

编程测试

- 程序说明
 - 输入：依次输入计算机组成，计算机体系结构，计算机逻辑，汇编与接口，嵌入式系统课程的成绩
 - 输出：最终的成绩结果值
- 框架解析：

```
#define COURSE_NUM 5 // Course number
#define CPUT_ORG 0 // Computer organization
#define CPUT_ARC 1 // Computer architecture
#define CPUT_LOG 2 // Logic and design
#define CPUT_ASM 3 // Assembly
#define CPUT_EMB 4 // Embedded system
```

定义了各门课程的标号，方便程序理解。

```
struct course;
typedef struct course *Hardware;
struct course{
    float courses[COURSE_NUM];
    float grade;
};
```

通过数据结构储存硬件课程的成绩。

```
float getHWGrade(Hardware Courses)
{
    float sum = 0;
    float average;
    for(int i = 0; i < COURSE_NUM; i++){
        // If required courses is lower 60, skip the course.
        if(i == CPUT_ORG || i == CPUT_ARC || i == CPUT_LOG)
            if(Courses->courses[i] < 60)
                continue;
        switch(i){
            case CPUT_LOG:
            case CPUT_ARC:
            case CPUT_ORG:
                // For organization, logic, architecture, the weight is
1
                sum += Courses->courses[i];
                break;
            case CPUT_ASM:
                // For assembly, the weight is 0.9
                sum += 0.9 * Courses->courses[i];
                break;
            case CPUT_EMB:
                // For embedded system, the weight is 0.8
                sum += 0.8 * Courses->courses[i];
                break;
```

```

    }
}
// Compare the average and 0.6 * Organization
average = sum / COURSE_NUM;
return (average) > Courses->courses[CPUT_ORG] * 0.6 ? average :
Courses->courses[CPUT_ORG] * 0.6;
}

```

子程序的接口为硬件课程的成绩，为了方便程序理解，我把五门课程的成绩先行储存在结构中。

之后依次读入几门成绩，并进行要求的判断，最终得出结果。

- 程序测试：

```

70 80 90 100 90
80.40
请按任意键继续. . .
50 60 80 90 100
60.20
请按任意键继续. . .
50 50 50 30 40
30.00
请按任意键继续. . .

```

详细测试科见code文件夹下的测试程序。