C语言：分支

1.分支中的if和switch

if语句：

基本语法

```
if(判断)
    表达式
```

也就是说当判断为真的时候程序就会执行if后面的内容。

例：

```
int mian()
{
    int x;
    scanf("%d ",&x);

    if (x % 2 == 0)
        printf("%d是偶数\n");

    return 0;
}
```

此代码判断了一个数是否为偶数。在此代码中 x%2==0 非常容易写成 x%=0。此处注意"="和 "=="的区别。

那么除了偶数还有奇数，可以将代码加上else改进一下：

```
int mian()
{
    int x;
    scanf("%d ",&x);

    if (x % 2 == 0)
        printf("%d是偶数\n");
    else
        printf("%d是奇数\n");
    return 0;
}
```

else后面的语句是当if的判断语句为假时执行

注意：if else语句后面默认都是只跟一条语句，如果有多条语句执行可以用"{}"将它们扩起来。

嵌套if

例：

```
int main()
{
    int a;
    scanf("%d",&a);

    if (a > 0)
        printf("正数\n");
    else if (a < 0)
        printf("负数\n");
    else
        printf("为零\n");
    return 0;
}
```

诸如此类当我们面对有很多情况需要判断时可以使用

```
if ()
表达式
else if()
表达式
else if()
表达式
else if()
表达式
else if()

.....

else
表达式
```

这样来进行判断。

悬空if else

```
int main()
{
    int a = 1;
    if (a > 2)
        if (a > 4)
            printf("wq不是大帅哥");
else
    printf("wq是大帅哥");
    return 0;
}
```

此代码的运行结果为void。因为在C语言中else 总是跟最接近的 if 匹配。终归来说是这段代码的格式问题，导致容易误判。

```
int main()
{
    int a = 1;
    if (a > 2)
        if (a > 4)
            printf("wq不是大帅哥");
        else
            printf("wq是大帅哥");
    return 0;
}
```

如果写成这样，就增加了可读性。

switch语句

基本语法

有没有发现在有多种情况时如果用多层if else太过麻烦。所以这里介绍了switch语句，可以说switch是if else的特殊形式

```
switch(expression)
{
case 1:statement;break;

case 2:statement;break;

case 3:statement;break;

...
```

```
default:statement;break;
}
```

switch中的break

switch语句根据expression中的值找到相对应的case并执行（break跳出），如果找不到，那么就执行default（然后跳出）

例：

```
int main()
{
    int a;
    scanf("%d",&a);
    a = a % 2;
    switch (a)
    {
    case 0:
        printf("偶数\n"); break;
    case 1:
        printf("奇数\n"); break;
    }
}
```

```
int main()
{
    int num;
    scanf("%d", &num);
    switch (num)
    {
    case 1:printf("星期一"); break;
    case 2:printf("星期二"); break;
    case 3:printf("星期三"); break;
    case 4:printf("星期四"); break;
    case 5:printf("星期五"); break;
    case 6:printf("星期六"); break;
    case 7:printf("星期天"); break;
    default:printf("输入错误! "); break;
    }
    return 0;
}
```

注：每个case语句写完后要加上break跳出语句。

如果去掉break，那么当一个case执行完后将会从上到下继续执行

如：

```
int main()
{
    int num;
    scanf("%d", &num);
    switch (num)
    {
    case 1:
    case 2:
    case 3:
    case 4:
    case 5:printf("上学"); break;
    case 6:
    case 7:printf("放假! "); break;
    default:printf("输入错误! "); break;
    }
    return 0;
}
```

