Universidad Nacional Autónoma de México

Facultad de Ciencias

Organización y Arquitectura de Computadoras
2025-2

Práctica 07

Docentes:
José Galaviz Ricardo Pérez Ximena Lezama

Autores:
Fernanda Ramírez Juárez Ianluck Rojo Peña

Fecha de entrega: Jueves 3 de abril de 2025

Ejercicios.

Ejercicio 1.

```cpp
#include <iostream>
using namespace std;

//Funcion que implementa la serie de Leibniz con el que nos aproximamos al valor de pi
int main() {
  cout << "Ingrese un valor para m: ";
  int m;          //Creamos la varible m para los terminos de la serie
  cin >> m;       //Leemos la varible del usuario
  float r = 0.0f; //El resultado a imprimir

  //Bucle para realizar la serie hasta los m terminos
  for (int n = 0; n <= m; n++) {
    //Calculamos cada termino de la serie, alternamos los valores de 1 y -1
    float j = ((n % 2 == 0) ? 1.0f : -1.0f) / ((2.0f * n) + 1.0f);
    r += j; //Sumamos la division obtenida a nuestros resultados previos
  }

  cout << "Resultado de la serie: " << r * 4.0f << endl;
}
```

Listing 1: Código en C++ Calculador de Serie

Ejercicio 4.

Razones por las cuales se asignaron los siguientes valores a las variables utilizadas en el cálculo:

Constante de gravitación universal (G)

- Valor asignado: $6.674e-11 = 6.67430 \times 10^{-11}$.
- Este valor, ademá de ser tomado como referencia en el archivo pdf para la práctica 7; es una constante física obtenida de forma empírica, que determina la intensidad de la fuerza de atracción gravitatoria entre los cuerpos. Se denota por G y aparece tanto en la ley de gravitación universal de Newton como en la teoría general de la relatividad de Einstein. La medida de G fue obtenida implícitamente por primera vez por Henry Cavendish en 1798. El valor actual corresponde al valor oficial publicado por CODATA de la forma:

$$6.67430 \times 10^{-11} \frac{N \cdot m^2}{kg^2}$$

- Fuente: Wikipedia - Constante de gravitación universal.

Masa de la Tierra:

- Valor asignado: $5.9722e24 = 5.9722 \times 10^{24}$.
- Este valor es ampliamente aceptado y corresponde a la mejor estimación actual con una incertidumbre relativa de 10^{-4}.
- Fuente: Wikipedia - Earth Mass.

Masa de la Luna:

- Valor asignado: $7.349e22 = 7.349 \times 10^{22}$.
- Valor aceptado en astronomía y utilizado en cálculos de fuerza gravitacional.
- Fuente: Wikipedia - Luna.

Distancia entre la Tierra y la Luna:

- Valor asignado: 384400.0.
- Basado en la información proporcionada por la NASA.
- Fuente: NASA Space Place.

Cálculo de la Fuerza Gravitacional:

Como se puede observar, en el programa en ensamblador para el ejercicio 4, el resultado obtenido es:

$$1.9823577584943926E26 = 1.9823577584943926 \times 10^{26} \quad (1)$$

Tomando como referencia el trabajdo realizado porFernández, J. L. en: Fisicalab - Fuerza de la Luna

El cual toma valores muy aproximados a los utilizados para el programa en ensamblador:

Masa de la Tierra = $M_T = 5.98 \times 10^{24} kg$
Masa de la Lunea = $M_L = 7.34 \times 10^{22} kg$
Distancia de la Tierra a la Luna = $R = 384 \times 10^3 km = 384 \times 10^6 m$
Fuerza Gravitacional = G = $6.67430 \times 10^{-11} \frac{N \cdot m^2}{kg^2}$
Aplicando la ley de gravitación universal:

$$F = G \cdot \frac{M_T \cdot M_L}{R^2} \tag{2}$$

Sustituyendo los valores:

$$F = 6.674 \times 10^{-11} \cdot \frac{(5.9722 \times 10^{24}) \cdot (7.349 \times 10^{22})}{(3.844 \times 10^8)^2} = 1.98 \times 10^{20} \tag{3}$$

Notemos que el resultado es similar, dentro del margen esperado debido a las diferencias en las cifras significativas y redondeos.

Referencias.

