DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
ESCUELA DE INGENIERÍA
PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

IIC2343 - Arquitectura de Computadores (I/2020)

Tarea 3

Caché y pipelining.
Fecha de entrega: 24 de junio, 2020. 18:00 horas.

Política de Integridad Académica

Los alumnos de la Escuela de Ingeniería deben mantener un comportamiento acorde al Código de Honor de la Universidad:

"Como miembro de la comunidad de la Pontificia Universidad Católica de Chile me comprometo a respetar los principios y normativas que la rigen. Asimismo, prometo actuar con rectitud y honestidad en las relaciones con los demás integrantes de la comunidad y en la realización de todo trabajo, particularmente en aquellas actividades vinculadas a la docencia, el aprendizaje y la creación, difusión y transferencia del conocimiento. Además, velaré por la integridad de las personas y cuidaré los bienes de la Universidad."

En particular, se espera que mantengan altos estándares de honestidad académica. Cualquier acto deshonesto o fraude académico está prohibido; los alumnos que incurran en este tipo de acciones se exponen a un procedimiento sumario. Específicamente, para los cursos del Departamento de Ciencia de la Computación, rige obligatoriamente la siguiente política de integridad académica. Todo trabajo presentado por un alumno (grupo) para los efectos de la evaluación de un curso debe ser hecho individualmente por el alumno (grupo), sin apoyo en material de terceros. Por "trabajo" se entiende en general las interrogaciones escritas, las tareas de programación u otras, los trabajos de laboratorio, los proyectos, el examen, entre otros. Si un alumno (grupo) copia un trabajo, los antecedentes serán enviados a la Dirección de Docencia de la Escuela de Ingeniería para evaluar posteriores sanciones en conjunto con la Universidad, las que pueden incluir reprobación del curso y un procedimiento sumario. Por "copia" se entiende incluir en el trabajo presentado como propio partes hechas por otra persona. Está permitido usar material disponible públicamente, por ejemplo, libros o contenidos tomados de Internet, siempre y cuando se incluya la cita correspondiente.

Objetivos

- Aprender a usar el tipo de dato `Decimal` en python.
- TBA[1].

Entrega y evaluación

Esta tarea es de carácter estrictamente **INDIVIDUAL**. Tiene dos tipos de preguntas evaluadas: programadas en *python* y teóricas de alternativas o similar disponibles en *canvas*. Para la pregunta del código de honor y las preguntas programadas, deberás enviar mediante un formulario de google:[2]

- Una imagen o PDF con tu respuesta a la pregunta del código de honor.
- Tus códigos en archivos `.py` independientes. **Sólo uno por pregunta.**

Debes referenciar correctamente[3] en caso de utilizar material externo. Tus programas serán corregidos vía *tests*, los *inputs* se darán por STDIN y debes entregar tus *outputs* por STDOUT, tal como en la tarea 1.

Importante: para evitar que se bloquée la corrección, se utilizará un *timeout* de 2 segundos de ejecución.

[1] TBA: *To Be Announced*

[2] Link completo:
https://docs.google.com/forms/d/e/1FAIpQLSe2m6Tqi2bfNlOk_O1BmT2rlq01hkxdOTnBmNpxPdn2jfRTHQ/viewform?usp=sf_link

[3] Si tienes dudas, usa el formato APA.

Código de honor

Escribe **a mano en un papel** el siguiente texto y fírmalo. A continuación tómale una foto o escanéalo y súbelo al formulario. **No hacerlo equivale a un 1 en todas las preguntas de la tarea.**

> Tarea 3 - IIC2343
>
> Yo, <tu nombre>, afirmo que respetaré el código de honor de la universidad y no cometeré ninguna falta a la ética ni a la política de integridad académica.
>
> <número de alumno>
> <firma>

1. Pregunta recuperativa de la Tarea 1

1.1. Contexto

¿Sabías que la convención IEEE754 en realidad detalla **5** tipos de datos distintos? Estos son: binary32 (*float*), binary64 (*double*), binary128, decimal64 y decimal128. Los primeros 3 utilizan la notación científica en base 2 mientras que los dos últimos la usan en base 10. En esta pregunta utilizarás la implementación de python del tipo de dato `Decimal` en un caso aplicado.

1.2. Lectura recomendada

1. Implementación de python. Contiene la documentación de la librería Decimal de python.

1.3. Caso aplicado:

La ley de redondeo establece que si el monto (en pesos chilenos) termina en 1, 2, 3, 4 o 5, se redondea hacia la decena inferior, mientras que si termina en 6, 7, 8 o 9, se redondea hacia la decena superior.

