[7] 函数与结构体

深入浅出程序设计竞赛
第 1 部分 – 语言入门
V 2021-04

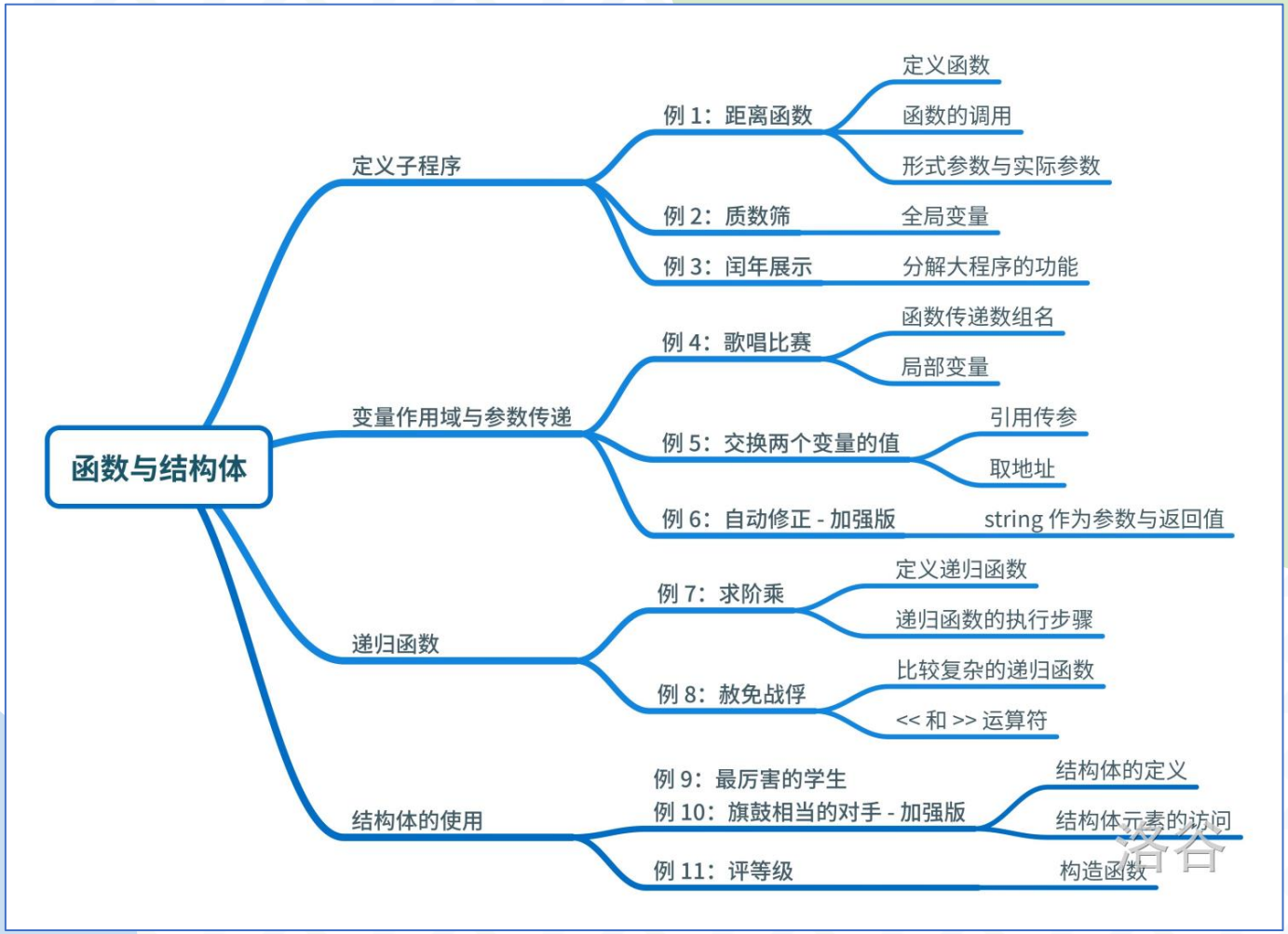
洛谷
www.luogu.com.cn

洛谷

版权声明

- 其它《深基》配套资源、购买本书等请参阅：https://www.luogu.com.cn/blog/kkksc03/IPC-resources
- 如果课件有任何错误，请在这里反馈 https://www.luogu.com.cn/discuss/show/296741

本章知识导图


函数与结构体
定义子程序
例 1：距离函数
定义函数
函数的调用
形式参数与实际参数
例 2：质数筛
全局变量
例 3：闰年展示
分解大程序的功能
变量作用域与参数传递
例 4：歌唱比赛
函数传递数组名
局部变量
例 5：交换两个变量的值
引用传参
取地址
例 6：自动修正 - 加强版
string 作为参数与返回值
递归函数
例 7：求阶乘
定义递归函数
递归函数的执行步骤
例 8：赦免战俘
比较复杂的递归函数
<< 和 >> 运算符
结构体的使用
例 9：最厉害的学生
结构体的定义
例 10：旗鼓相当的对手 - 加强版
结构体元素的访问
例 11：评等级
构造函数

