



aprenderaprogramar.com

Ejercicios ejemplo: verificación de algoritmos del valor suma de una sucesión matemática. (CU00238A)

Sección: Cursos

Categoría: Curso Bases de la programación Nivel II

Fecha revisión: 2024

Autor: Mario R. Rancel

Resumen: Entrega nº 37 del Curso Bases de la programación Nivel II

24

EJERCICIO 1 VERIFICACIÓN DEL ALGORITMO DE CÁLCULO DEL VALOR DE LA SUMA DE UNA SUCESIÓN

Se desea realizar una verificación funcional basada en 3 casos, del algoritmo que se muestra a continuación, definido para realizar el cálculo del valor de la suma de la sucesión:

$$\frac{1}{a}, \frac{1}{a-1}, \frac{1}{a-2}, \dots, \frac{1}{1}$$

siendo a un entero comprendido entre 1 y 100.

PROGRAMA SUC 01

Variables

Reales: a , Suma

1. Inicio [Pseudocódigo aprenderaprogramar.com]

2. Llamar EntraDatos
3. Llamar Calculo
4. Mostrar "El valor del sumatorio es", Suma

5. Fin

Módulo EntraDatos

1. Mientras $a \leq 0$ ó $a > 100$ Hacer

1.1 Mostrar "Por favor introduzca un número entero comprendido entre 1 y 100"

1.2 Pedir a

1.3 $a = \text{Redondear}(a)$

Repetir

2. Mostrar "El dato base es", a

FinMódulo

Módulo Calculo

Variables

Enteras: i

1. Hacer

1.1 $\text{Suma} = \text{Suma} + 1 / a$

1.2 $a = a - 1$

1.3 $i = i + 1$

Repetir Mientras $a < > 0$

2. Mostrar "Contabilizados", i , "términos"

FinMódulo

SOLUCIÓN

Dado que nos pide evaluar 3 casos hemos de decidir entre 1 y 100, qué valores escogemos. El criterio que vamos a seguir es coger los dos extremos más un punto intermedio. Los valores de “arranque” y “terminación” suelen ser singulares, por lo que normalmente los probaremos cuando verifiquemos el algoritmo. Los valores que elegimos son 1, 50, 100. Venimos utilizando tablas de variables pues resultan didácticas para nuestro objetivo. Pero si quisiéramos realizar una evaluación con 50 pasadas por un bucle necesitaríamos, como mínimo, 50 filas de una tabla. Y para 100, pues 100 filas. Resulta excesivo. Tendríamos dos opciones:

- Con una buena calculadora realizar la verificación mental del algoritmo.
- Resolverlo con “programación rápida”.

Preferimos el segundo método, ya que “confiar en nuestra cabeza” para un gran número de operaciones no nos garantiza el estar libres de error. Dado que aún no hemos hablado de programación con un lenguaje específico, vamos a resolver el ejercicio para los valores 1, 3, 5.

a = 1			
Estado	i	Suma	a
Inicio de Cálculo	0	0	1
1ª iteración	1	1	0 [Entra bucle] [Sale bucle]
Muestra en algoritmo principal	~	1	~

a = 3			
Estado	i	Suma	a
Inicio de Cálculo	0	0	3
1ª iteración	1	1/3	2 [Entra bucle]
2ª iteración	2	$(1/3) + (1/2)$	1
3ª iteración	3	$(1/3) + (1/2) + 1$	0 [Sale bucle]
Muestra en algoritmo principal	~	1,833	~

a = 5			
Estado	i	Suma	a
Inicio de Cálculo	0	0	5
1ª iteración	1	1/5	4 [Entra bucle]
2ª iteración	2	(1/5) + (1/4)	3
3ª iteración	3	(1/5) + (1/4) + (1/3)	2
4ª iteración	4	(1/5) + (1/4) + (1/3) + (1/2)	1
5ª iteración	5	(1/5) + (1/4) + (1/3) + (1/2) + 1	0 [Sale bucle]
Muestra en algoritmo principal	~	2,283	~

Comentarios: Las tablas se han construido respetando expresiones tipo algebraico en vez de poner sus equivalentes decimales para facilitar la comprensión de los procesos. En efecto, resulta más claro escribir $(1/5) + (1/4) + (1/3)$ que $0,78\bar{3}$.

EJERCICIO 2 VERIFICACIÓN DEL ALGORITMO DE CÁLCULO DEL VALOR DE LA SUMA DE UNA SUCESIÓN

Verificar que el programa SUC02, algoritmo que se facilita a continuación, funciona correctamente para valores de entrada 0, 1,25 y 5:

- Suponiendo que se ejecuta el programa tres veces.
- Suponiendo que el usuario realiza los tres cálculos en un misma ejecución del programa.