Eres el encargado de programar el software contable de las tiendas DCComercio y tu jefe te pide que le calcules: ¿Cómo y cuánto cambiarían los ingresos mensuales si el redondeo hacia arriba fuera a partir de 5, como nos enseñaron en el colegio?

Además, el DCComercio pidió un crédito hace algunos años para abrir sus nuevos locales, por lo que deberás calcular (utilizando la fórmula del interés compuesto descrita más abajo), en cuánto tiempo más (en meses) conseguirían pagar el crédito (sólo con el ingreso mensual calculado redondeando 5 hacia abajo).

Considera que destinarán el 33 % de sus ingresos mensuales de manera fija para ello. Utiliza el ingreso mensual que calcules y asume que se mantendrá estable en el tiempo.

Los montos se trabajarán en **decenas de pesos**, eso quiere decir que la cantidad `Decimal("421.5")` representa el monto "cuatro mil doscientos quince". Utiliza las 28 cifras de precisión que vienen por defecto.

1.3.1. Cálculo del interés compuesto

Supongamos que el monto del crédito que queda pendiente es x, la tasa de interés es i y el ingreso mensual destinado al pago del crédito es m.

Entonces, en cada mes, lo que se hará es pagar m pesos descontándolo del monto x y luego el monto resultante se multiplicará por $1 + i$ y esa será la cantidad que quede para el mes siguiente. Expresado matemáticamente:

$$\text{queda para el mes siguiente} = (x - m) \cdot (1 + i)$$

Como consideración adicional, existe un punto de equilibrio tal que pagar esa cantidad mantendrá el monto que queda para el pago y, si se paga menos, el monto que queda aumentará. El punto de equilibrio es:

$$m = \frac{x \cdot i}{1 + i}$$

1.4. *Inputs:*

Recibirás a través de STDIN:

1. El número de compras N que han realizado los clientes ese mes.
2. N líneas, cada una con el detalle de una compra. Se utilizará el formato:

   `k cantidad producto precio_unitario cantidad producto precio_unitario ....`

   Donde `k` indica el número de productos distintos que ha comprado ese cliente.
3. El monto que queda del crédito.
4. La tasa de interés.

Ejemplo:[4]

```
3
2 1 tomate 100.4 6 cerveza 659.9
4 8 tomate 803.2 6 huevo 435.0 2 salchicha 113.2 4 salmón 1298.0
1 1 toallitas_de_bebés 100.7
60000.0         // seiscientos mil pesos
0.02            // esto indica que es un 2% mensual
```

1.5. *Outputs:*

Deberás entregar a través de STDOUT:

1. El ingreso mensual redondeando el 5 hacia abajo.
2. El número de meses que faltan para terminar de pagar el crédito, redondeando el 5 hacia abajo. (Ver fórmula descrita más arriba)
3. El ingreso mensual redondeando el 5 hacia arriba.

[4] Lo que está con `//` son comentarios para la explicación, no forman parte del *input*.

Si con el ingreso mensual ves que no es posible terminar de pagar el crédito, debes imprimir el texto `"NUNCA"` en lugar de la cantidad de meses.

Ejemplo:

```
18614   // redondeando el 5 hacia abajo
11      // total de meses
18615   // redondeando el 5 hacia arriba
```

Anexo: paso de datos por STDIN

Para evitar tener que escribir constantemente o copiar y pegar los valores de entrada, los programadores ancestrales crearon una técnica milenaria para entregar a un proceso el contenido almacenado en un archivo por STDIN: el operador "<". A continuación les traspasaré los conocimientos de esta técnica arcana.

Supongamos que mi código está en el archivo `solucion_decimal.py` y tengo mi archivo `test.txt` con los valores de entrada del ejemplo que tengo más arriba.

Linux, MacOS y CMD:

Es tan simple como ejecutar en la consola:

```
$ python3 "./solucion_decimal.py" < "./test.txt"
```

El "$" representa el marcador de la línea.

PowerShell:

En la powershell tienen el operador reservado pero todavía no lo implementan (les hice una issue hace tiempo en el repositorio de GitHub). Sin embargo existe también una forma de hacerlo:

```
PS > get-content "./test.txt" | python3 "./solucion_decimal.py"
```

El "PS >" representa el marcador de la línea.