- Colaboradores de Wikipedia. **(2025, enero 29)**. *Constante de gravitación universal.* [Wikipedia, la Enciclopedia Libre]. `https://es.wikipedia.org/wiki/Constante_de_gravitaci%C3%B3n_universal`
- Contributors to Wikimedia projects. **(2025, 28 marzo)**. *Earth mass.* [Wikipedia, la Enciclopedia Libre].. `https://en.wikipedia.org/wiki/Earth_mass`
- Colaboradores de Wikipedia. **(2025, marzo 26)**. *Luna.* [Wikipedia, la Enciclopedia Libre].. `https://es.wikipedia.org/wiki/Luna`
- NASA. **(s. f.)**. *How Far Away Is the Moon?.* [NASA]. `https://spaceplace.nasa.gov/moon-distance/sp/`
- Fernández, J. L. **(s. f.)**. *Ejercicio: La fuerza de la Luna.* [Fisicalab]. `https://www.fisicalab.com/ejercicio/898`

Preguntas.

1. En el ejercicio 1:

   - ¿A qué valor tiende la serie?

     Cómo se mencionó en los comentarios del código, la serie es una serie de Leibniz la cual tiende al valor de pi π.

     Lo podemos comprobar fácilmente al ejecutar el código e ingresar valores relativamente grandes para m y esperar como salida 3.1416...

   - ¿A cuántos dígitos estaría limitado nuestro resultado con precisión sencilla? (Flotante) Justifica tu respuesta.

     En C++ los valores para el tipo float maneja una precisión de entre 6 y 7 dígitos decimales. En ensamblador en MIPS los registro de punto flotante almacenan 32 bits, lo que nos da una precisión similar. Es decir la serie esta limitada con punto flotante de precisión sencilla a aproximadamente 6-7 dígitos significativos.

   - ¿Cuántas iteraciones son necesarias para calcular el mayor número de dígitos?

     Cómo lo comentamos anteriormente, a medida que aumentamos el valor de m (el número de iteraciones en la serie de Leibniz) la aproximación a π se vuelve más precisa.

     Sin embargo, debido a que el programa utiliza números de punto flotante, tenemos que el límite en la cantidad de dígitos exactos que podemos obtener es de 7; antes de que la precisión se vea afectada por los errores numéricos.

     Al ejecutar el programa con $m = 10^6(1,000,000\ iteraciones)$, el resultado obtenido es 3.1415954, notemos que los primeros cinco decimales coinciden con los de π. Si aumentamos el número de iteraciones a $m = 10^7(10,000,000\ iteraciones)$, el resultado cambia a 3.1415968, notemos que, aunque los primeros cinco decimales siguen siendo correctos, el sexto decimal se desvía, ya que en π el sexto decimal es 2, mientras que en nuestra aproximación es 6.

     Por otro lado, si reducimos las iteraciones a $m = 10^5(100,000\ iteraciones)$, el resultado obtenido es 3.1415858, donde solo los cuatro primeros decimales coinciden con π, y a partir del quinto la precisión comienza a deteriorarse.

     Con esto, podemos concluir que el número óptimo de iteraciones para obtener la mayor cantidad de dígitos exactos de π es aproximadamente es 10^6 iteraciones.

2. Cuando tenemos un número doble guardado a lo largo de 2 registros, ¿Qué datos guarda cada registro?

   Tomemos de ejemplo el siguiente codigo:

