

Organización y Arquitectura de Computadoras

Práctica 6: Lenguaje ensamblador

Sandra del Mar Soto Corderi
Edgar Quiroz Castañeda

31 de marzo del 2019

1. Preguntas

1. A partir del ejercicio 4:

$$4 \cdot \sum_{n=0}^m \frac{(-1)^n}{2n+1}$$

a) ¿A qué valor tiende la serie?

Es la fórmula de Leibniz[2] para aproximar π .

b) ¿A cuántos dígitos se puede aproximar ese valor?

Como es precisión sencilla, se pueden aproximar hasta 23 bits del valor en binario, que equivalen a 7 dígitos de precisión en decimal[1].

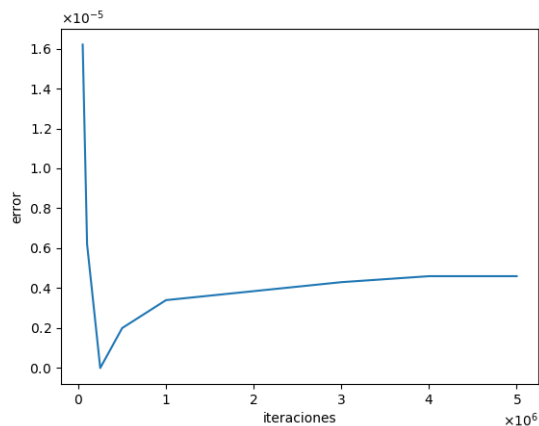
c) ¿Cuántas iteraciones son necesarias para llegar a esa aproximación?

Figura 1: Resultado de las aproximaciones

Iteraciones	Aproximación
50000	3.1415758
10000	3.1415758
250000	3.141592
500000	3.141594
1000000	3.1415954
3000000	3.1415963
4000000	3.1415966
5000000	3.1415966

Los primeros 7 dígitos de π son 3.141592[3].

Figura 2: Error por iteraciones al intentar aproximar 3.141592



Se puede apreciar en (2) que se llega a un error de 0 en 250,000 iteraciones. Tanto antes como después de ese punto, la aproximación empeora entre más se aleja de ese número.

Esto es de esperarse para iteraciones menores a 250,000, pero es inusual para valores mayores, pues entre mayor sea m el valor de la serie calculada debería ser más cercana al valor de la serie infinita.

Este error puede deberse a que al ser el m grande, el término $\frac{1}{2n+1}$ se vuelve más pequeño, por lo el error en su representación en punto flotante 32 bits puede aumentar. Así, la aproximación empeora.

2. ¿Existe alguna diferencia en escribir programas en lenguaje ensamblador comparado con escribir programas en lenguajes de alto nivel?

Escribir programas de alto nivel permite abstraer e implementar conceptos sin necesidad de tener en cuenta la traducción a código máquina, lo que hace que se pueda ignorar toda la parte física de la computadora.

3. ¿En qué casos es preferible escribir programas en lenguaje ensamblador y en qué casos es preferible hacerlo con un lenguaje de alto nivel?

Al depender de la arquitectura de la computadora, el código en lenguaje ensamblador no es portable, además de que requiere conocimiento de la máquina particular con la se está trabajando. Sin embargo, el código producido es lo más eficaz posible.

Entonces, se debería usar ensamblador cuando se esté diseñando código específico de hardware que requiere ser muy eficaz, como lo requieren algunos compiladores o ciertos fragmentos de sistemas operativos.

Referencias

- [1] *IEEE Standard for Floating-Point Arithmetic*, IEEE Std 754-2008, 2008. [Online]. Disponible: <https://ieeexplore.ieee.org/document/4610935>. [Consultado: 28-Mar-2019].
- [2] E. W. Weisstein, *Pi Formulas*, Wolfram MathWorld. [Online]. Disponible: <http://mathworld.wolfram.com/PiFormulas.html>. [Consultado: 28-Mar-2019].
- [3] M. Huberty, K. Hayashi and C. Vang, *100,000 Digits of Pi*, Geom.uiuc.edu, 1997. [Online]. Disponible: <http://www.geom.uiuc.edu/~huberty/math5337/groupe/digits.html>. [Consultado: 28-Mar-2019]