


第7讲 函数与递归函数1



例1：求表达式 $\sqrt{1} + \sqrt{2} + \dots + \sqrt{10}$ 的值。

C++提供的各种标准函数： `sqrt(x)`

```
//sqrt(1)+...+sqrt(100);  
#include<iostream>  
#include<cmath>  
using namespace std;  
int main()  
{  
    double s=0;  
    for(int i=1;i<=100;i++)  
        s=s+sqrt(i);  
    cout<<s<<endl;  
    return 0;  
}
```

例2. 求: $10! + 8! + 6! + 12!$

1. 函数定义的语法形式

数据类型 函数名 (形式参数表)

{

函数体

//执行语句

}

2. 函数的调用


声明了函数原型之后，便可以按如下形式调用函数：

函数名 (实参列表) //例题中语句 `sum+=js(i);`

3. 函数的返回值

在组成函数体的各类语句中，值得注意的是返回语句 `return`。它的一般形式是：

`return (表达式);` // 例题中语句 `return s;`

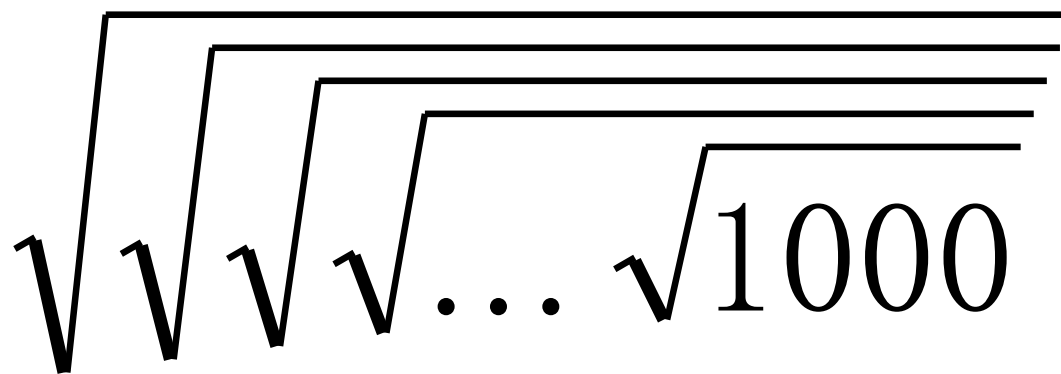


3. 输出1000以内的素数。

写一个判断素数的函数是返回1，不是返回0。

递归函数

