Leer puntos
        SumarPuntos(notas, CANTIDAD, puntos)
        Escribir "Puntos aplicados."
      SiNo
        Escribir "Todavia no hay notas."
      FinSi
```

PIENSE

`SumarPuntos` no dice `Por Referencia`. ¿Cambiarán las notas del principal?

Sí. En PSeInt los **arreglos** siempre se pasan por referencia: el módulo trabaja sobre el arreglo original. En cambio, `puntos` y `n` son simples y llegan por valor.

NOTA

Las líneas 5 a 7 ponen un **tope**: una nota de 95 más 15 puntos queda en 100, no en 110.

Etapa 4. Plan de pruebas

Ejecute los siete casos en orden y anote si funcionó

No.	Caso	Qué escribir	Resultado esperado
1	Sin datos	Opción `2` al iniciar	`Todavia no hay notas.`
2	Opción inválida	`9`	`Opcion invalida`
3	Validación	Opción `1`; luego `105`, `-5` y `80`	Pide de nuevo dos veces y acepta el 80
4	Carga normal	`80`, `55`, `90`, `70`, `45`; opción `2`	Cinco líneas con su categoría
5	Estadísticas	Opción `3`	
6	Puntos extra	Opción `4` con `15`; luego `2` y `3`	
7	Salir	`5`	`Gracias, hasta pronto.`

RECOMENDACIÓN

Punto de control 4: los siete casos dan lo esperado. Guarde y respalde en su USB o en la nube.

Etapa 4. Plan de pruebas

Ejecute los siete casos en orden y anote si funcionó

No.	Caso	Qué escribir	Resultado esperado
1	Sin datos	Opción `2` al iniciar	`Todavia no hay notas.`
2	Opción inválida	`9`	`Opcion invalida`
3	Validación	Opción `1`; luego `105`, `-5` y `80`	Pide de nuevo dos veces y acepta el 80
4	Carga normal	`80`, `55`, `90`, `70`, `45`; opción `2`	Cinco líneas con su categoría
5	Estadísticas	Opción `3`	Promedio 68, más alta 90, aprobados 3
6	Puntos extra	Opción `4` con `15`; luego `2` y `3`	
7	Salir	`5`	`Gracias, hasta pronto.`

RECOMENDACIÓN

Punto de control 4: los siete casos dan lo esperado. Guarde y respalde en su USB o en la nube.

Etapa 4. Plan de pruebas

Ejecute los siete casos en orden y anote si funcionó

No.	Caso	Qué escribir	Resultado esperado
1	Sin datos	Opción `2` al iniciar	`Todavia no hay notas.`
2	Opción inválida	`9`	`Opcion invalida`
3	Validación	Opción `1`; luego `105`, `-5` y `80`	Pide de nuevo dos veces y acepta el 80
4	Carga normal	`80`, `55`, `90`, `70`, `45`; opción `2`	Cinco líneas con su categoría
5	Estadísticas	Opción `3`	Promedio 68, más alta 90, aprobados 3
6	Puntos extra	Opción `4` con `15`; luego `2` y `3`	Notas 95, 70, 100, 85 y 60; promedio 82, más alta 100, aprobados 4
7	Salir	`5`	`Gracias, hasta pronto.`

En el caso 6, $90 + 15 = 105$ queda en **100** por el tope, y $45 + 15 = 60$ **sigue reprobado**: le falta 1 punto.

RECOMENDACIÓN

Punto de control 4: los siete casos dan lo esperado. Guarde y respalde en su USB o en la nube.

Etapa 5. Toques finales

Un procedimiento sin parámetros y una validación con Y y NO

15 min

El noveno módulo: el menú (arriba de Algoritmo)

```
SubProceso MostrarMenu
  Escribir " "
  Escribir "===== SISTEMA DE NOTAS FE729 ====="
  Escribir "1. Ingresar notas"
  Escribir "2. Mostrar notas y categorias"
  Escribir "3. Estadisticas"
  Escribir "4. Sumar puntos extra"
  Escribir "5. Salir"
  Escribir "Opcion:"
FinSubProceso
```

En el principal, el Repetir queda así

```
Repetir
  MostrarMenu
  Leer opcion
  Segun opcion Hacer ...
```

Opción 4: los puntos solo de 0 a 20

