PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

IIC2343 - Arquitectura de Computadores
Abril 2021-1

Tarea 1

Fecha de entrega: 3 de mayo de 2021, hasta las 23:59.

Objetivos

Para esta tarea, esperamos que se familiaricen con el lenguaje de programación en assembly RISC-V[1] (*risk-five*) y con el ambiente de programación y emulación de RARS[2], que será el emulador que utilizaremos durante el curso. Para esto, deberán programar un algoritmo simple y utilizando exclusivamente valores de 8 bits (1 byte), determinar si se cumple o no cierta condición.

Enunciado

En computación, es muy útil saber cuando una matriz es invertible (antes de intentar invertirla), para una serie de aplicaciones, como ray-tracing en gráficos 3D o tecnologías Multiple-input, Multiple-output en telecomunicaciones. Una forma de determinar si una matriz cuadrada $M_{n\times n}$ es no singular (o invertible), es determinando si M tiene una diagonal estrictamente dominante, y este resultado se conoce como el teorema de Levy–Desplanques[3]. Para ello, se necesita que M cumpla con la siguiente condición

$$|a_{i,i}| > \sum_{j\neq i} |a_{i,j}| \ , \quad \forall\ i$$

donde $a_{i,j}$ denota la entrada de la i-ésima fila de la j-ésima columna. Dicho de otra manera, para todas las filas, se debe cumplir que el valor absoluto del elemento $a_{i,i}$, que es el valor que adquiere la diagonal para dicha fila, debe ser mayor a la suma absoluta de todos los otros valores $a_{i,j}$ de la i-ésima fila.

En su programa, ustedes deberán revisar si es que una matriz M cumple con esta condición. Si la matriz resultante posee una diagonal estrictamente dominante, deberán escribir `1` en el registro `a0` (Revisar *green card* de la sección Assembly) y terminar el programa, de lo contrario, deberán guardar `0` en el registro `a0` y terminar el programa. El no acabar el programa con `0` o `1` en dicho registro, o no reflejando correctamente la condición exigida para la matriz, resultará en 0 puntos en el test correspondiente.

[1] `https://riscv.org/`
[2] `https://github.com/TheThirdOne/rars`
[3] `https://dml.cz/bitstream/handle/10338.dmlcz/101601/CzechMathJ_29-1979-2_9.pdf`

Input

Cada programa deberá tener la siguiente estructura

```
.globl  start
.data
    # --- MATRIZ ---
    M: .byte a11, a12, a13, ..., ann # representación de matriz M
    N: .byte i # tamano de la matriz
    # --- END MATRIZ ---
    # de aca para abajo van sus variables en memoria
.text
    start:
        # aca va su codigo :3
```

Donde `M` es la dirección de inicio en memoria de un arreglo que representa los valores de cada entrada en la matriz **cuadrada** $M_{n\times n}$, y `N` es la dirección en memoria del byte que representa el tamaño de la matriz. Son libres de agregar más variables que ustedes consideren pertinentes a la sección `.data`, pero cuando probemos sus tareas, utilizaremos las letras anteriormente mencionadas para indicar la matriz y su tamaño, por lo que les pedimos dejar esa parte al principio del programa.

Las entradas `aij` pueden adquirer valores enteros en el intervalo $[-127, 127]$, y el tamaño máximo de la matriz es de 11×11.

Evaluación

Tests

Su algoritmo será probado sobre 6 matrices distintas, de distintos tamaños y con distintos valores. Cada test aprobado vale 0,5 puntos y el total es la suma de estos.

3 puntos.

Correctitud

Se evaluará correctitud del algoritmo, es decir, que se recorra adecuadamente la matriz y se obtengan los valores pertinentes para determinar si se cumple la condición de matriz con diagonal estrictamente dominante.

2 puntos.

Assembly

El código deberá estar adecuadamente comentado, de manera de facilitar la corrección, además de respetar las convenciones de llamada para los registros, especificadas en la segunda página del *green card*[4] de RISC-V. En la figura 1 se indica la convención de llamada, de acuerdo a esta, por ejemplo, para guardar una variable en el registro `x5`, deberán hacerlo a través de `t0`. En su código nunca debiesen llamar a un registro a través de la notación `xN`, además de usar los registros de acuerdo con la descripción provista.

[4] `https://inst.eecs.berkeley.edu/~cs61c/fa17/img/riscvcard.pdf`

```
# para sumar dos numeros, escribir
addi a2, a2, 3
addi a3, a3, zero
add a4, a3, a4
# NO HACER
addi x12, x12, 3
addi x13, x13, x0
add x14, x13, x14
```

RISC-V Calling Convention			
Register	ABI Name	Saver	Description
x0	zero	---	Hard-wired zero
x1	ra	Caller	Return address
x2	sp	Callee	Stack pointer
x3	gp	---	Global pointer
x4	tp	---	Thread pointer
x5-7	t0-2	Caller	Temporaries
x8	s0/fp	Callee	Saved register/frame pointer
x9	s1	Callee	Saved register
x10-11	a0-1	Caller	Function arguments/return values
x12-17	a2-7	Caller	Function arguments
x18-27	s2-11	Callee	Saved registers
x28-31	t3-t6	Caller	Temporaries
f0-7	ft0-7	Caller	FP temporaries
f8-9	fs0-1	Callee	FP saved registers
f10-11	fa0-1	Caller	FP arguments/return values
f12-17	fa2-7	Caller	FP arguments
f18-27	fs2-11	Callee	FP saved registers
f28-31	ft8-11	Caller	FP temporaries

Figura 1: Convención de llamada para registros. Free & Open RISC-V Reference Card, RISC-V Organization.

Deberán trabajar sólo con enteros, con signo o sin signo, de a lo más 8 bits, `.byte` en RISC-V. Además de tomar las precauciones necesarias para manejar condiciones de borde.

1 punto.

Emulador (IMPORTANTE)

Si bien son libres de programar en el editor que prefieran e incluso compilar/emular en la herramienta que les sea más cómoda, la corrección será con el emulador RARS anteriormente mencionado, por lo que su tarea deberá pasar el *assembler* y ejecutar en dicho emulador. **No pasar la etapa del *assembler* en RARS implica la reprobación automática de la tarea con nota 1.0**.

El emulador también tiene soporte para una serie de *syscalls*, el uso de cualquiera de estas implica la reprobación automática de la tarea con nota 1.0.

Nota tarea

La nota se calculará de la siguiente manera:

$$Puntos + 1 = Nota\ tarea$$

La nota de la tarea se redondea a la decena.

Entrega

La entrega de la tarea será a través de la plataforma SIDING en el formulario habilitado para ello hasta el día 3 de mayo de 2021, a las 23:59.

El formato a entregar será un (1) archivo `.txt` con el código assembly RISC-V de la tarea. La entrega de un binario compilado, log, dump de memoria, programa en Python, x86 o cualquier otra cosa resultará en la reprobación de la tarea.