Nro. ord.	Apellido y nombre	L.U.

Organización del Computador I - **Parcial**

2023 1C

Ej.1	Ej.2	Ej.3	Ej.4	Ej.5	Nota

Corrector:

Aclaraciones

- Anote apellido, nombre, LU en *todos* los archivos entregados.
- El parcial es domiciliario y todos los ejercicios deben estar aprobados para que el parcial se considere aprobado. Hay dos fechas de entrega, en ambos casos el conjunto de ejercicios a entregar es el mismo. En la primera instancia deberán defender su trabajo frente a su tutorx, quien les ayudará también a encaminar el trabajo de los ejercicios pendientes, si los hubiera.
- El link de entrega es: `https://forms.gle/ae8C7qZtTydU96eCA`. Ante cualquier problema pueden comunicarse con la lista docente o preferentemente con el/la corrector/a.
- La fecha límite de entrega es el martes 21 de Noviembre a las 17:00. El coloquio será el jueves 30 de Noviembre en el horario de cursada de los jueves (TM: 9 a 13hs - TT: 17 a 21hs) de **forma presencial** en un aula que figura en el calendario. Durante el coloquio vamos a relizar preguntas sobre todos los temas vistos en la práctica, incluyendo convención de llamada y uso de la pila.
- Todas las respuestas deben estar correctamente justificadas.
- Vamos a realizar un chequeo por diferencia binaria sobre los archivos entregados para ordernarlos según un ranking de similitud y poder evaluar situaciones anómalas con las entregas.

Introducción

Este parcial está dividido en **dos** ejercicios de implementación de código ensamblador para `RISCV32I`. Uds van a recibir un archivo con un esqueleto de código que deben completar, pueden utilizar el simulador `RIPES` para probar su programa. Pueden descargarlo en `https://github.com/mortbopet/Ripes`. Toda la información necesaria está disponible en la `Guía Práctica de RISC-V` que se puede acceder libremente en:
`http://riscvbook.com/spanish/guia-practica-de-risc-v-1.0.5.pdf`.

Esperamos que entreguen el archivo con la implementación y otro donde expliquen cómo resolvieron los ejercicios.

Ejercicios

Ejercicio 1

Escribir un programa que recorra un arreglo de 12 elementos de 32 bits, donde cada elemento contiene un valor representable en 16 bits, extienda este valor a su representación en complemento a 2 de 32 bits y lo sume a un acumulador. Por ejemplo, si en los primeros 32 bits se encuentra el valor $0xf000$ debe extenderlo como $0xffffff000$ y sumarlo al acumulador.

```c
#include <stdio.h>
#include <stdint.h>

#define N 12

int main()
{
    int32_t arr[N] = {
        0xffff, 0x1111, 0xfff0, 0x8888,
        0x0003, 0x0001, 0x0000, 0x8088,
        0x0202, 0x8081, 0x2222, 0x5555
    };
    int32_t suma = 0;
    int32_t i = 0;
```

```
    int32_t n = 12;
    for(i = 0; i < n; i++){
        int32_t tmp = arr[i] << 16;
  suma += (tmp >> 16);
    }
    printf("Suma %08x", suma);

    return 0;
}
```

Ejercicio 2

Escribir un programa que tome los números del arreglo fuente, aplica un xor sobre éstos y el valor correspondiente del destino y lo almacena en destino.

```
#include <stdio.h>
#include <stdint.h>

#define N 12

int main()
{
    int32_t src[N] = {
        0xffffffff, 0x95555555, 0xf4444444, 0xf1111111,
        0xffffff00, 0xf5005555, 0x95444444, 0xf1113311,
        0xff00ffff, 0xf5550055, 0xa4444433, 0xa1551111
    };
    int32_t dst[N] = {
        0xf5005555, 0x95444444, 0xf1113311, 0xffffff00,
        0xf1111111, 0xffffffff, 0x95555555, 0xf4444444,
        0xa1551111, 0xff00ffff, 0xf5550055, 0xa4444433
    };
    int32_t i = 0;
    int32_t n = 12;
    for(i = 0; i < n; i++){
  dst[i] ^= src[i];
  printf(" %08x", dst[i]);
    }

    return 0;
}
```