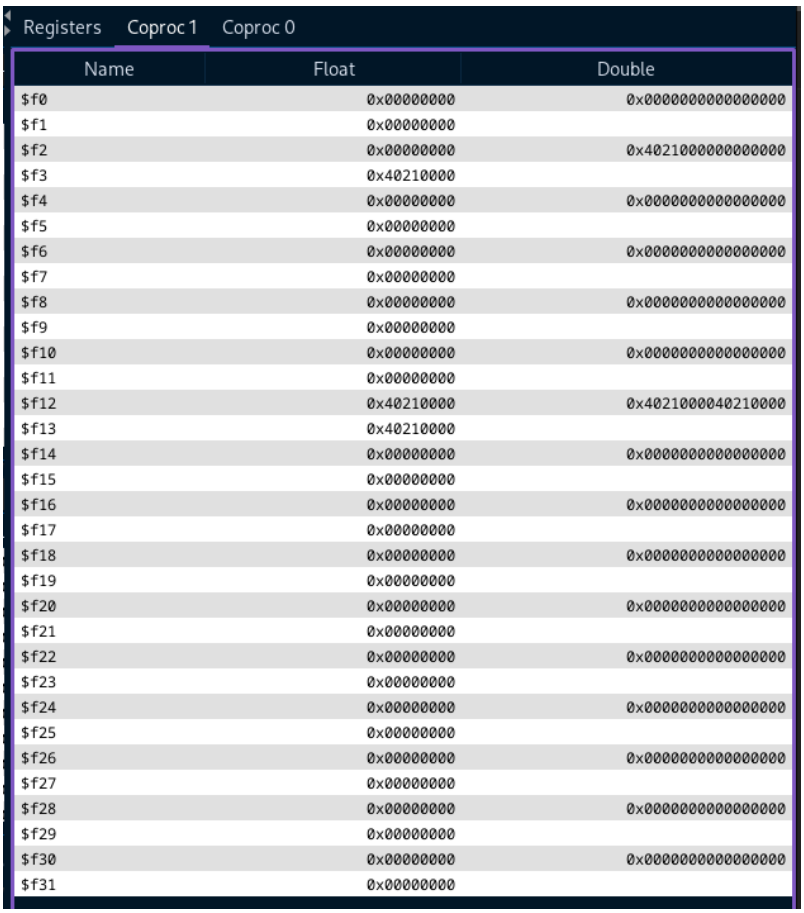
```
    .data
    num1: .double 8.5
    end_line: .asciiz ".\n"

    .text
    .globl main

    main:
    l.d $f2, num1
    add.d $f2, $f2, $f0
    add.d $f12, $f2, $f0
    li $v0, 3
    syscall
    li $v0, 4
    la $a0, end_line
    syscall
    add.s $f12, $f3, $f0
    li $v0, 2
    syscall
    li $v0, 10
    syscall
```

Listing 2: Código en MIPS

   En la siguiente imagen vemos lo que cada registro guarda:



Name	Float	Double
$f0	0x00000000	0x0000000000000000
$f1	0x00000000	
$f2	0x00000000	0x4021000000000000
$f3	0x40210000	
$f4	0x00000000	0x0000000000000000
$f5	0x00000000	
$f6	0x00000000	0x0000000000000000
$f7	0x00000000	
$f8	0x00000000	0x0000000000000000
$f9	0x00000000	
$f10	0x00000000	0x0000000000000000
$f11	0x00000000	
$f12	0x40210000	0x4021000040210000
$f13	0x40210000	
$f14	0x00000000	0x0000000000000000
$f15	0x00000000	
$f16	0x00000000	0x0000000000000000
$f17	0x00000000	
$f18	0x00000000	0x0000000000000000
$f19	0x00000000	
$f20	0x00000000	0x0000000000000000
$f21	0x00000000	
$f22	0x00000000	0x0000000000000000
$f23	0x00000000	
$f24	0x00000000	0x0000000000000000
$f25	0x00000000	
$f26	0x00000000	0x0000000000000000
$f27	0x00000000	
$f28	0x00000000	0x0000000000000000
$f29	0x00000000	
$f30	0x00000000	0x0000000000000000
$f31	0x00000000	

