


Arquitectura de Computadores /Interfaces y Arquitectura Hardware

SEGUNDO EXÁMEN PARCIAL

Nombre1: Nombre2: Nombre3:	Fecha 17 de abril de 2020
	Duración: 24 horas máximo

Trabajo en grupo: **antes de empezar** forme un grupo de exactamente 3 personas, las cuales van a trabajar en equipo para resolver el siguiente problema. Ingrese a la siguiente hoja de cálculo y escriba frente a los tres nombres de los integrantes del grupo un número entre 1 a 10 que los identifique.

	<u>Grupos Ejercicio Segundo Parcial</u>

En principio tendrán un día de plazo para entregar la solución del problema, con solo resolver el problema ya tiene una nota mínima de 4.0.

Pero aquí inicia una competencia entre los grupos, y el ganador será el que primero entregue la solución correcta del problema, es decir el primero obtendrá una calificación de éste componente de 5.0, El segundo 4.9, el tercero 4.8 y así sucesivamente.

Todos los grupos van a tener que resolver el problema, pero sin preguntarles ni pedir ayuda a otros compañeros, es un reto que pueden resolver entre ustedes.

El enlace se cierra mañana sábado 4:00 p.m, deben subir el programa .asm.

Recuerden que cualquier intento de plagio o copia será reportada.



Ejercicio práctico Assembly Language Segundo Parcial (20%)

[20% Resta BigDecimal]

Se requiere que usted escriba un programa en lenguaje ensamblador x86 NASM que realice la sustracción iterativa entre dos números en el formato Bigdecimal. Como verá, dentro de una estructura *for* se hace un llamado a una función o método llamado `subtrac()`, que recibe los parámetros a través de la pila. El resultado de la resta se almacena en la variable 'Result' ('dato1' - 'delta'). Dicho resultado se copia luego en `dato1`.

El programa Java que debe implementar en ensamblador es el siguiente:

```

1  import java.math.BigDecimal;
2  public class MyClass {
3      public static void main(String args[]) {
4          BigDecimal dato1 = new BigDecimal("000005.30");
5          BigDecimal delta = new BigDecimal("001013.90");
6          BigDecimal Result = new BigDecimal("000000.00");
7          for(int i=0; i<11; i++){
8              Result = dato1.subtract(delta); // data1=data1-delta;
9              dato1=Result;
10             System.out.println("BigDecimal dato1 = " + dato1);
11         }
12     }
13 }
14 }
15

```

El método subtract debe funcionar en todos los casos (es decir debe entregar el resultado correcto ya sea cuando sea positivo o sea negativo) y para operar para cualquier tamaño o cantidad de dígitos, suponga que las tres variables son del mismo tamaño.

El resultado de las operaciones del programa es el siguiente:

```

BigDecimal dato1 = -1008.60
BigDecimal dato1 = -2022.50
BigDecimal dato1 = -3036.40
BigDecimal dato1 = -4050.30
BigDecimal dato1 = -5064.20
BigDecimal dato1 = -6078.10
BigDecimal dato1 = -7092.00
BigDecimal dato1 = -8105.90
BigDecimal dato1 = -9119.80
BigDecimal dato1 = -10133.70
BigDecimal dato1 = -11147.60

```