Instituto Tecnológico de Costa Rica **Área Académica de Ingeniería en Computadores** **Programa de Licenciatura en Ingeniería en Computadores** **Curso:** CE-4301 Arquitectura de Computadores I **Profesor:** Ronald García Fernández **Semestre:** I, 2019	**Simulacro I Examen Parcial** **Fecha:** 29 de Marzo, 2019 **Puntos totales:** 100 **Puntos obtenidos:**_____

Nombre: _____ Carné: _____

Instrucciones Generales

- Trabaje individualmente
- Utilice cuaderno de examen u hojas blancas numeradas para resolver la prueba.
- Escriba de manera legible y ordenada.
- Sea lo más detallado posible en sus respuestas **(cuando se le pide)** no deje nada abierto a interpretaciones.
- Utilice bolígrafo para resolver la prueba. No se aceptarán reclamos sobre respuestas con lápiz
- El fraude se castiga según estipula el reglamento de enseñanza-aprendizaje del TEC.
- Tiempo para resolver la prueba es de 2 horas.
- No se permite el uso de celulares o algún otro tipo de dispositivo móvil.
- Todo código o programa debe poseer comentarios. Caso contrario se asignará un puntaje igual a cero.

Parte I. Teoría [X puntos]
Responda las siguientes preguntas

1- Explique en que consiste el ILP, como puede ser mejorado (Elabore)
2- Describa detalladamente los componentes de un ISA
3- Explique por qué se dice que la arquitectura RISC-V es un ISA modular
4- En que consiste un ISA ortogonal
5- Describa las consideraciones principales de un ISA
6- Según la taxonomía de ***Flynn*** describa en que consiste un "***stream***", basado en esto realice una comparación detallada entre **MISD** y **SIMD**
7- Cuales son las consideraciones o premisas sobre las cuales se basa la ley de Amdahl
8- Describa la diferencia entre un sistema simétrico y uno no-simétrico
9- Explique en que 2 consideraciones que se deben tener al explorar el TLP en sistemas *multi-core*
10- Explique en como comúnmente se relacionan las clasificaciones ***CISC*** y ***RISC*** con ***LOAD/STORE*** y ***REG/MEM***. ¿Qué ventajas y desventajas poseen?
11- Describa mediante ejemplos las clasificaciones ***RAW, WAR, WAW***, además justifique por es erróneo pensar que ***WAW*** no genera ningún riesgo
12- Describa los diferentes modos de direccionamiento descritos según Hennessy. ¿Cuándo son estos normalmente usados?

Parte II. Desarrollo [100-X puntos]

Resuelva cada uno de los siguientes problemas recuerde indicar todos los pasos que lo llevaron a la solución, además **debe adjuntar su green card** en la cual indique las instrucciones empleadas, con el fin de evaluar su implementación.

1- Una operación que es requerida al manipular arreglos de caracteres (***char***) es la inversión del orden de estos, dicha operación consiste en cambiar el orden de los elementos de forma tal que el elemento inicial se convierta en el final, y el final en el inicial, como se muestra en el siguiente ejemplo (seudo C):
Ejemplo:

```
1   char array[N]  = {'a', 'b', 'c', 'd', … , 'n'};
2   //reverse process
3   char  array[N] = {'n', … , 'd', 'c', 'b', 'a' };
```

Su trabajo es implementar dicha función en lenguaje ensamblador (*asm*) **de su elección** con la restricción de que no puede usar instrucciones "mágicas" que permitan la inversión directa de una región en memoria. Para efectos prácticos suponga que su tamaño ***N*** de ***array*** se encuentra en memoria en la dirección ***0xFFFF1*** y que ***array*** inicia en la siguiente posición de memoria naturalmente alineada con el tipo de datos descrito, se le pide lo siguiente

a- Descripción del algoritmo que va implementar, indicando como va leer la memoria, que estructura de flujo va utilizar, etc.
b- ¿Cuál es la posición de memoria correcta en función de su ISA en donde inicia ***array***, como cambiaria dicha dirección si los datos son de tipo ***struct***? Justifique su respuesta
c- Código de asm que implemente lo descrito en el enunciado tomando en cuenta los tipos de datos,

modos de direccionamiento, etc.

d- Suponga que su código de asm va correr en una micro-arquitectura con pipeline de 5 etapas balanceado, identifique los riesgos que se encuentran en su implementación.

2- El siguiente código presenta el algoritmo para calcular el promedio simple de un vector de datos *a[]*

```
1   int sum, i;
2   int n = 12;
3   int a[] = {2, 6, 7, 4, 9, 12};
4
5   double mean() {
6   |
7   |  sum = 0;
8   |  for(i = 0; i < n; i++) {
9   |  |  sum+=a[i];
10  |  }
11  |  return (sum/n);
12  }
```

Respecto al código se le pide lo siguiente:

a- ¿Cuál es la problemática en la línea 11? ¿Cómo lo solucionaría?

b- Implemente el código en asm partiendo que el problema en 'a' fue resuelto

c- Mediante la técnica de loop unrolling transforme el código desarrollado en 'b', indicando todos los pasos, suponga que no tiene límite de registros, **pero no puede emplear un factor de 12**.

d- Suponiendo que el branch predictor empleado en la micro-arquitectura tiene una penalización de 10 ciclos de reloj cuando falla su predicción y una probabilidad de fallo de 25%, cuál sería la penalización máxima en el caso desarrollado en 'b' y 'c'

3- Se tiene un ISA con 32 registros, que permite operaciones entre registros, y valores inmediatos de 16 bits, el ISA tiene 142 instrucciones en categorizadas de la siguiente forma:

Tipo A: Instrucciones que emplean un registro de *source*, un registro de *destination* y un valor inmediato.

Tipo B: Instrucciones que emplean 2 registros de *source*, un registro de *destination*.

Tipo C: Instrucciones que emplean un valor inmediato de *source* y un registro de *destination*.

Tipo D: Instrucciones que emplean un registro de *source* y un registro de *destination*.

Si el ISA permite tamaño de instrucción variable con la restricción de que estas sean múltiplos de 1 byte se le pide:

a- Determine el tamaño de cada tipo de instrucción.
b- Determine el tamaño mínimo si NO se puede emplear tamaño variable.
c- Si se tiene una aplicación que posee 30% instrucciones tipo A, 25% tipo B, 25% tipo C y 20% tipo D. ¿Cuánto espacio se ahorraría en promedio, usando tamaño variable vs tamaño fijo?

4- Se tiene un programa que puede ser mejorado de tres diferentes maneras:

a- Mejora X brinda *Speedup* de 10

b- Mejora Y brinda *Speedup* de 25

c- Mejora Z brinda *Speedup* de 15

Una de las restricciones de este este programa es que **únicamente posible tener una mejora activa a la vez**.

Basado en lo anterior cómo es posible plantear la Ley de Amdahl para manejar múltiples mejoras activas una a la vez.

Empleando esta generalización como se podría obtener un *Speedup* total de 15, si las mejoras X y Z sólo pueden estar activas un 23% del tiempo.