

**과 목 명:** 컴퓨터신기술특강

**담당교수**: 김병국 교수님

**학 과**: 컴퓨터소프트웨어학과

**학 년**: 4학년

**학 번**: 202312704

**이 름**: 서성준

**제 출 일**: 2023년 5월 12일

**SISD와 SIMD기법을 이용한 산술연산 프로그램**

1. **작업과정 설계**

**1-1. SISD를 이용한 산술연산 프로그램 작성**

* 1. **문제를 이해하고, 필요한 데이터와 변수를 정의합니다.**
  2. **프로그램에서 필요한 산술연산(덧셈, 뺄셈, 곱셈, 나눗셈 등)을 선택합니다.**
  3. **나눗셈에 대한 예외처리를 작성한다.**
  4. **반복문을 이용하여, 정의한 데이터와 변수에 대해 산술연산을 수행합니다.**
  5. **산술연산 결과를 출력합니다.**

**1-2. SIMD를 이용한 산술연산 프로그램 작성**

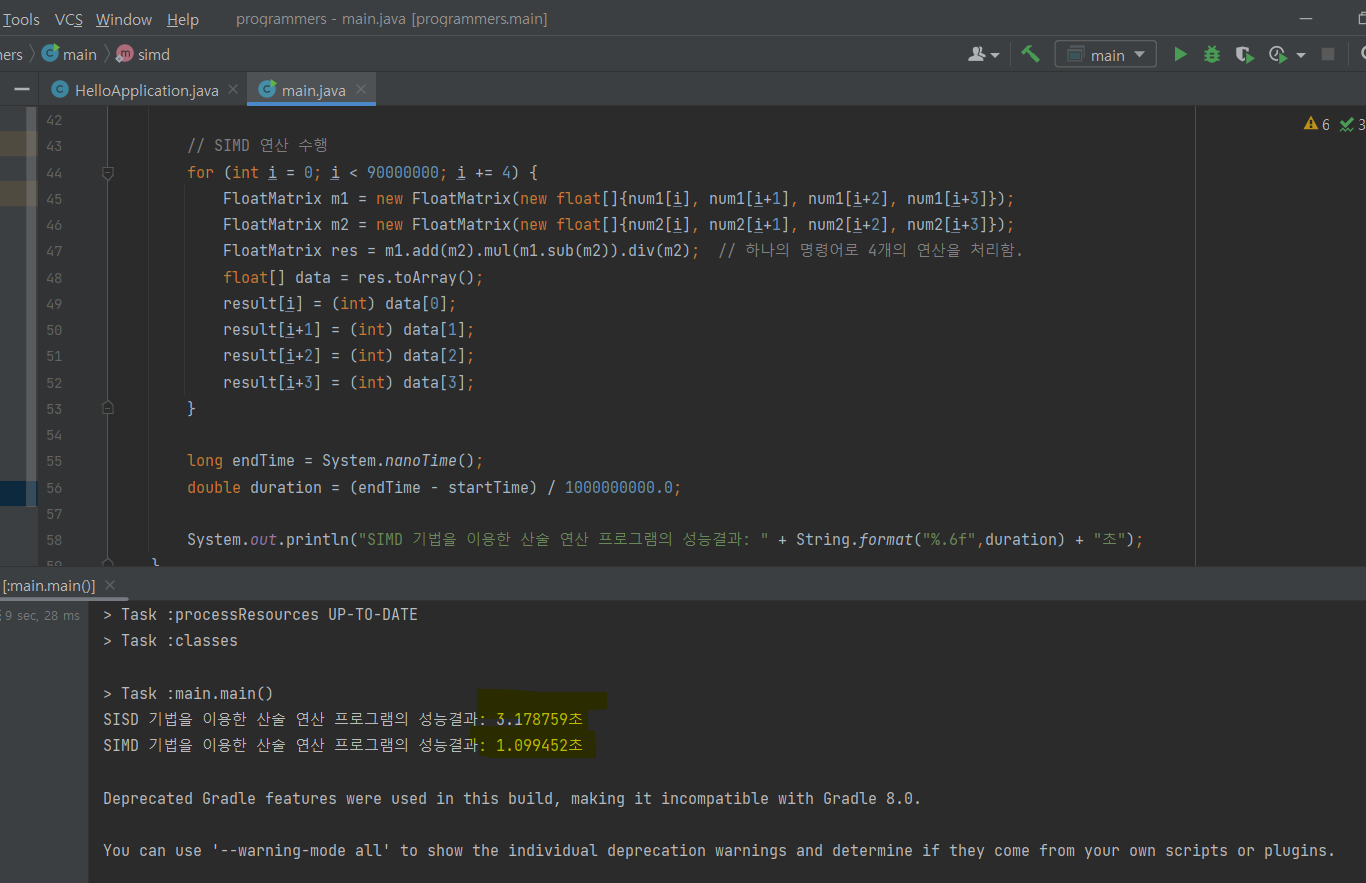
* 1. **문제를 이해하고, 필요한 데이터와 변수를 정의합니다.**
  2. **프로그램에서 필요한 산술연산(덧셈, 뺄셈, 곱셈, 나눗셈 등)을 선택합니다.**
  3. **나눗셈에 대한 예외처리를 작성한다.**
  4. **SIMD기술 중 하나인, 256비트 백터 레지스터를 사용하는 AVX기술로 SIMD연산을 수행한다.**
  5. **산술연산 결과를 출력합니다.**

