

实验目的：

- ✓ C 语句的分类
- ✓ 空语句与复合语句
- ✓ If 条件分支语句
- ✓ If 语句的嵌套
- ✓ Switch 语句及其使用

## 要点

### 1. 结构化程序设计

任何程序都可以用三种基本结构，就是顺序结构，选择结构和循环结构的组合来实现，这三种结构具有一个入口和一个出口的特征。结构化程序设计思想要求限制无条件转移语句的使用。

### 2. C 语句的分类

- 简单的顺序执行语句
  - 函数调用语句：函数调用加分号构成
  - 表达式语句：表达式加分号构成
  - 空语句：只有一个分号的语句
  - 复合语句：花括号括起来的语句
- 流程控制语句
  - 分支转移
    - ◆ If ( ) ~else~ 条件分支语句
    - ◆ Switch 多分支选择语句
    - ◆ Break continue return goto 无条件转移语句
  - 循环
    - ◆ While 当循环
    - ◆ Do while 直到型循环
    - ◆ For for 循环
- 标号语句
  - 上面的语句添加标号，构成标号语句

### 3. 空语句

空语句是仅仅只有一个分号的语句，不进行任何操作。如一下语句的作用是过滤掉输入字符流中的空白，回车和换行制表符，循环是一个空语句。

```
While((c=getchar())!=' '||c=='\n'||c=='\t');
```

### 4. 复合语句

程序中用花括号括起来的若干语句称为复合语句。一般

```
{  
    数据说明部分;  
    执行语句部分;  
}
```

- 在复合语句内部定义的变量，其作用域仅限于该复合语句的内部。

- 执行语句部分可以使用任何语句
- 复合语句在语法上等同普通的语句，凡是单一语句可以出现的位置都可以使用复合语句
- 复合语句可以嵌套

#### 5. 顺序结构

是由简单的顺序执行语句构成的语句块，块中的语句按照具体出现的先后顺序执行。

#### 6. If 语句的基本形式

- If (表达式) 语句一 else 语句二
- If (表达式) 语句一
- If 语句中的表达式可以是任意的形式，表达式的值非零就执行语句一，
- 12 可以是非常简单的语句，此时要注意不要漏掉结束标志，也可以是复合语句，要加大括号
- 语句一空的时候要用空语句来表示

#### 7. If 语句的嵌套

- If 语句可以相互嵌套
  - If (表达式 1) 语句 1  
Else if (表达式 2) 语句 2  
.....  
Else if (表达式 n) 语句 n  
Else 语句 n+1
  - If(表达式 1)  
If (表达式 2)  
If (表达式 3) 语句 1  
Else 语句 2  
Else 语句 3  
Else 语句 4
  - If(表达式 1)if(表达式 2)语句 1  
Else 语句 2
- If 语句嵌套使用时，需要特别注意 else 与 if 的配对问题。C 语法规则 else 总是与它前面的同一语法层次中最接近的尚未配对的 if 配对
- 进入 if 语句的时候，不管嵌套多么复杂，最多执行其中一个语句

#### 8. Switch

- Switch ( ) {  
Case 常量表达式 1: 语句 1  
Case 常量表达式 2: 语句 2  
.....  
Case 常量表达式 n: 语句 n  
[default:语句 n+1]  
}
- Switch 后面括号内的表达式必须是整数类型，也就是说可以是 int 型变量，char 型变量，也可以直接是整数或字符常量，负数也行，但不可以是实数，float，double，小数常量都会导致语法错误。
- 当 switch 后面的表达式与某个 case 后面的常量表达式相等时，就执行 case 后面的语句，执行完一个 case 后面的语句之后，流程控制转移到下一个 case 继续执行，

- 如果只想执行这一个 case，要在后面加；break；，跳出 switch
- 所有的 case 中的常量表达式的值都没有与 switch 后面括号内表达式的值相等，就执行 default 后面的语句，
- Case 的常量表达式只起到语句标号的作用，而不是进行判断，

