ICI517 Programación Paralela, Control #4
Escuela de Ingeniería Civil Informática,
Universidad de Valparaíso
2022, Semestre 1

Nombre Completo:_____

El Código 1 es un trozo de código que utiliza la unidad SIMD de un procesador. la variable **`vectorB`** es una matriz unidimendicional de tamaño $nx=2^k$, con $k \geq 3$. Utilice la documentación de Intel para conocer el funcionamiento de las Intrinsics y justificar de adecuadamente sus respuestas.

```
T out;
T* vectorB = new T[nx];
__m256 regDataIn256, regTotal256;

regTotal256 = _mm256_setzero_ps();

for (int i = 0; i < nx; i += R) {
  regDataIn256 = _mm256_load_ps(vectorB + i);
  regTotal256  = _mm256_add_ps(regDataIn256, regTotal256);
}

vectorOut = new float[S];
_mm256_storeu_ps(vectorOut, regTotal256);
out = vectorOut[0] + ... + vectorOut[S-1]
```

Código 1.

a) Determine cuál es el tipo de datos de **`T`**. (1pts)

El tipo de datos **`T`** tiene relación con el arreglo `vectorB` y la variable `out`.

opción 1)

La variable `out` es la suma de números en precisión simple (float), por lo que **`T`** debe ser `float`.

opción 2)

El arrreglo `vectorB` se utiliza en la operación `_mm256_load_ps()`, que según la documentación de intel, su sintaxis es:

```
__m256 _mm256_load_ps (float const * mem_addr)
```

Luego, vectorB debe ser un arreglo de float. Esto implica que T debe ser float.

b) Determine el valor de **R**. (2pts)

El ciclo for carga desde la memoria principal al registro vectorial `regDataIn256`.

```
regDataIn256 = _mm256_load_ps(vectorB + i);
```

Según la documentación de Intel, `_mm256_load_ps(float const* A)` carga 8 floats (256 bits en total) desde la dirección A en el registro vectorial `regDataIn256`.

Luego, realiza la siguiente operación:

```
regTotal256  = _mm256_add_ps(regDataIn256, regTotal256);
```

Según la documentación de Intel, `_mm256_add_ps(__mm256 A, __m256 B)` suma en forma vectorial los 8 floats de A y B.

Luego, el ciclo suma 8 `float` en cada iteración. Para que esto tenga sentido, el offset i debe ser equivalente a 8 `floats`, esto es, 256 bits.

Por lo tanto **R**=256.

c) Determine el valor de **S**. (2pts)

El arreglo `vectorOut` debe almacenar en memoria principal el contenido del registro vectorial `regTotal256`. Luego, debe almacer 8 `float`.

Por lo tanto, **S**=8

d) Explique el objetivo del algoritmo. (3pts)

En base a lo anterior, el algoritmo realiza la suma de todos los elementos del arreglo `vectorB`. Estos elementos son `floats`. Esta suma se realiza en forma vectoral, en grupos de 8 `float`.

e) Determine un algoritmo secuencial equivalente. (2pts)

```
acumulador = 0
Para cada elemento i de vectorB:
     acumulador = vectorB[i] + acumulador.
Fin Para Cada
sumaElementosArreglo = acumulador.
```