Assignment 2

COMPUTER ORGANIZATION AND ARCHITECTURE

Group 14

1. 65015101 นายพนธกร พวงพะยอม 2. 65015137 นายวัฒชัย เตชะลือ

จากโจทย์

ให้แต่ละกลุ่มเขียนโปรแกรมเปรียบเทียบความเร็วในการทำงานระหว่างภาษา C กับ Assembly โดยใช้ฟังก์ชัน time จับเวลา โดยสร้างฟังก์ชันอะไรก็ได้ เช่น shi9 เพื่Oอคูณ 4 หรือ บวกเลข หรืออื่Onๆ

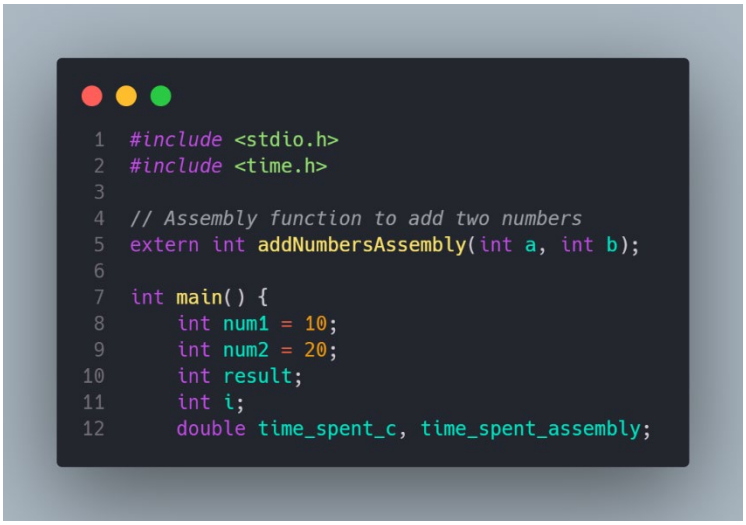
```
#include <stdio.h> #include <time.h>

int main ()

}

time t seconds;

seconds = time(NULL);

printf("Seconds since January 1, 1970 = %ld\n", seconds);

return(0);

}
```

เวลาที่ได้จะเป็นจำนวนวินาทีตั้งแต่ January 1, 1970 ดังนั้นต้องอ่านค่า 2 ครั้งมาลบกัน ครั้งแรกก่อนทำ และครั้งที่ 2 หลัง ทำ และเพื่อให้เกิดจำนวนเวลามากๆ ให้วนลูปทำหลาย ๆ ครั้งเช่น 1 ล้านรอบ การแสดงผลให้แสดงผล 2 ค่า คือ เวลาที่ใช้ เมื่อใช้ภาษา C และเวลาที่ใช้เมื่อใช้ภาษา Assembly

ตัวอย่างโคด

```c
#include <stdio.h>
#include <time.h>

// Assembly function to add two numbers
extern int addNumbersAssembly(int a, int b);

int main() {
    int num1 = 10;
    int num2 = 20;
    int result;
    int i;
    double time_spent_c, time_spent_assembly;

    // Test speed in C
    time_t begin_c = clock();
    for(i = 0; i < 100000000; i++) {
        result = num1 + num2;
    }
    time_t end_c = clock();
    time_spent_c = (double)(end_c - begin_c) / CLOCKS_PER_SEC;

    // Test speed in assembly
    time_t begin_assembly = clock();
    for(i = 0; i < 100000000; i++) {
        result = addNumbersAssembly(num1, num2);
    }
    time_t end_assembly = clock();
    time_spent_assembly = (double)(end_assembly - begin_assembly) / CLOCKS_PER_SEC;

    printf("Time Speed C: %f seconds\n", time_spent_c);
    printf("Time Speed Assembly: %f seconds\n", time_spent_assembly);

    return 0;
}
```

การทำงานของโคด



```c
#include <stdio.h>
#include <time.h>

// Assembly function to add two numbers
extern int addNumbersAssembly(int a, int b);

int main() {
    int num1 = 10;
    int num2 = 20;
    int result;
    int i;
    double time_spent_c, time_spent_assembly;
```

#include <stdio.h>

จะเก็บคำสั่งเกี่ยวกับการรับค่า แสดงค่าของโปรแกรม

#include <time.h>

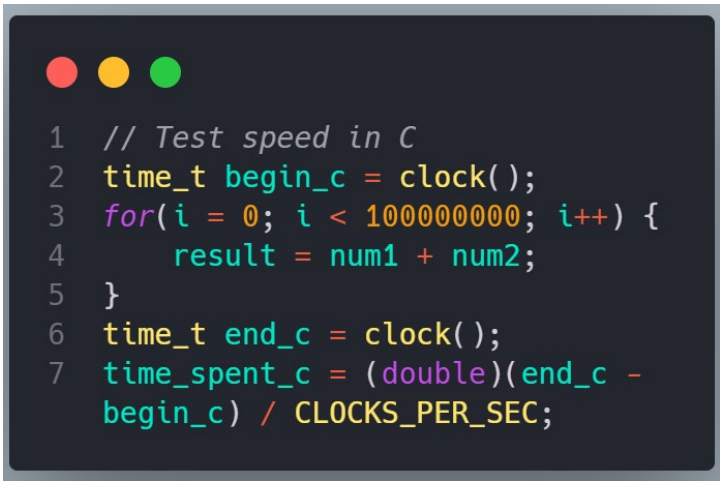
รวมตัวแปรและฟังก์ชันที่ใช้สำหรับทำงานและ