#### 例题一

#### 实验代码

```
#include<stdio.h>
main(){
    int yy,mm,len;
    printf("year,month:");
    scanf("%d%d",&yy,&mm);
    switch(mm){
        case 1:
        case 3:
        case 5:
        case 7:
        case 8:
        case 10:
        case 12:len=31;break;
        case 4:
        case 6:
        case 9:
        case 11:len=30;break;
        case 2:if(yy%4==0 && yy%100!=0 || yy%400==0)
            len=29;
            else
            len=28;
            break;
        default:printf("input error");
        break;
    }
    printf("The length of %d.%d is %d\n",yy,mm,len);
}
```

#### 调试分析

```
#include<stdio.h>
main(){
    int yy,mm,len;
    printf("year,month:");
    scanf("%d%d",&yy,&mm);
    switch(mm){
        case 1:
        case 3:
```

```

case 5:
case 7:
case 8:
case 10:
case 12:len=31;break;
case 4:
case 6:
case 9:
case 11:len=30;break;
case 2:if(yy%4==0 && yy%100!=0 || yy%400==0)//这里的时候就因为!=打成=! 而

```

一直编译错误

```

        len=29;
    else
        len=28;
    break;
default:printf("input error");
break;
}
printf("The length of %d.%d is %d\n",yy,mm,len);
}

```

实验结果

```

year,month:1996 2
The length of 1996.2 is 29

```

实验总结:

符号的形式要记清楚

例题二

实验代码

```

#include<stdio.h>
#include<math.h>
main(){
    double a,b,c,disc,twoa,term1,term2;
    printf("enter a,b,c:");
    scanf("%lf%lf%lf",&a,&b,&c);
    if(a==0)
    if(b==0)
    printf("no answer due to input error\n");
    else
    printf("the single root is %f\n",-c/b);
    else{
        disc=b*b-4*a*c;
        twoa=2*a;
        term1=-b/twoa;
    }
}

```

```

        term2=sqrt(fabs(disc))/twoa;
        if(disc<0)
            printf("complex roots\n real part=%fimag part=%f\n",term1,term2);
        else
            printf("real roots\n root1=%f root2=%f\n",term1+term2,term1-term2);
    }
}

```

调试分析

Sqrt 一定要加上 math.h

实验结果

```

enter a,b,c:1 -1 -3
real roots
root1=2.302776 root2=-1.302776

```

```

enter a,b,c:1 1 -12
real roots
root1=3.000000 root2=-4.000000

```

```

enter a,b,c:1 -2 -4
real roots
root1=3.236068 root2=-1.236068

```

习题一

实验代码

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

```
#include<math.h>
```

```
main(){
```

```
    double a,b,c,disc,twoa,term1,term2,x2;
```

```
    printf("enter a,b,c:");
```

```
    scanf("%lf%lf%lf",&a,&b,&c);
```

```
    if(a==0)
```

```
    if(b==0)
```

```
        printf("no answer due to input error\n");
```

```
    else
```

```
        printf("the single root is %f\n",-c/b);
```

```
    else{
```

```
        disc=b*b-4*a*c;
```

```
        twoa=2*a;
```

```
        term1=-b/twoa;
```

```
        term2=sqrt(fabs(disc))/twoa;
```

```
        if(disc<0)
```

```
            printf("complex roots\n real part=%fimag part=%f\n",term1,term2);
```

```
        else if(disc==0)
```

```

        printf("real root\n %f\n",term1);
        else if(fabs(term1+term2)>fabs(term1-term2))
        {printf("real root\n %f \n",term1+term2);
        x2=c/(a*(term1+term2));
        printf("real root\n %f %f \n",term1-term2,x2);}
        else if(fabs(term1+term2)<fabs(term1-term2))
        {printf("real root\n %f \n",term1-term2);
        x2=c/(a*(term1-term2));
        printf("real root\n %f %f \n",term1+term2,x2);}
    }
}

```

调试分析:

