# 实验一 数据表示与程序控制

## 题目 4 计算公司员工周工资

### 实验描述：

编写一个程序，输入某雇员的每周工作时间（以小时计）和每小时的工资数，计算并输出他的工资。若雇员周工作小时超过40 小时，则超过部分按原工资的1.5 倍的加班工资来计算。若雇员每周工作小时超过60 小时，则超过60 的部分按原工资的3 倍的加班工资来计算，而40 到50 小时的工资仍按照原工资的1.5 倍的加班工资来计算。

### 实验代码：

#include <iostream>

using namespace std;

int main()// Define the main function, which is the entry point of the program

{

    // Define three double variables to store the employee's work time, wage per hour, and total wage

    double work\_time, wage\_per\_hour, wage; // Ask the user to input the employee's work time and wage per hour, and store the inputs in the respective variables

    cout << "Please input employee's work time and wage\_per\_hour: " << endl;

cin >> work\_time >> wage\_per\_hour;

if (work\_time < 0 || work\_time > 24 \* 7 || wage\_per\_hour < 0)

    {

        cout << "Invalid input!" << endl;

            continue;

    }

    // If the employee worked 40 hours or less, calculate their wage as the product of work time and wage per hour

    if (work\_time <= 40)

        wage = work\_time \* wage\_per\_hour;

    // If the employee worked more than 40 hours but 60 hours or less, calculate their wage as the sum of 40 hours at regular wage plus the remaining hours at 1.5 times the regular wage

    else if (work\_time <= 60)

        wage = 40 \* wage\_per\_hour + (work\_time - 40) \* wage\_per\_hour \* 1.5;

    // If the employee worked more than 60 hours, calculate their wage as the sum of 40 hours at regular wage, 20 hours at 1.5 times the regular wage, and the remaining hours at 3 times the regular wage

    else

        wage = 40 \* wage\_per\_hour + 20 \* wage\_per\_hour \* 1.5 + (work\_time - 60) \* wage\_per\_hour \* 3;

    // Output the employee's wage to the console

    cout << "The employee's wage: " << wage << endl;

    return 0;

}

### 实验结果：

输入：30 4



输入：45 4.5



输入：60 5



### 思考与扩展：

1. 尝试分别用if 和switch 语句实现选择结构语句

将代码中间部分的if结构换成以下switch结构

int count;

    // If the employee worked more than 60 hours, set the count to 3, otherwise calculate the count as the number of 20-hour blocks the employee worked

    if (work\_time > 60)

        count = 3;

    else

        count = work\_time / 20;

    // Use a switch statement to calculate the employee's wage based on the count of the wage calculation scenario

    switch (count)

    {

    // If the count is 0 or 1, the employee worked less than 20 or between 20 and 40 hours, respectively

    case 0:case 1:

        // Calculate the wage as the product of work time and wage per hour

        wage = work\_time \* wage\_per\_hour;

        break;

    // If the count is 2, the employee worked between 40 and 60 hours

    case 2:

        // Calculate the wage as the sum of 40 hours at regular wage plus the remaining hours at 1.5 times the regular wage

        wage = ((work\_time - 40) \* 1.5 + 40) \* wage\_per\_hour;

        break;

    // If the count is 3, the employee worked more than 60 hours

    case 3:

        // Calculate the wage as the sum of 40 hours at regular wage, 20 hours at 1.5 times the regular wage, and the remaining hours at 3 times the regular wage

        wage = ((work\_time - 60) \* 3 + 20 \* 1.5 + 40) \* wage\_per\_hour;

        break;

    }

1. 如果程序运行时循环输入并给出计算结果，并当用户输入0 时程序结束，程序该如何修改？

将主体放入while循环体中，用if判断是否输入0

while (true)

    {

        cout << "Please input employee's work time and wage\_per\_hour: " << endl;

        cin >> work\_time >> wage\_per\_hour;

        if (work\_time == 0)

            break;

        // If the employee worked 40 hours or less, calculate their wage as the product of work time and wage per hour

        if (work\_time <= 40)

            wage = work\_time \* wage\_per\_hour;

        // If the employee worked more than 40 hours but 60 hours or less, calculate their wage as the sum of 40 hours at regular wage plus the remaining hours at 1.5 times the regular wage

        else if (work\_time <= 60)

            wage = 40 \* wage\_per\_hour + (work\_time - 40) \* wage\_per\_hour \* 1.5;

        // If the employee worked more than 60 hours, calculate their wage as the sum of 40 hours at regular wage, 20 hours at 1.5 times the regular wage, and the remaining hours at 3 times the regular wage

        else

            wage = 40 \* wage\_per\_hour + 20 \* wage\_per\_hour \* 1.5 + (work\_time - 60) \* wage\_per\_hour \* 3;

        cout << "The employee's wage: " << wage << endl;

