12.1-函数的声明、定义和调用

**函数的分类：**

1.标准库函数：

系统已经预定义，可以直接在程序中调用的函数；调用时应用include包含相关的头文件——

例如strcmp、strcat等函数，包含在头文件<string>或<cstring>中

sqrt、pow等函数包含在头文件<math.h>或<cmath>当中

2.用户自定义函数：

由用户自定义的函数与系统预定义的标准库函数的不同点在于，自定义函数的函数名、参数个数、函数返回值类型以及函数所实现的功能等都完全由用户程序来规定

再细分——**普通函数、内联函数**、类的成员函数、类的友元函数...

**函数的作用：**

将一段常用的代码进行“封装”，以减少程序中重复的代码段

（目前我们的所有程序几乎都写在主函数当中，而一个较大的程序，一般会被划分为若干个程序块，每个模块能够实现特定的功能）

当一个程序中包含多处功能相同的代码（处理数据的类型、处理过程相同或相似）我们一般会使用函数进行代码的“重用”，以实现程序的“简化”和“模块化”

**函数的定义：**

返回值类型 函数名(参数列表)

{

函数体(代码语句)

return 表达式

}

1.返回值类型：调用函数后反馈给主函数或其他函数的数据类型

2.函数名：起名，同“标识符”规则

3.参数列表：将数据从其他函数传入到函数体中

4.函数体：具体的代码语句

5.return语句：(表达式的值的类型与返回值类型相同)

函数的返回表示函数执行结束，将结果（无论是否有具体的数据）返回到调用函数的地方

返回值类型

void（无值型）、数值型、引用类型

代码实例1:

定义一个函数，使其能够计算两个浮点数相加的结果

确定返回值类型：浮点数相加的结果也是浮点型

参数列表：两个数相加，应该传入两个浮点型数的值

return语句：将结果传给其他函数

double add(double num1,double num2)

{

int sum=num1+num2;

return sum;

}

函数的调用：

使用已经定义好的函数

语法:函数名(参数)

参数要求：与函数定义时的参数表中的参数数目一致、类型相对（顺序不能乱）

函数调用的执行顺序：

1.根据调用语句中的函数名在整个程序中搜索同名函数定义；

2.核对实参数的参数个数，类型，顺序，判定是否与函数定义中的形参表对应一致

3.根据参数的类型（值参数或引用参数）进行值参数的值传递或引用参数的换名

运行函数体代码

4.返回调用点，并返回所要求的函数值

#include<iostream>

using namespace std;

double add(double num1,double num2)

{

double sum=num1+num2;

return sum;

}

int main()

{

double a,b;

cin>>a>>b;

cout<<a+b<<endl;

cout<<add(a,b)<<endl;

a=2.5;

b=3.6;

double m=add(a,b);

cout<<m<<endl;

return 0;

}

函数的声明:

返回值类型 函数名(参数列表);

函数的声明必须在调用之前，定义可以在调用之后



#include<iostream>

using namespace std;

double add(double num1,double num2); //num1和num2可省略

int main()

{

double a,b;

cin>>a>>b;

cout<<a+b<<endl;

cout<<add(a,b)<<endl;

a=2.5;

b=3.6;

double m=add(a,b);

cout<<m<<endl;

return 0;

}

double add(double num1,double num2)

{

double sum=num1+num2;

return sum;

}

代码示例2：

第一行输入一个整数n(n<=10)

接下来的n行，每行输入一个正整数x（x<=100），求x的平方和（1^2+2^2+...+x^2）

int n;

cin>>n;

int i,j,x;

int a[100];

int sum[10]={0};

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

{

cin>>x;

for(j=0;j<x;j++)

{

a[j]=(x-j)\*(x-j); //x^2+(x-1)^2+.....+1^2

sum[i]+=a[j]; //逐项累加

}

}

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

{

cout<<sum[i]<<endl;

}

**函数方法：**

#include<iostream>

using namespace std;

int square\_sum(int x)

{

int i;

int sum=0;

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

{

sum+=(x-i)\*(x-i);

}

return sum;

}

int main()

{

int n;

cin>>n;

int i,x;

int sum[10]={0};

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

{

cin>>x;

sum[i]=square\_sum(x);

}

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

{

cout<<sum[i]<<endl;

}

return 0;

}