# Evaluación del Trabajo de Laboratorio Número 1

Arquitectura de Computadoras 17/04/2018

---

**Instrucciones**

- La presente evaluación es individual y podrá utilizar todo el material escrito del que disponga.
- La misma consta de 3 ejercicios en total y la duración será de 90 minutos.
- En cada ejercicio se indica el tiempo estimado para resolverlo y el puntaje que otorga el mismo.
- La evaluación se desarrollará en el simulador y deberá mostrarse a un ayudante funcionando antes de enviarla por correo electrónico a *evolentini@gmail.com*
- La legibilidad del código, la cantidad de comentarios y la buena presentación serán tenidas en cuenta para la calificación final.

---

**Ejercicios para resolver**

1) **45 Minutos - 60 Puntos (20, 20, 20)**
   En los ambientes de procesamiento gráfico o procesamiento de señales, son muy usadas las interpolaciones lineales de la forma $Z = X * a + Y * (1 - a)$, donde $X$, $Y$, $Z$ son vectores y $a$ es una constante entre 0 y 1.
   a) Escriba una subrutina `intlin` en lenguaje ensamblador RV32IM del RISC-V que reciba como parámetros los valores correspondientes a los elementos $X_i$, $Y_i$ y la constante $a$, y devuelva el valor correspondiente de $Z_i$. A pesar de que la constante es un valor real entre 0 y 1, considérela como un entero entre 0 y 1000 para simplificar.
   b) Escriba la subrutina `intvec` en lenguaje ensamblador RV32IM del del RISC-V, que reciba como parámetros las direcciones base de los vectores $X$, $Y$ y $Z$, el tamaño $m$ de los vectores, y el valor de la constante $a$ y calcule la interpolación lineal para todo el vector utilizando la función realizada en el apartado anterior.
   c) Escribe un programa que calcule la interpolación lineal sobre dos vectores utilizando la subrutinas anteriores y muestre el vector de resultado por la pantalla.

2) **25 Minutos - 25 Puntos**
   Escriba un programa en lenguaje ensamblador RV32IM del RISC-V que reciba un número entero en el registro **`a0`** y devuelva por consola la raíz cuadrada aproximada del mismo. Para calcular la misma utilice aproximaciones sucesivas calculando el producto de los números enteros hasta obtener el más cercano inferior al número ingresado. Agregue las cadenas de texto adecuadas para indicar qué significa el resultado.

3) **20 Minutos - 15 Puntos**
   Escriba un programa en lenguaje ensamblador RV32IM del RISC-V que reciba la dirección de una cadena de caracteres en el registro **`a0`**, que vaya imprimiendo uno a uno los caracteres de la misma hasta encontrar el valor cero, y en ese momento finalice mostrando por consola en una linea separada la longitud de la cadena.

**Soluciones propuestas**

1) ...

2) ...

3) ...

## Soluciones completas

1)

```
            .data
X:          .word 10, 20, 30, 40, 50
Y:          .word 50, 40, 30, 20, 10
Z:          .word  0,  0,  0,  0,  0
sep:        .asciiz ", "

            .text
            .globl main

# a0: constante de proporcion a
# a1: direccion del elemento Xi
# a2: direccion del elemento Yi
# a3: direccion del elemento Zi
intlin:     lw t0, 0(a1)
            mul t0, t0, a0
            lw t1, 0(a2)
            li t2, 1000
            sub t2, t2, a0
            mul t1, t1, t2
            add t0, t0, t1
            sw t0, 0(a3)
            jr ra

# a0: constante de proporcion a
# a1: direccion del elemento Xi
# a2: direccion del elemento Yi
# a3: direccion del elemento Zi
# a4: cantidad de elementos en los vectores
intvec:     addi sp, sp, -4
            sw ra, 0(sp)
intlazo:    beq a4, zero, intfin
            jal intlin
            addi a1, a1, 4
            addi a2, a2, 4
            addi a3, a3, 4
            addi a4, a4, -1
            j intlazo
intfin:     lw ra, 0(sp)
            addi sp, sp, 4
            jr ra

main:       li a0, 700
            la a1, X
            la a2, Y
            la a3, Z
            li a4, 5
            jal intvec

            li s0, 5
            la s1, Z
lazo:       beq t0, zero, final
            li a0, 1
            lw a1, 0(s1)
            ecall
            addi s1, s1, 4
            addi s0, s0, -1
            beq s0, zero, final
            li a0, 4
            la a1, sep
            ecall
            j lazo
final:      li a0, 10
```

```
            ecall
```

2)

```
        .data
cad:    .asciiz "La raiz cuadrada es: "

        .text
        .globl main

main:   li a0, 143

        li t0, 1            # Inicia el valor del resultado en uno
lazo:   addi  t0, t0, 1     # Incrementa el valor del resultado
        mul t1, t0, t0      # Calcula el cuadrado del resultado
        bge a0, t1, lazo    # Comprar el cuadrado con el numero
        addi t0, t0, -1     # Decrementar el ultimo incremento del resultado

fin:    li a0,4             # Mostrar una cadena por pantalla
        la a1, cad          # Puntero a la cadena para mostrar
        ecall               # Pedido de servicio
        li a0,1             # Mostrar un entero por pantalla
        mv a1, t0           # Entero para mostrar por pantalla
        ecall               # Pedido de servicio

        li a0, 10           # Terminar el programa
        ecall               # Pedido de servicio
```

3) .

```
        .data
cad:    .asciiz "\nLa longitud de la cadena es: "
datos:  .asciiz "Cadena de prueba"

        .text
        .globl main

main:   la a0, datos
        mv s1, a0               # Apunto al inicio de la cadena
        mv zero, s0             # Inicializo la cantidad en cero
        li a0, 11               # Servicio para mostrar un caracter ASCII
lazo:   lb a1, 0(s1)            # Cargo el elemento del vector
        beq a1, zero, final     # Si el elemento es cero termino el lazo
        ecall                   # Muestro por consola el caracter leido
        addi s0, s0, 1          # Incremento la cantidad de elementos
        addi s1, s1, 1          # Muevo el puntero al siguiente elemento
        add s0, s0, t0          # Sumo el elemento al acumulador
        j lazo                  # Repito para el siguiente elemento
final:  li a0, 4                # Servicio del entorno para mostrar cadenas
        la a1, cad              # Apunto a la cadena a mostrar
        ecall                   # Muestro por pantalla la primer cadena
        li a0, 1                # Servicio del entorno para mostrar enteros
        mv a1, s0               # Cargo el valor de la suma para mostrar
        ecall                   # Muestro el valor de la suma por pantalla
        li a0, 10               # Servicio del entorno para terminar el programa
        ecall                   # Termino el programa
```