    }

1. 测试数据的选择应该考虑选择结构中的每一种情况都被执行一次，也就是选择的测

试数据应该具有代表性，对本题来说，选择的数据因该分几种情况：

１）小于 40 的数据

２）大于 40 而小于50 的数据

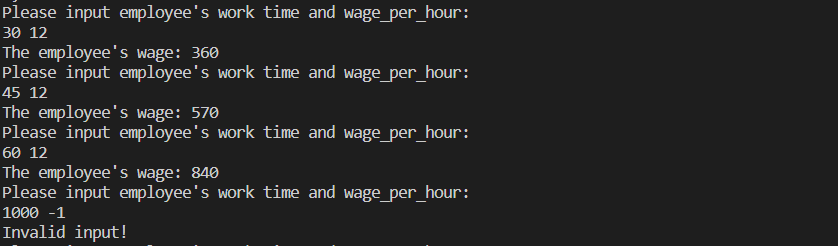
３）大于 50 的数据

４）不合理的数据比如负数或大于 7\*24 数，（注意一周最多只有7\*24 小时）

设计一组测试数据。

1. 30 12
2. 45 12
3. 60 12
4. 1000 -1

测试结果



## 题目 5 设计一个计算 + 、 - 、 \* 、 / 的程序

### 实验描述：

程序循环从标准输入读入表达式：

输入： 4+2 回车

则输出：4+2=6

输入： 4\*2 回车

则输出：4\*2=8

………

当用户输入：０ ０ ０ 回车时程序退出

### 实验代码：

#include <iostream>

#include <sstream>

using namespace std;

int main()

{

    while (true) // Use a while loop to repeatedly ask the user for an arithmetic expression until they enter "0 0 0"

    {

        // Prompt the user to input an arithmetic expression

        cout << "Please input arithmetic expression: "; // Declare a string variable to store the user's input

        string line;

        getline(cin, line); // Use getline to read a line of text from the console into the line variable

        if (line == "0 0 0")// If the user enters "0 0 0", exit the loop

            break;

        // Create an istringstream object initialized with the user's input to parse the expression

        istringstream iss(line);

        int a, b;

        char op;

        iss >> a >> op >> b;

        if (op == '/' && b == 0) // Check for division by zero, and if found, output an error message and continue the loop

        {

            cout << "Divided by zero!" << endl;

            continue;

        }

        // Declare an integer variable result to store the result of the arithmetic expression

        int result;

        // Use a switch statement to evaluate the operator and calculate the result

        switch (op)

        {

        case '+':

            result = a + b;

            break;

        case '-':

            result = a - b;

            break;

        case '\*':

            result = a \* b;

            break;

        case '/':

            result = a / b;

            break;

        default:

            cout << "Invalid operator: " << op << endl;

            continue;

        }

        cout << a << op << b << "=" << result << endl;

    }

    return 0;

}

### 实验结果：

输入：

1/0

1+1

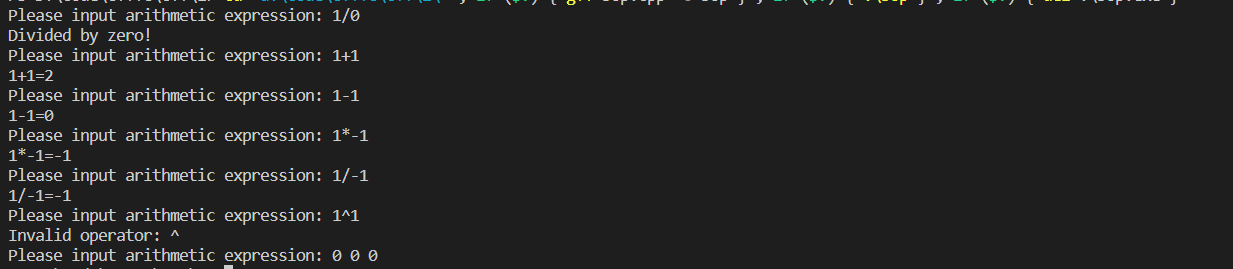
1-1

1\*-1

1/-1

1^1

0 0 0



### 思考与扩展：

1. 为什么要在while 循环体中输入三个数据（表达式）？

因为需要输入两个操作数和一个操作符。

1. 使用switch 语句中的break 起什么作用？
   1. switch中的break作用是结束switch循环，就是说不再执行下面的case语句。
   2. 如果不加的话，当一个case语句中执行完毕后，会进入下一个case语句，继续执行。