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

```
#include<math.h>
```

```
main(){
```

```
    double a,b,c,disc,twoa,term1,term2,x2;
```

```
    printf("enter a,b,c:");
```

```
    scanf("%lf%lf%lf",&a,&b,&c);
```

```
    if(a==0)
```

```
    if(b==0)
```

```
    printf("no answer due to input error\n");
```

```
    else
```

```
    printf("the single root is %f\n",-c/b);
```

```
    else{
```

```
        disc=b*b-4*a*c;
```

```
        twoa=2*a;
```

```
        term1=-b/twoa;
```

```
        term2=sqrt(fabs(disc))/twoa;
```

```
        if(disc<0)
```

```
        printf("complex roots\n real part=%fimag part=%f\n",term1,term2);
```

```
        else if(disc==0)
```

```
        printf("real root\n %f\n",term1);
```

```
        else if(fabs(term1+term2)>fabs(term1-term2))
```

```
        {printf("real root\n %f \n",term1+term2);
```

```
        x2=c/(a*(term1+term2));
```

```
        printf("real root\n %f %f \n",term1-term2,x2);}
        else if(fabs(term1+term2)<fabs(term1-term2))
```

```
        {printf("real root\n %f \n",term1-term2);
        x2=c/(a*(term1-term2));
```

```
        printf("real root\n %f %f \n",term1+term2,x2);}//这里和上面的这里原本是没有加{}的,
```

所以在跳过 else if 下面的第一句以后又会接着执行第二句，所以如果只有一句可以不加分号，但有第二句就一定要加了。

```
    }
}
```

## 实验结果

```
enter a,b,c:1 1 -12
real root
-4.000000
real root
3.000000 3.000000

enter a,b,c:1 -2 -4
real root
3.236068
real root
-1.236068 -1.236068
```

## 第二题

### 实验代码

```
#include<stdio.h>
#include<math.h>
main(){
    double a,b,c,p,s;
    scanf("%lf%lf%lf",&a,&b,&c);
    p=(a+b+c)/2.0;
    printf("%lf %lf %lf %lf\n",a,b,c,p);
    if(a+b<=c && a+c<=b && b+c<=a)
        printf("不能构成三角形 (DATA ERROR) \n");
    else if(a==b==c){
        printf("等边三角形\n");
        s=sqrt(p*(p-a)*(p-b)*(p-c));
        printf("%f",s);}
    else if(a==b||a==c||b==c){
        printf("等腰三角形\n");
        s=sqrt(p*(p-a)*(p-b)*(p-c));
        printf("%f",s);}
    else if(a*a+b*b==c*c || a*a+c*c==b*b || b*b+c*c==a*a){
        printf("直角三角形\n");
        s=sqrt(p*(p-a)*(p-b)*(p-c));
        printf("%f",s);}
    else{
        printf("一般三角形\n");
        s=sqrt(p*(p-a)*(p-b)*(p-c));
        printf("%f",s);}
}
```

### 调试分析

我发现在输出输入格式中出现怎样离谱的错误都是不会报错的，所以以后要小心了。

实验结果

```
1.0 1.0 1.0
1.000000 1.000000 1.000000 1.500000
等边三角形
0.433013
```

```
1.0 1.0 1.5
1.000000 1.000000 1.500000 1.750000
等腰三角形
0.496078
```

```
6.0 8.0 10.0
6.000000 8.000000 10.000000 12.000000
直角三角形
24.000000
```

```
7.0 11.0 13.0
7.000000 11.000000 13.000000 15.500000
一般三角形
38.499188
```

实验总结

所以还是应该仔细弄明白计算机执行代码的顺序并且加以习惯。

习题三

实验代码

```
#include<stdio.h>
main(){
    double a,b;
    scanf("%lf",&a);
    if(a<3000.0)
        printf("个人所得税： 0");
    else if(a>3000.0 && a<3500.0)
    {
        b=(a-3000.0)*0.05;
        printf("个人所得税： %f",b);
    }
    else if(a>=3500.0 && a<4500.0)
    {
        b=25.0+(a-3500.0)*0.1;
        printf("个人所得税： %f",b);
    }
}
```

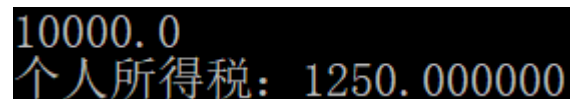


```

else if(a>=4500.0 && a<6000.0)
{
    b=125.0+(a-4500.0)*0.15;
    printf("个人所得税: %f",b);
}
else if(a>=6000.0 && a<8000.0)
{
    b=350.0+(a-6000.0)*0.2;
    printf("个人所得税: %f",b);
}
else if(a>=8000.0)
{
    b=750.0+(a-8000.0)*0.25;
    printf("个人所得税: %f",b);
}
}