PROGRAMA SUC02
Variables: Enteras: E Reales: Dato, Raiz01, Raiz02, Suce
1. Inicio [Pseudocódigo aprenderaprogramar.com]
2. Mientras E <> 2 Hacer
2.1 Mostrar "Elija 1. Cálculo 2. Salir" 2.2 Pedir E 2.3 Si E = 1 Entonces Llamar EntraDatos Llamar Proceso(Dato) PorValor Llamar Resultados FinSi
Repetir
3. Fin

Módulo EntraDatos

1. **Hacer**

Mostrar "Por favor, introduzca número entero entre 0 y 100"

Pedir Dato

Dato = Redondear(Dato)

Repetir Mientras Dato < 0 ó Dato > 100

FinMódulo

Módulo Proceso(Num: Enteros)

1. Raiz01 = SQR(Num)

2. Raiz02 = - Raiz01

3. Suce = 0

4. **Mientras Num >= 0 Hacer**

Suce = Suce + SQR(Num)

Num = Num - 1

Repetir

FinMódulo

Módulo Resultados

Mostrar "Dato base =", Dato

Mostrar "Raiz01=", Raiz01

Mostrar "Raiz02=", Raiz02

Mostrar "Valor de la suma de los términos de la sucesión =", Suce

FinMódulo

SOLUCIÓN

Entrada 0					
Estado	Dato	Copia Dato (Num)	Raiz01	Raiz02	Suce
Módulo Entradatos	0	~	~	~	~ [Entra bucle] ~ [Sale bucle]
Inicio módulo Proceso	0	0	0	0	0
Líneas 1, 2, 3 módulo Proceso	~	~	0	0	0
1ª iteración	~	- 1	~	~	0 [Entra bucle] 0 [Sale bucle]
Módulo Resultados	0	~	0	0	0

Entrada 1,25					
Estado	Dato	Copia Dato (Num)	Raiz01	Raiz02	Suce
Módulo Entradatos	1,25	~	~	~	~ [Entra bucle]
Redondeo	1	~	~	~	~ [Sale bucle]
Inicio módulo Proceso	1	1	0	0	0
Líneas 1, 2, 3 módulo Proceso	~	~	1	- 1	0
1ª iteración	~	0	~	~	1 [Entra bucle]
2ª iteración	~	- 1	~	~	1 [Sale bucle]
Módulo Resultados	1	~	1	- 1	1

Entrada 5					
Estado	Dato	Copia Dato (Num)	Raiz01	Raiz02	Suce
Módulo Entradatos	5	~	~	~	~ [Entra bucle] ~ [Sale bucle]
Inicio módulo Proceso	5	5	0	0	0

Estado	Dato	Copia Dato (Num)	Raiz01	Raiz02	Suce
Líneas 1, 2, 3 módulo Proceso	~	~	$\sqrt{5}$	$-\sqrt{5}$	0
1ª iteración	~	4	~	~	$\sqrt{5}$ [Entra bucle]
2ª iteración	~	3	~	~	$\sqrt{5} + \sqrt{4}$
3ª iteración	~	2	~	~	$\sqrt{5} + \sqrt{4} + \sqrt{3}$
4ª iteración	~	1	~	~	$\sqrt{5} + \sqrt{4} + \sqrt{3} + \sqrt{2}$
5ª iteración	~	0	~	~	$\sqrt{5} + \sqrt{4} + \sqrt{3} + \sqrt{2} + \sqrt{1}$
6ª iteración	~	-1	~	~	$\sqrt{5} + \sqrt{4} + \sqrt{3} + \sqrt{2} + \sqrt{1}$ [Sale bucle]
Módulo Resultados	5	~	2,236	-2,236	8,382

Entrada 0, 1,25, 5						
Estado	E	Dato	Copia Dato (Num)	Raiz01	Raiz02	Suce
Evaluación inicial	0	~	~	~	~	~ [Entrada bucle exterior]
Pide E	1	~	~	~	~	~
Módulo Entradatos	~	0	~	~	~	~ [Entra bucle] [Sale bucle]
Inicio módulo Proceso	~	0	0	0	0	0
Líneas 1, 2, 3 módulo Proceso	~	~	~	0	0	0
1ª iteración	~	~	-1	~	~	0 [Entra bucle] [Sale bucle]
Módulo Resultados	~	0	~	0	0	0
Evaluación inicial	1	~	~	~	~	~
Pide E	1	~	~	~	~	~
Módulo Entradatos	~	1,25	~	~	~	~