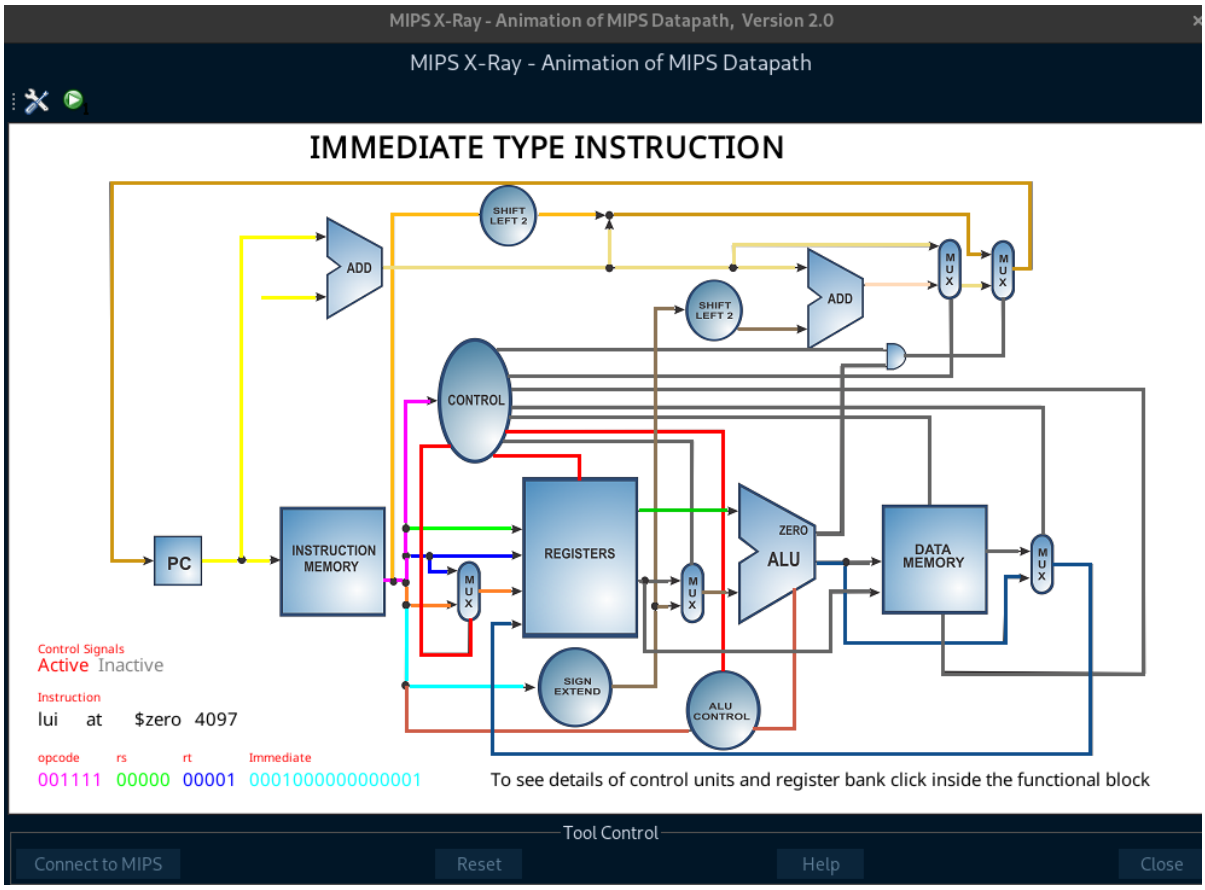
En MIPS, los números de punto flotante de doble precisión (double) ocupan dos registros consecutivos de punto flotante. Como se puede ver en la imagen, al guardar el double num1 en $f2, también MIPS lo guarda en $f3 pero en tipo float.

- **Registro Par (Inferior):** El registro par (en este caso $f2) contiene la mitad inferior de los 64 bits del double, 0x4021000000000000. Lo mismo para el registro $f12 en un principio.
- **Registro Impar (Superior):** El registro impar siguiente (en este caso $f3) contiene la mitad superior de los 64 bits del double, 0x40210000 Lo mismo se guarda en el resigtro $f13.

Después en la línea `add.s $f12 $f3 $f0` cargamos desde memoria, la mitad inferior 0x40210000 y la mitad superior 0x40210000. Es por eso que tenemos al final 0x4021000040210000 está almacenado en $f12.

Cómo última mencion, al correr el programa, la primer impresión del número es 8.5, que corresponde al primer valor guardado en $f12 (0x4021000000000000) antes de la suma con el valor de la mitad inferior. Debajo se imprime 2.515625, que es el valor final asignado (0x4021000040210000) de la suma de la mitad inferior con la mitad superior.

3. En MARS, en la barra de herramientas, en la pestaña de Tools, existe la herramienta llamada MIPS X-Ray, conecta esta herramienta y corre un programa línea por línea. ¿Qué significan los números resaltados de color magenta, verde, azul y azul claro que se encuentran abajo de la instrucción?



Los números resaltados en colores son los diferentes campos de la instrucción MIPS en formato de código de máquina.

- **Magenta (001111):** Como se ve en la imagen es el *opcode*, el cual indica el tipo de instrucción. 001111 corresponde a la instrucción lui *(Load Upper Immediate)*.
- **Verde (00000):** Es el registro rs *(source register)*. En la instrucción lui, este campo no se usa por lo que suele establecerse en 0.
- **Azul (00001):** Es el registro rt *(target register)*, donde se almacenará el valor inmediato cargado en la parte alta del registro. El cual corresponde al registro $at.
- **Azul claro (0001000000000001):** Representa el valor inmediato *(Immediate)*. En el lui, este valor se coloca en los 16 bits superiores del registro destino. El inmediato salió como 4097 en decimal.