

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 ordenarlos 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 RISC-V32I. 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 y acá la conjunción de aquellos que estén en posiciones impares.

```

1 #include <stdio.h>
2 #include <stdint.h>
3
4 #define N 12
5
6 int main()
7 {
8     int16_t arr[N] = {
9         0xffff, 0x9555, 0xf444, 0xf111,
10        0xffff, 0xf500, 0x9544, 0xf111,
11        0xff00, 0xf555, 0xa444, 0xa155
12    };
13    int16_t suma = 0xffff;
14    int32_t i = 0;
15    int32_t n = 12;
16    for(i = 0; i < n; i++){

```

```

17         if((i & 0x1) == 0x1)
18             suma &= arr[i];
19     }
20     printf("Suma %04x", suma);
21
22     return 0;
23 }

```

## 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.

```

1  #include <stdio.h>
2  #include <stdint.h>
3
4  #define N 12
5
6  int main()
7  {
8      int32_t src[N] = {
9          0xffffffff, 0x95555555, 0xf4444444, 0xf1111111,
10         0xffffffff00, 0xf5005555, 0x95444444, 0xf1113311,
11         0xff00ffff, 0xf5550055, 0xa4444433, 0xa1551111
12     };
13     int32_t dst[N] = {
14         0xf5005555, 0x95444444, 0xf1113311, 0xffffffff00,
15         0xf1111111, 0xffffffff, 0x95555555, 0xf4444444,
16         0xa1551111, 0xff00ffff, 0xf5550055, 0xa4444433
17     };
18     int32_t i = 0;
19     int32_t n = 12;
20     for(i = 0; i < n; i++){
21         dst[i] ^= src[i];
22         printf(" %08x", dst[i]);
23     }
24
25     return 0;
26 }

```