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

IIC2343 - Arquitectura de Computadores (I/2025)

Actividad de programación

Sección 3 - Pauta de evaluación

Pregunta 1: Explique el código (3 ptos.)

En el siguiente fragmento de código se realiza el llamado de una subrutina `func_x_n`:

```
.data
  x:            .word 7
  n:            .word 3
  result:       .word 0
.text
    lw a0, x
    lw a1, n
    la s0, result
    addi s1, zero, 1
    addi sp, sp, -4
    sw ra, 0(sp)
    jal ra, func_x_n
    lw ra, 0(sp)
    addi sp, sp, 4
    sw a0, 0(s0)
    li a7, 10
    ecall
  func_x_n:
    beq a1, s1, func_x_n_end
    addi sp, sp, -8
    sw ra, 0(sp)
    sw a0, 4(sp)
    addi a1, a1, -1
    jal ra, func_x_n
    lw ra, 0(sp)
    lw t0, 4(sp)
    addi sp, sp, 8
    mul a0, a0, t0
    func_x_n_end:
      jalr zero, 0(ra)
```

Este fragmento representa el cómputo de una función $f(x, n)$. A partir de este:

1. (1.5 ptos.) Indique, con argumentos y en términos de x y n, lo que retorna la función $f(x, n)$. Por ejemplo, $f(x, n) = x + n$. Se otorgan **0.75 ptos.** por la correctitud de la descripción del retorno y **0.75 ptos.** por justificación.

   Solución: La función computa la potencia de x a la n, *i.e.* $f(x, n) = x^n$. Esto se evidencia en el hecho de que el valor de retorno, almacenado en el registro `a0`, es igual a $x \times f(x, n-1)$, con caso base $f(x, 1) = x$. De esta forma, se tiene que:

   $$f(x, n) = x \times f(x, n-1) = x \times x \times f(x, n-2) = \cdots = x \times x \times x \times \cdots \times x = x^n$$

   No es necesario que se haga un detalle completo del código, pero sí que se demuestre que se entiende el cómputo recursivo de la potencia de x a la n.

2. (1.5 ptos.) Indique, con argumentos, si el fragmento anterior respeta o no la convención de llamadas de RISC-V. Se otorgan **0.75 ptos.** si indica de forma correcta si se respeta o no la convención y **0 ptos.** si su respuesta es incorrecta. Por otra parte, se otorgan **0.75 ptos.** por entregar una justificación válida respecto a su respuesta.

   Solución: El fragmento anterior **sí respeta** la convención de llamadas de RISC-V. Los motivos son los siguientes:

   - Se respalda `ra` antes de cada llamado.
   - Se usan los registros `a0` y `a1` para los argumentos de `func_x_n`, y `a0` para su retorno.
   - Se respalda el registro `a0` antes de cada llamado recursivo, considerando su sobreescritura; su uso posterior al llamado; y el hecho de ser *caller-saved*.

Pregunta 2: Elabore el código (3 ptos.)

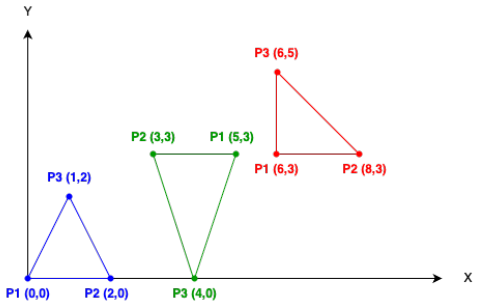
Elabore, utilizando el Assembly RISC-V, un programa que **calcule el área del triángulo que se forma a partir de tres puntos en un plano de dos dimensiones**. Las coordenadas de los puntos se guardarán individualmente en la memoria, como se muestra a continuación:

```
# Calcular el área de un triángulo que tiene un segmento paralelo al eje x
.data
  area:   .word 0
  b:      .word 0
  h:      .word 0
  P1:     .word 0, 0  # Coordenadas (X,Y) del punto P1
  P2:     .word 2, 0  # Coordenadas (X,Y) del punto P2
  P3:     .word 1, 2  # Coordenadas (X,Y) del punto P3
.text
  # Su código aquí. Guarde los resultados de base, altura y área en las variables correspondientes.
```

Puede asumir los siguientes supuestos:

- Los tres puntos **siempre** formarán un triángulo válido.
- Los tres puntos **siempre** se encontrarán en el primer cuadrante del plano cartesiano, *i.e.* $x \geq 0$ y $y \geq 0$ para todo punto (x, y).
- Los puntos `P1` y `P2` conformarán la base del triángulo y **siempre** será paralela al eje x.
- El cálculo del área se obtiene a partir de la fórmula $\frac{b \times h}{2}$, siendo b la extensión de la base y h la extensión de la altura del triángulo.

A continuación, tres ejemplos de triángulos válidos en el contexto de este problema:



La asignación de puntaje se distribuirá de la siguiente forma:

- **1 pto.** por calcular correctamente la base. Se descuentan **0.5 ptos.** si existe como máximo un error de implementación y no se asigna puntaje si existe más de uno.
- **1 pto.** por calcular correctamente la altura. Se descuentan **0.5 ptos.** si existe como máximo un error de implementación y no se asigna puntaje si existe más de uno.
- **1 pto.** por calcular correctamente el área del triángulo. Se descuentan **0.5 ptos.** si existe como máximo un error de implementación y no se asigna puntaje si existe más de uno.

IMPORTANTE: No es necesario que respete la convención en este ejercicio.

Solución: A continuación, un código que cumple lo pedido:

```
# Calcular el área de un triángulo que tiene un segmento paralelo al eje x
.data
  area:    .word 0
  b:       .word 0
  h:       .word 0
  P1:      .word 0, 0  # Coordenadas (X,Y) del punto P1
  P2:      .word 2, 0  # Coordenadas (X,Y) del punto P2
  P3:      .word 1, 2  # Coordenadas (X,Y) del punto P3
.text
  init:
    addi s0, zero, -1
    addi s1, zero, 2
    la s2, b
    la s3, h
    la s4, area
    la t0, P1
    lw t2, 4(t0)      # t2 = P1.y
    lw t0, P1         # t0 = P1.x
    lw t1, P2         # t1 = P2.x
    la t3, P3
    lw t3, 4(t3)      # t3 = P3.y
  base_compute:
    sub t0, t0, t1
    bgt t0, zero, height_compute
    mul t0, t0, s0
  height_compute:
    sw t0, 0(s2)
    sub t2, t2, t3
    bgt t2, zero, area_compute
    mul t2, t2, s0
  area_compute:
    sw t2, 0(s3)
    mul t4, t0, t2
    div t4, t4, s1
    sw t4, 0(s4)
  end:
    addi a7, zero, 10
    ecall
```

Para evaluar su correctitud, se modificarán los valores de `P1`, `P2` y `P3` para verificar el cómputo correcto del área según sea el caso. Si se detecta *hardcodeo*, **se otorgarán 0 puntos.**