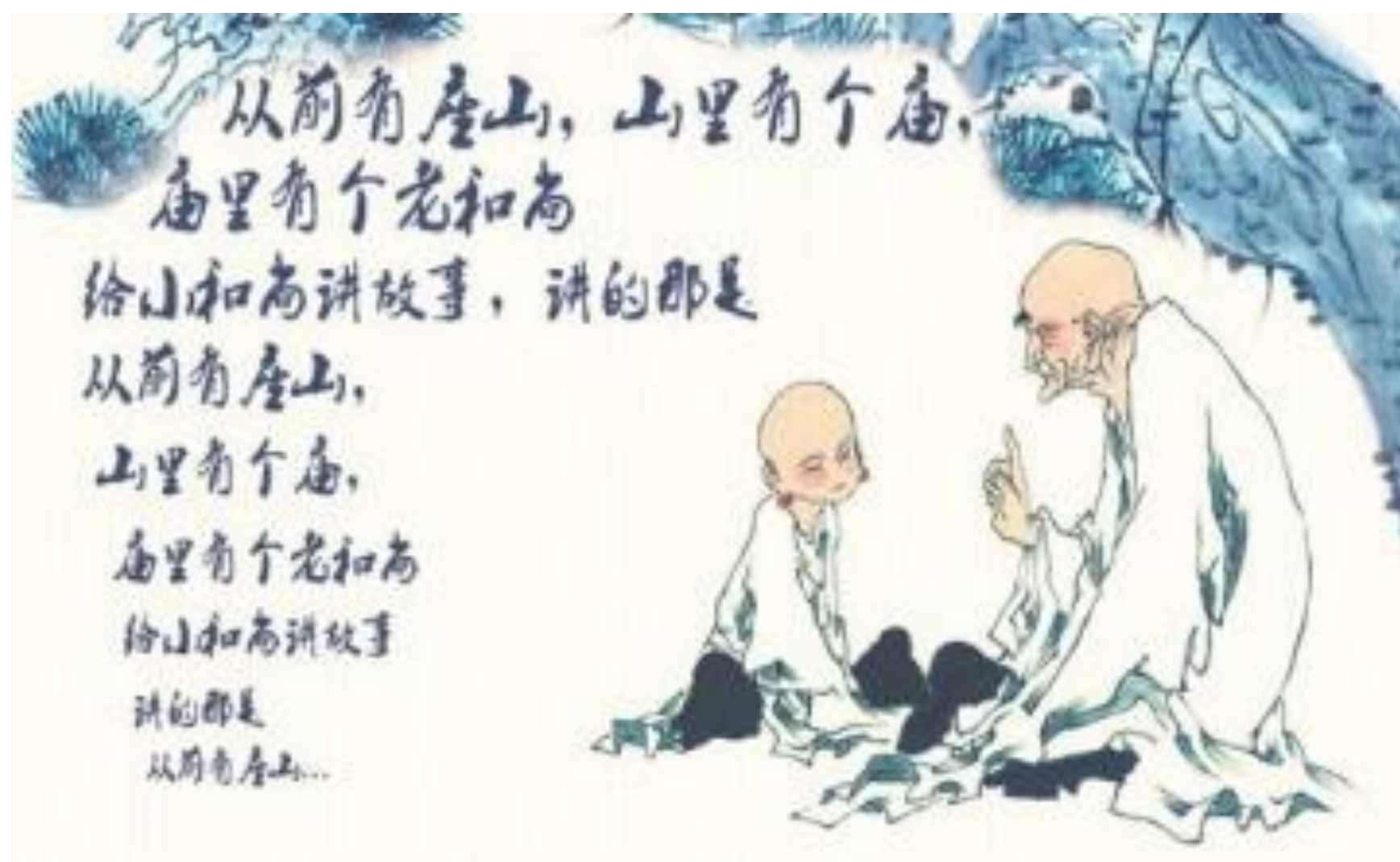
10层




The diagram illustrates a 10-layer nested square root function. It consists of ten square root symbols ($\sqrt{}$) arranged horizontally. The first four are explicitly drawn, followed by an ellipsis (\dots), and then the final square root symbol containing the number 1000. Above the sequence of square roots, there are ten horizontal lines that connect the top of each square root symbol to the top of the next one to its right, forming a series of parallel lines that represent the recursive calls in the function.

$$\sqrt{\sqrt{\sqrt{\sqrt{\dots \sqrt{1000}}}}}$$







"从前有座山山上有座庙,庙里有个老和尚,老和尚在给小和尚讲故事:

"从前有座山山上有座庙,庙里有个老和尚,老和尚在给小和尚讲故事:

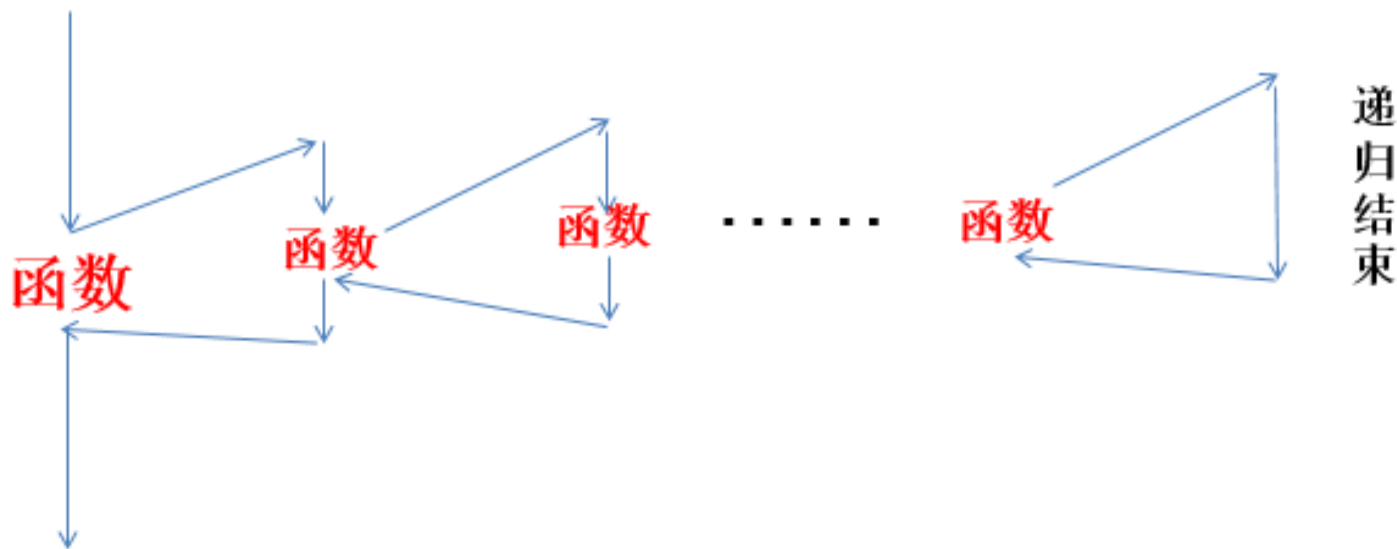
"从前有座山山上有座庙,庙里有个老和尚,老和尚在给小和尚讲故事:

.....

"从前有座山山上有座庙,庙里有个老和尚,老和尚在给小和尚讲故事:

在函数的定义中，内部又直接或间接的调用自己，这样的函数称为递归函数。

递归函数必须有终止条件，否则不停的进行下去，造成栈溢出



◆ 自定义函数

➤ 函数类型 函数名（参数） {
 • 函数体;
➤ }

函数类型:

Void: 无返回值。

Int: 有返回值

◆两种形式：

➤ 带有返回值的递归函数（调用结束返回需要的值）：

```
-int dfs(){  
    • dfs();  
-}
```

➤ 没有返回值的递归函数（完成某个功能）

```
-Void dfs(){  
    • dfs();  
-}
```

实现递归的关键：

- 1.确定递归公式（关系）
- 2.确定边界(终止)条件


当递归没有到达边界终止时，继续向前，直至边界才返回。

◆ 带有返回值的递归函数

➤ 递归函数结束后返回一个具体的数值。

◆例1：植树

- 植树节那天，有10位同学参加了植树活动，他们完成植树的棵数都不相同。问第一位同学植了多少棵时，他指着旁边的第二位同学说比他多植了两棵；追问第二位同学，他又说比第三位同学多植了两棵；如此追问，都说比另一位同学多植两棵，最后问到第10位同学时，他说自己植了5棵。
- 问第一位同学到底植了多少棵树？



➤ $f(x)=5; \quad (x=10)$

➤ $f(x)=f(x+1)+2 \quad (x<10)$

```
#include<iostream>
using namespace std;
int f(int x) {
    if(x==10) return 5;
    else return f(x+1)+2;
}
int main() {
    cout<<f(1)<<endl;
    return 0;
}
```



➤ 例2: 输入 n , 利用递归求 $n!=1*2*...*n$ 。

➤ $f(1)=1$

➤ $f(n)=n*f(n-1)$

```
#include<iostream>
using namespace std;
int f(int x) {
    if(x==1) return 1;
    return x*f(x-1);
}
int main() {
    int n;
    cin>>n;
    cout<<f(n)<<endl;
    return 0;
}
```