```

实验结果



```

10000.0
个人所得税: 1250.000000

```

第四题

实验代码

```

#include<stdio.h>
main(){
    double x,f;
    printf("请输入自变量的值: ");
    scanf("%lf",&x);
    if(x<0 && x!=-3)
    {
        f=x*x+x-6;
        printf("x=%f f(x)=%f\n",x,f);
    }
    else if(x>=0 && x<10 && x!=2 && x!=3)
    {
        f=x*x-5*x+6;
        printf("x=%f f(x)=%f\n",x,f);
    }
    else
    {
        f=x*x-x-1;
        printf("x=%f f(x)=%f\n",x,f);
    }
}

```

```
}  
}
```

实验结果

```
请输入自变量的值: -5.0  
x=-5.000000 f(x)=14.000000
```

```
请输入自变量的值: 1.0  
x=1.000000 f(x)=2.000000
```

```
请输入自变量的值: -3.0  
x=-3.000000 f(x)=11.000000
```

```
请输入自变量的值: 2.0  
x=2.000000 f(x)=1.000000
```

```
请输入自变量的值: 2.5  
x=2.500000 f(x)=-0.250000
```

```
请输入自变量的值: 3.0  
x=3.000000 f(x)=5.000000
```

```
请输入自变量的值: 5.0  
x=5.000000 f(x)=6.000000
```

习题五

实验代码

```
#include<stdio.h>  
main() {  
    int a;  
    printf("请输入你的百分制成绩: ");  
    scanf("%d",&a);  
    if(a>100 || a<0)  
        printf("是错误的表达式, 请关闭重新输入");  
    else{  
        switch (a/10) {  
            case 10:  
            case 9 : printf("百分制成绩: %d 等级: A",a); break;  
            case 8 : printf("百分制成绩: %d 等级: B",a); break;  
            case 7 : printf("百分制成绩: %d 等级: C",a); break;  
            case 6 : printf("百分制成绩: %d 等级: D",a); break;  
            default : printf("百分制成绩: %d 等级: E",a); break;  
        }  
    }  
}
```

```
}  
}
```

## 实验结果

```
请输入你的百分制成绩: -90 请输入你的百分制成绩: 100 请输入你的百分制成绩: 90 请输入你的百分制成绩: 85  
是错误的表达式, 请关闭重新输入 百分制成绩: 100 等级: A 百分制成绩: 90 等级: A 百分制成绩: 85 等级: B
```

```
请输入你的百分制成绩: 70 请输入你的百分制成绩: 60 请输入你的百分制成绩: 45 请输入你的百分制成绩: 101  
百分制成绩: 70 等级: C 百分制成绩: 60 等级: D 百分制成绩: 45 等级: E 是错误的表达式, 请关闭重新输入
```

## 第六题

### 实验代码

```
#include<stdio.h>  
main() {  
    int y;  
    scanf("%d",&y);  
    if(y%4==0 && y%100!=0 || y%400==0)  
        printf("是闰年");  
    else  
        printf("不是闰年");  
}
```

### 实验结果

1880	2020	2000	1964	1892	1881
是闰年	是闰年	是闰年	是闰年	是闰年	不是闰年

### 实验总结

每次写逻辑运算符的时候要万分小心, 尤其是条件多想一次判断结束的, 写完以后一定要把所有可能性列出来检查, 而且要熟练掌握常用的组合逻辑运算符, 这里是第一个。

## 第七题

### 实验代码

```
#include<stdio.h>  
main() {  
    float x,y;  
    float z=50.8;  
    scanf("%f",&x);y=x/z;  
    if (x==0)printf("enter error!\n");  
    else printf("y=%f\n",y);  
}
```

### 实验结果

```
5.08  
y=0.100000
```

## 习题八

实验代码

```
#include<stdio.h>
main(){
    char a;
    scanf("%c",&a);
    if(a>96 && a<123)
    a=a-32;
    switch(a)
    {
        case 'A':
        case 'E':
        case 'I':
        case 'O':
        case 'U':
            printf("Yes. 大写: %d 小写: %d ",a,a+32);
            break;
        default :
            printf("No. 大写: %d 小写: %d ",a,a+32);
    }
}
```

实验结果

b	a
No. 大写: 66 小写: 98	Yes. 大写: 65 小写: 97

实验总结

有时间要好好掌握 ASCII 码