จัดการเกี่ยวกับเรื่องวันที่และเวลา (date & time)

จากนั้นสร้างตัวแปรต่างๆ ที่เราจะใช้งานในฟังก์ชัน

การทำงานภาษา C

เริ่มต้นตัวแปร begin_c ประเภท time_t และกำหนดค่าที่ส่งคืนโดยclock()ฟังก์ชัน ฟังก์ชัน นี้clock() ใช้เพื่อวัดเวลาตัวประมวลผลที่ใช้โดยโปรแกรม



```c
// Test speed in C
time_t begin_c = clock();
for(i = 0; i < 100000000; i++) {
    result = num1 + num2;
}
time_t end_c = clock();
time_spent_c = (double)(end_c -
begin_c) / CLOCKS_PER_SEC;
```

บรรทัด for loop ที่ดำเนินการเพิ่ม num1 + num2 เป็น 100,000,000 ครั้ง ผลลัพธ์ถูกกำหนดให้กับตัวแปร result กำหนดค่าที่ส่งคืนที่ ฟังก์ชันclock()

CLOCKS_PER_SEC ซึ่งเป็นค่าคงที่ที่แสดงจำนวนสัญญาณนาฬิกาต่อวินาที รหัสจะคำนวณเวลาที่ใช้ในหน่วยวินาที ผลลัพธ์จะถูกกำหนดให้ time_spent_c

โดยสรุป โค้ดนี้วัดเวลาดำเนินการของลูปที่ดำเนินการเพิ่ม num1 + num2เป็น 100,000,000 ครั้ง และเก็บเวลาที่ใช้เป็นวินาทีในฟังก์ชัน time_spent_c

บรรทัดแรก คือให้ begin_assembly ของ time_t
และกำหนดฟังก์ชัน clock()

```c
// Test speed in assembly
    time_t begin_assembly = clock();
    for(i = 0; i < 100000000; i++) {
        result = addNumbersAssembly(num1, num2);
    }
    time_t end_assembly = clock();
    time_spent_assembly = (double)(end_assembly
- begin_assembly)
        / CLOCKS_PER_SEC;

    printf("Time Speed C: %f seconds\n",
        time_spent_c);
    printf("Time Speed Assembly: %f seconds\n",
        time_spent_assembly);

```

เช่นเดียวกับข้อมูลโค้ดก่อนหน้า การวนซ้ำนี้ดำเนินการเพิ่ม
addNumbersAssembly(num1, num2)
จำนวน 100,000,000 ครั้ง

หลังจากที่ทำการลูปเสร็จส่งค่าคืนที่ฟังก์ชัน clock()
ให้กับตัวแปร end_assembly โดยจับเวลาตัวประมวลผล

จากนั้นในบรรทัดที่ time_spent_assembly = (double)(end_assembly - begin_assembly) / CLOCKS_PER_SEC; ก็คือ time_spent_assembly เป็นตัวแปรประเภท double และ end_assembly, begin_assembly แสดงถึงเวลาตัวประมวลผลที่ใช้โดยการใช้แอสเซมบลีหารความแตกต่างนี้ด้วย CLOCKS_PER_SEC หลังจากนั้นจะคำนวณเวลาที่ใช้ในหน่วยวินาทีและผลลัพธ์จะถูกกำหนดให้กับ time_spent_assembly และรองสองบรรทัดสุดท้ายก็ปริ้นค่าออกมา

และเราจะหาผลลัพธ์ได้ยังไงล่ะ?

โดยที่กลุ่มผู้เรียนได้เขียนไฟล์แยก โดยการให้คำสั่ง extern int addNumbersAssembly(int a, int b); ในบรรทัดต้นๆของภาษา C โดยคีย์เวิร์ด extern ใช้เพื่อประกาศไฟล์ โดยมีการกำหนดการใช้งานไฟล์แอสเซมบลีของฟังก์ชัน addNumbersAssembly ในไฟล์ต้นฉบับแอสเซมบลีที่แยกต่างหากครับ

```assembly
.text
.global addNumbersAssembly

addNumbersAssembly:
    push {lr}

    ldr r2, [sp, #15]
    ldr r3, [sp, #20]

    add r0, r2, r3

    pop {pc}
```

ผลการทำงานจะได้ดังนี้

```
Time Speed C: 0.008653 seconds
Time Speed Assembly: 0.012888 seconds
```

สรุปผลการทำงานในการเขียนโคด การใช้ภาษา C มีความเร็วในการทำงานมากว่า ภาษา Assembly