Profesores: Carlos González, Leo Medina
Ayudantes: Alexander Palma, Florencia Corvalán, Marco Hernández, Maximiliano Orellana, Ricardo Ruz

Laboratorio 3. Acercándose al Hardware: Programación en Lenguaje Ensamblador

Objetivos de aprendizaje

- Usar MARS (un *IDE* para MIPS) para escribir, ensamblar y depurar programas MIPS
- Escribir programas MIPS incluyendo instrucciones aritméticas, de salto y memoria
- Comprender el uso de subrutinas en MIPS, incluyendo el manejo del *stack*
- Realizar llamadas de sistema en MIPS mediante "*syscall*"
- Implementar algoritmos en MIPS para resolver problemas matemáticos

Entrega

Sube los archivos creados, junto con tu informe, a través de Google Classroom. Todos los archivos con código MIPS deben poder ensamblarse y ejecutarse en el simulador MARS, y deben estar debidamente comentados. Guarda cada programa en un archivo distinto con el nombre de la parte que corresponde. Por ejemplo, tu trabajo de la Parte 1, pregunta A, guárdalo en el archivo "parte1a.asm".

Parte 1: Aproximación de funciones matemáticas

Se pide escribir un programa que calcule una aproximación de las siguientes funciones, evaluadas en un número entero no negativo:

1) Coseno
2) Seno hiperbólico
3) Logaritmo natural

Para aproximar estas funciones, utiliza expansiones de Taylor en torno a 0, de orden 7 o superior. El programa **debe** utilizar para el cálculo de las multiplicaciones, divisiones y la factorial los procedimientos del Laboratorio 2. Además, el programa debe pedir al usuario, vía consola MARS, el número entero no negativo para el cual se quieren evaluar las funciones, y debe mostrar en la misma consola los resultados con 2 decimales, sin atender a errores de precisión. Finalmente, cuantifica la eficiencia en la ejecución de instrucciones comparando el número de instrucciones efectivamente ejecutadas versus el número de instrucciones escritas (utiliza las herramientas "Instruction Counter" o "Instruction Statistics" en MARS).

Parte 2: Implementación en MIPS de una función recursiva

La sucesión de Fibonacci comienza con los números 0 y 1 y, a partir de éstos, cada término se calcula como la suma de los dos anteriores. Considera la siguiente implementación en lenguaje de alto nivel de una función recursiva que entrega en elemento n-ésimo de la sucesión de Fibonacci:

Profesores: Carlos González, Leo Medina
Ayudantes: Alexander Palma, Florencia Corvalán, Marco Hernández, Maximiliano Orellana, Ricardo Ruz

```
int fibonacci(int n) {
    if (n == 0) {
        return 0;
    }else if (n == 1) {
        return 1;
    }
    return fibonacci(n-1) + fibonacci(n-2);
}
```

Escribe un programa en MIPS que calcule el elemento n-ésimo de la sucesión de Fibonacci según el esquema recursivo de la función mostrada arriba. Como condición, se pide no convertir a una solución iterativa.

Parte 3: "Memoización" en MIPS

Modifica tu implementación de la Parte 2 para "memoizar" los resultados, esto es, almacenarlos en un "búfer" para un posterior acceso más rápido. Por simplicidad, asume que la "memoización" se realiza en un arreglo de largo "n" para cualquier elemento "n". Además, el arreglo se inicializa con ceros. La función modificada luce de la siguiente manera en lenguaje de alto nivel.

```
int fib(int n, int* memolist) {
    if (n == 0) {
        return 0;
    } else if (n == 1) {
        return 1; }
    if (memolist[n]) {
        return memolist[n];
    }
    memolist[n] = fib(n-1, memolist) + fib(n-2, memolist);
    return memolist[n];
}
```

Informe

El informe para entregar debe contar con lo siguiente:

- Introducción que incluya el problema, solución y objetivos de esta experiencia
- Marco teórico que explique los conceptos necesarios para entender el trabajo desarrollado
- Explicación breve del desarrollo de la solución y cómo se llegó a esta
- Resultados de cada parte del laboratorio

Profesores: Carlos González, Leo Medina
Ayudantes: Alexander Palma, Florencia Corvalán, Marco Hernández, Maximiliano Orellana, Ricardo Ruz

- Conclusiones

Exigencias

- El informe escrito debe ser entregado en formato PDF y no puede exceder 10 páginas de contenido, sin considerar portada ni índice. En caso contrario, por cada página extra se descontará 5 décimas a la nota final.
- Tanto el código fuente como el informe deben ser enviados por medio de la plataforma Google Classroom en un archivo comprimido, cuyo nombre debe incluir el RUT de el/la alumno/a (ej.: lab2_12345678-9.zip).

Recomendaciones

- **Consultar a los ayudantes en sus correspondientes sesiones de laboratorio.**
- En caso de dudas o problemas en el desarrollo, asiste a la sesión de laboratorio.
- **Consultar a los ayudantes en sus correspondientes sesiones de laboratorio.**
- Por si no quedó claro :) :P : **Consultar a los ayudantes en sus correspondientes sesiones de laboratorio.**

Descuentos

- Por cada día de atraso se descontará un punto a la nota final de este laboratorio.
- Por cada tres faltas ortográficas o gramaticales en el informe, se descontará una décima a la nota del informe.
- Por cada falta de formato en el informe se descontará una décima a la nota del informe.
- Por cada página extra en el informe se descontarán 5 décimas a la nota final del informe.

Evaluación

- La nota del laboratorio será el promedio aritmético del código fuente con el informe.
- En caso de que no se entregue el informe o el código fuente, se evaluará con la nota mínima.
- Este laboratorio debe ser entregado el viernes 25 de Junio de 2021, hasta las 11:59 hrs. Los descuentos por atraso corren a contar de las 12:01 hrs del mismo día.