10.1-无条件跳转语句（break和continue）

最大用途：改变循环执行的状态

break：

主要功能：

（1）无条件跳出循环体，即无论循环条件是否成立，只要见到break，循环就被提前结束；

（2）无条件从switch语句中跳出；（8.3中讲过，不再赘述）

跳转位置：循环语句或switch语句的下一条语句。

continue：

主要功能：结束当前（本次）循环

跳转位置：跳过循环体中尚未执行的语句，直接判断下一次循环能否执行

代码示例1：

输入一些学生的体测成绩并求平均成绩，刷卡签到才有成绩且只要刷了卡，成绩必然是正数。若学生成绩为0，则说明该学生未参加测试，不纳入平均成绩的统计；当输入为负数时，表示完成输入，系统自动计算结果并输出。

int i,count=0,score=0,s=0;

for(i=0;i<1000;i++)

{

cin>>s;

if(s<0) //所有成绩已录入，循环无需再执行

{ break;}

if(s==0)

{ continue;}//该学生未参加测试，无需统计，跳过此次循环（判定i<1000?）

score+=s; //计算总分

count++; //统计学生数量

}

cout<<"共有"<<count<<"个学生参加测试"<<endl;

cout<<score/count<<endl;

总结：

break结束的是整个循环的过程，不再判断执行条件是否成立；

continue只是结束本次循环，即跳过循环体中尚未执行的语句，直接判断下一次循环能否执行

代码示例2：

输入正整数N（N<1000），输出N以内的全部质数,并输出N以内质数的个数。

int N,i,j,isprime,count=0;

cin>>N;

for(i=2;i<=N;i++)

{

isprime=1;

//sqrt()只支持浮点数，使用CSDN查询

for(j=sqrt(float(i));j>1;j--) //查找是否有因子

{

if(i%j==0)

{

isprime=0;

break; //当发现i有一个因子之后，就已经确定i不是素数了，那就不需要再继续寻找因子了，直接结束本次循环（结束的是哪个循环？）

}

}

if(!isprime) //isprime=0

{

continue; //不是素数直接跳过下面的判定，进行下一次循环能否执行的判定（i<=N？）

}

if(isprime/\*=1\*/)

{

cout<<i<<" ";

count++;

}

}

cout<<endl;

cout<<N<<"以内的质数有"<<count<<"个"<<endl;

总结：通过分析break和continue对哪个循环起作用，阐述如何辨别其控制的循环语句——多重循环中，break只能终止其所在的最小循环；continue只能结束其所在的最小循环的本次循环，跳转到其所在循环的下次循环条件的判定。