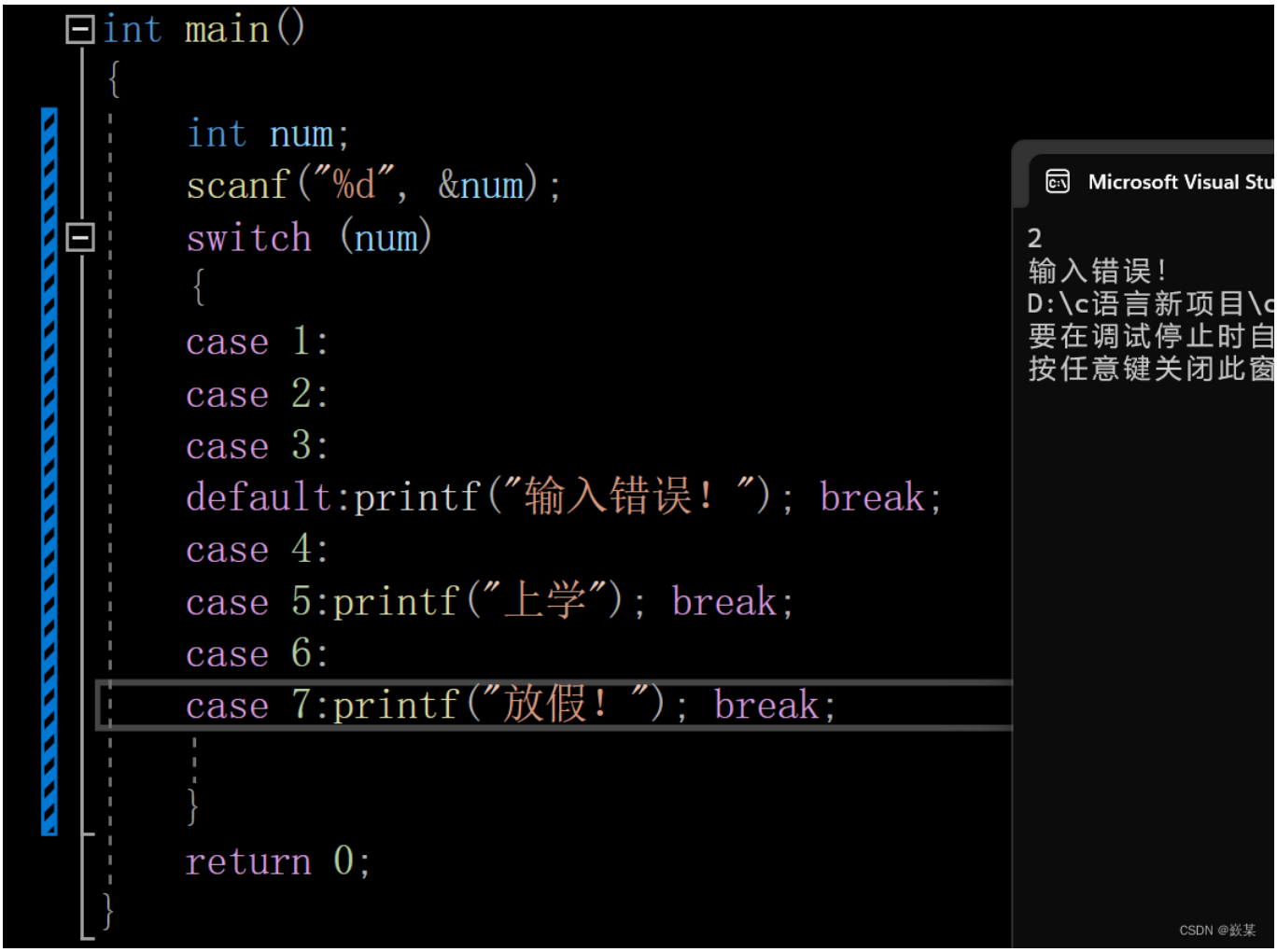
可以将上面的代码修改。如果输入1~5则输出 '上学'，输入6，7输出 '放假'。如果输入的数字不是在1~7那么输出 '输入错误'。

所以break不是每个case里都要有，要根据实际情况决定。

switch语句中的case和default的顺序问题

在switch中我们通常将case放在中间，将defautl放在开头或结尾。但最好不要将default放在中间。

否则因为case从上到下执行，就会出现此类情况：


int main()
{
int num;
scanf("%d", &num);
switch (num)
{
case 1:
case 2:
case 3:
default:printf("输入错误！"); break;
case 4:
case 5:printf("上学"); break;
case 6:
case 7:printf("放假！"); break;
}
return 0;
}
Microsoft Visual Stu
2
输入错误！
D:\c语言新项目\c
要在调试停止时自
按任意键关闭此窗
CSDN @嶽某

编辑