```
Escribir "Puntos a sumar (0 a 20):"
Leer puntos
Mientras NO (puntos >= 0 Y puntos <= 20) Hacer
  Escribir "Solo de 0 a 20. Escribalos de nuevo:"
  Leer puntos
FinMientras
```

FÍJESE EN

- Sin parámetros, `MostrarMenu` se escribe y se llama **sin paréntesis** (sesión 3).
- `NO (puntos >= 0 Y puntos <= 20)` equivale a `puntos < 0 O puntos > 20` (sesión 1).

RECOMENDACIÓN

Punto de control 5: el menú se ve igual; con `25` pide de nuevo y con `15` acepta.

Si algo no sale

Errores típicos al integrar un programa

SÍNTOMA → CAUSA PROBABLE

- **Siempre "Todavia no hay notas"** → falta `Por Referencia` en `hayDatos`.
- **Acepta la nota 105** → se usó `Y` en lugar de `O`, o falta el `Leer` dentro del `Mientras`.
- **El promedio sale 0 o raro** → `suma` no empieza en 0, o se divide dentro del ciclo.
- **"Variable no definida"** → falta un `Definir` dentro del módulo (cada uno tiene los suyos).
- **No pasa nada al elegir 3** → se llamó la función sin `Escribir`: el valor se perdió.
- **Una nota queda en 110** → falta el tope en `SumarPuntos`.

RECOMENDACIÓN

Use la **ejecución paso a paso** y observe cómo el programa entra a cada módulo y regresa al menú.

NOTA

Pruebe cada módulo en cuanto lo escribe. Si algo falla, el error casi siempre está en lo último que agregó.

CIBERSEGURIDAD

La validación de la etapa 2 es su primera defensa: **nunca confíe** en lo que escribe el usuario.

Retos extra

Para quien termine antes: elija uno o varios

NIVEL 1

- Agregue la función `NotaMenor` y muéstrela en la opción 3.
- Antes de salir, pregunte `¿Seguro? (S/N)`.
- Muestre el promedio entero en decenas y unidades con `DIV` y `MOD`: 68 son 6 decenas y 8 unidades.

NIVEL 2

- Opción 6: contar cuántos estudiantes hay en cada categoría.
- Guarde también los **nombres** en un segundo arreglo `nombres[5]` de tipo `Cadena`.
- Opción 7: buscar un estudiante por su número (1 a 5) y mostrar solo su nota.
- Cargue notas hasta que escriban `-1` (**centinela**), con un máximo de cinco.

NOTA

Cada reto es un módulo nuevo más una línea en el menú. Esa es la ventaja del diseño descendente: el programa crece sin desordenarse.

Lista de verificación y entrega

Lo que el instructor revisa en su proyecto

SU PROGRAMA

- El menú se repite hasta elegir 5 y rechaza opciones inválidas.
- Las notas fuera de 0 a 100 se vuelven a pedir.
- Las opciones 2, 3 y 4 avisan si no hay notas.
- Los puntos extra fuera de 0 a 20 se vuelven a pedir.
- Cada tarea está en su módulo: 4 procedimientos y 5 funciones.
- Los siete casos de prueba dan lo esperado.
- El archivo tiene su nombre y la fecha en los comentarios.

ENTREGA

1. Carpeta `Practica_04` con `P04_SistemaNotas.psc`.
2. Un `README` de tres líneas: qué hace el programa y cómo se usa.
3. Súbala a su repositorio `Algoritmos-FE729`.

RECOMENDACIÓN

Con esta carpeta su repositorio queda completo: cuatro prácticas, la evidencia de su primer curso.

¡Práctica final entregada!

Acaba de escribir un programa completo, de principio a fin

9 módulos

3 ciclos distintos

1 arreglo

7 pruebas superadas

CONFIRME SU ENTREGA

- ✔ `P04_SistemaNotas.psc` en `Practica_04`.
- ✔ `README` de tres líneas.
- ✔ Subido a `Algoritmos-FE729` en GitHub.
- ✔ El instructor vio los siete casos.

PIÉNSELO

- ¿Qué etapa le costó más y por qué?
- ¿Qué módulo reutilizaría en otro programa?
- ¿Qué error encontró gracias a las pruebas?

Analizó, diseñó en módulos, construyó por etapas y probó: es el mismo camino que sigue un programador profesional.

NOTA

Guarde todo y cierre PSeInt: sigue la evaluación en Socrative.

Evaluación en Socrative

25 min Individual • 40 puntos

EVALUACIÓN EN LÍNEA (SOCRATIVE)

1. Abra `socrative.com` en el navegador o en la aplicación.
2. Elija **Iniciar sesión** y luego **Estudiante**.
3. Escriba el nombre de la sala.
4. Ingrese su **nombre completo**, tal como aparece en la lista.

NOMBRE DE LA SALA



NOTA

Es la evaluación de cierre: las preguntas recorren las cuatro sesiones y el proyecto que usted acaba de construir.

03 PARTE
ALGORITMOS

Cierre del curso

Lo que aprendieron en cuatro sesiones

De no saber qué es un algoritmo a un programa completo con menú y módulos



Del problema al algoritmo
EPS, pseudocódigo, constantes, variables y expresiones

Decisiones y repeticiones
`Si`, `Segun`, tres ciclos, contador, acumulador y arreglos

Dividir el trabajo
Diseño descendente, `SubProceso`, `Funcion` y parámetros

Proyecto integrador
Todo junto: analizar, construir por etapas, probar y entregar

Lo más importante no es la sintaxis de PSeInt: es la forma de **pensar** un problema, partirlo en pasos y comprobar que la solución funciona.

EN VOZ ALTA

¿Qué fue lo más difícil del curso? ¿Qué parte del proyecto de hoy le dio más satisfacción?

¿Qué sigue?

Lo de este curso reaparece durante toda la carrera

EN LA CARRERA

- Los mismos algoritmos se escribirán en lenguajes como **C++** o **Python**.
- Los arreglos crecerán hasta ser estructuras de datos y bases de datos.
- En **IA** y **ciberseguridad** todo empieza por un algoritmo bien pensado.

SU PORTAFOLIO

- Mantenga vivo su repositorio: suba cada práctica de los próximos cursos.
- Escriba un `README` claro en cada proyecto.
- Un portafolio ordenado habla por usted al buscar prácticas o empleo.

CONSEJOS FINALES

- Programe un poco **cada día**; se aprende haciendo.
- Lea los mensajes de error con calma: casi siempre dicen dónde está el problema.
- Pregunte y trabaje en equipo: nadie programa solo.

Esta tarde, en **Electrónica Digital**, pasará lo mismo con el hardware: un proyecto que junta todo lo visto en un solo circuito.

UNIVERSIDAD RURAL DE GUATEMALA

¡Gracias!

Fue un gusto acompañarlos en su primer curso de la carrera.
¡Mucho éxito en lo que viene!

github.com/MynorCifuentes