**1-3. 성능 평가**

1. **데이터의 수가 적을 때는 SISD기법의 성능이 더 좋을 수 있다.**
2. **데이터의 수가 많을 때는 SIMD기법의 성능이 더 좋을 수 있다.   
   (여러 개의 큰 데이터를 한 번에 병렬로 처리하기 때문에)**
3. **코드**

package com.example.programmers;  
  
import org.jblas.FloatMatrix;  
  
public class main {  
  
 public static void main(String[] args) {  
 *sisd*();  
 *simd*();  
 }  
  
 public static void sisd(){ // SISD 기법  
 long startTime = System.*nanoTime*();  
 for (int i = 0; i < 90000000; i++) {  
 int num1 = (int) (Math.*random*()\*i) + 10000000;  
 int num2 = (int) (Math.*random*()\*i) - 10000000;  
 int hap = num1 + num2;  
 int sub = num1 - num2;  
 int gop = num1 \* num2;  
 float div = num2 != 0 ? num1 / (float)num2 : 0; // 나눗셈 (0으로 나누려하면 0 반환)  
 }  
  
 long endTime = System.*nanoTime*();  
 double duration = (endTime - startTime) / 1000000000.0;  
 System.*out*.println("SISD 기법을 이용한 산술 연산 프로그램의 성능결과: " + String.*format*("%.6f",duration) + "초");  
 }  
  
  
 public static void simd(){ // SIMD 기법  
  
 int[] num1 = new int[90000000];  
 int[] num2 = new int[90000000];  
 int[] result = new int[90000000];  
  
 // 배열 초기화  
 for (int i = 0; i < 90000000; i++) {  
 num1[i] = (int) (Math.*random*()\*i) + 10000000;  
 num2[i] = (int) (Math.*random*()\*i) - 10000000;  
 }  
  
 long startTime = System.*nanoTime*();  
  
 // SIMD 연산 수행  
 for (int i = 0; i < 90000000; i += 4) {  
 FloatMatrix m1 = new FloatMatrix(new float[]{num1[i], num1[i+1], num1[i+2], num1[i+3]});  
 FloatMatrix m2 = new FloatMatrix(new float[]{num2[i], num2[i+1], num2[i+2], num2[i+3]});  
 FloatMatrix res = m1.add(m2).mul(m1.sub(m2)).div(m2); // 하나의 명령어로 4개의 연산을 처리함.  
 float[] data = res.toArray();  
 result[i] = (int) data[0];  
 result[i+1] = (int) data[1];  
 result[i+2] = (int) data[2];  
 result[i+3] = (int) data[3];  
 }  
  
 long endTime = System.*nanoTime*();  
 double duration = (endTime - startTime) / 1000000000.0;  
  
 System.*out*.println("SIMD 기법을 이용한 산술 연산 프로그램의 성능결과: " + String.*format*("%.6f",duration) + "초");  
 }  
}

1. **결과**

****

**감사합니다.**