Estado	E	Dato	Copia Dato (Num)	Raiz01	Raiz02	Suce
Redondeo	~	1	~	~	~	~
Inicio módulo Proceso	~	1	1	0	0	0
Líneas 1, 2, 3 módulo Proceso	~	~	~	1	-1	0
1ª iteración	~	~	0	~	~	1 [Entra bucle]
2ª iteración	~	~	-1	~	~	1 [Sale bucle]
Módulo Resultados	~	1	~	1	-1	1
Evaluación inicial	1	~	~	~	~	~
Pide E	1	~	~	~	~	~
Módulo Entradatos	~	5	~	~	~	~ [Entra bucle] ~ [Sale bucle]
Inicio módulo Proceso	~	5	5	1	-1	1 [Ojo a esta situación]
Líneas 1, 2, 3 módulo Proceso	~	~	~	$\sqrt{5}$	$-\sqrt{5}$	0
1ª iteración	~	~	4	~	~	$\sqrt{5}$ [Entra bucle]
2ª iteración	~	~	3	~	~	$\sqrt{5} + \sqrt{4}$
3ª iteración	~	~	2	~	~	$\sqrt{5} + \sqrt{4} + \sqrt{3}$
4ª iteración	~	~	1	~	~	$\sqrt{5} + \sqrt{4} + \sqrt{3} + \sqrt{2}$
5ª iteración	~	~	0	~	~	$\sqrt{5} + \sqrt{4} + \sqrt{3} + \sqrt{2} + \sqrt{1}$
6ª iteración	~	~	-1	~	~	$\sqrt{5} + \sqrt{4} + \sqrt{3} + \sqrt{2} + \sqrt{1}$ [Sale bucle]
Módulo Resultados	~	5	~	2,236	-2,236	8,382
Evaluación inicial	1	~	~	~	~	~
Pide E	2	~	~	~	~	~ [Impide entrada al núcleo programa]
Evaluación inicial	2	~	~	~	~	~ [Sale del bucle exterior]

Comentarios: Si comparamos el resultado del programa, tanto para tres ejecuciones con un dato por ejecución como para una sola ejecución con tres datos, con el “cálculo normal”, comprobamos que el funcionamiento es correcto.

En las tres primeras tablas no hemos evaluado la variable E porque no tiene trascendencia: se entiende que para meter un dato el usuario ha de elegir el 1 y después sale eligiendo el 2. No tiene que hacerse un seguimiento de todas las variables: el programador decidirá cuáles seguir en función del objetivo de la verificación que esté haciendo.

En la cuarta tabla sí evaluamos la variable E por convertirse en elemento activo de la gestión del programa.

Observamos que existe una columna llamada *Dato* y otra *Copia Dato(Num)*. Esto obedece a la existencia del módulo genérico *Proceso*, el cual es invocado en el algoritmo principal de esta manera:

Llamar Proceso(Dato) PorValor

Recordemos que el hecho de invocar *PorValor* supone que el valor de *Dato* se transfiere a *Num* y *Num* funciona como variable local temporal. Por ello disponemos una columna independiente para seguir la evolución de *Num*. Si la invocación fuera *PorVariable*, se transferiría la variable *Dato* al módulo. En este caso no habiéramos tenido una columna *Copia Dato(Num)*, sino simplemente habiéramos reflejado los cambios en la columna *Dato*.

Otra cuestión a reseñar es la que se produce en la fila número 20 de la cuarta tabla, que hemos señalado con un comentario de [Ojo a esta situación]. Lo que comprobamos ahí es que *Raiz01*, *Raiz02* y *Suce* presentan unos valores no asociados al dato que se está procesando en ese momento (5), sino al que se procesó anteriormente (1). Ya lo comentamos cuando desarrollamos el programa pero volvemos a repetirlo. Si el acumulador *Suce* no se establece a cero al empezar el módulo *Proceso*, obtendremos resultados erróneos. Esto es así porque las variables *Raiz01*, *Raiz02* y *Suce* son globales, y por tanto conservan “memoria” de todo lo que ha pasado, hecho que no ocurre con las variables locales.

En las tablas que venimos usando a veces empleamos expresiones algebraicas como $\sqrt{5} + \sqrt{4} + \sqrt{3}$ en vez de valores reales equivalentes como 5,968 para facilitar el seguimiento del programa y el de la tabla. Piénsese que si escribimos 5,968 no tenemos información de cómo se ha generado ese dato, que igual podría provenir de $\sqrt{5} + \sqrt{4} + \sqrt{3}$ como de $\sqrt{356184}$ ó de $4,718 + 1,25$. En resumen, tenemos que pensar que la tabla nos puede ser de utilidad en una consulta dentro de un mes y cuanto más clara sea más fácil nos resultará entenderla.

Próxima entrega: CU00239A

Acceso al curso completo en [aprenderaprogramar.com](http://www.aprenderaprogramar.com) --> Cursos, o en la dirección siguiente: http://www.aprenderaprogramar.com/index.php?option=com_content&view=category&id=36&Itemid=60