第7章 函数与结构体

定义子程序

我可以自己造一个类似于 sqrt() 这样的函数吗？这样的话一些操作就可以反复的调用了。

请翻至课本 P99

距离函数

例 7.1（洛谷 P5735）

给出平面坐标上不在一条直线上三个点坐标 (x_1, y_1), (x_2, y_2), (x_3, y_3) ，坐标值是实数，绝对值不超过 100，求围成三角形周长。

分析：三个点，两两组成一条线段。分别计算并累加，得到答案。

平面上两点距离是 $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$。

```
#include <cstdio>
#include <cmath>
using namespace std;
int main() {
    double x1, y1, x2, y2, x3, y3, ans;
    scanf("%lf%lf%lf%lf%lf%lf", &x1, &y1, &x2, &y2, &x3, &y3);
    ans = sqrt((x2 - x1) * (x2 - x1) + (y2 - y1) * (y2 - y1));
    ans += sqrt((x3 - x2) * (x3 - x2) + (y3 - y2) * (y3 - y2));
    ans += sqrt((x3 - x1) * (x3 - x1) + (y3 - y1) * (y3 - y1));
    printf("%.2f", ans);
}
```

重复的的地方有点啰嗦？

距离函数

有没有办法减少重复的代码，简化程序的主干呢？

使用函数！改进后的代码如下：

```
#include <cstdio>
#include <cmath>
using namespace std;
double sq(double x) { // 计算平方
    return x * x;
}
double dist(double x1, double y1, double x2, double y2) { // 两点距离
    return sqrt(sq(x1 - x2) + sq(y1 - y2));
}
int main() {
    double x1, y1, x2, y2, x3, y3, ans;
    scanf("%lf%lf%lf%lf%lf%lf", &x1, &y1, &x2, &y2, &x3, &y3);
    ans = dist(x1, y1, x2, y2);
    ans += dist(x1, y1, x3, y3);
    ans += dist(x2, y2, x3, y3);
    printf("%.2f", ans);
}
```

函数

函数定义的一般形式如下：

```
返回类型 函数名(参数类型1 参数名1, ... ,参数类型n 参数名n) {
    函数体
    return 结果;
}
```

函数 1： sq()，喂进一个 double 类型的变量 x，经过计算后吐出一个 double 类型的结果。

函数 2： dist()，喂进 4 个 double 类型的变量 x1、y1、x2、y2，经过计算后吐出一个 double 类型的结果。

```
double sq(double x) { // 计算平方
    return x * x;
}

double dist(double x1, double y1, double x2, double y2) { // 两点距离
    return sqrt(sq(x1 - x2) + sq(y1 - y2));
}
```

因为引用的顺序，sq 函数应在 dist 函数前面定义，dist 函数应在 main 函数前面定义。

函数

当输入数据为 0 0 3 0 0 4，
计算 dist(x2, y2, x3, y3); 时
步骤是这样的：

```
double sq(double x) { return x*x; }

double dist(double x1, double y1,
            double x2, double y2) {
  return sqrt(sq(x1-x2) + sq(y1-y2));
}
```

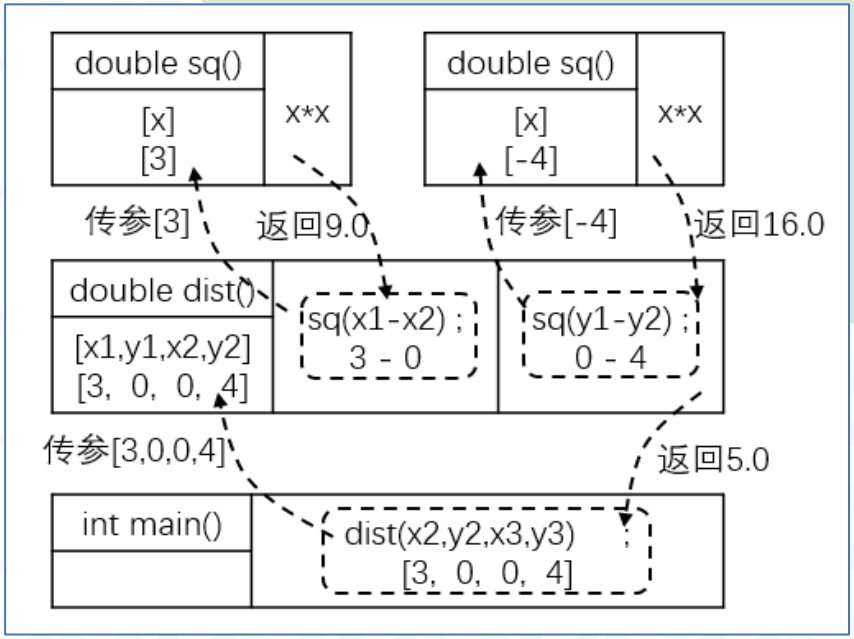
1. 进入 main 函数。收集了 4 个参数 [3,0,0,4]，调用 dist(3,0,0,4)。

2. 进入 dist 函数，x1=3，y1=0，x2=0，y2=4。收集到了参数 [3-0]，调用 sq(3)。

3. 进入 sq 函数，x=3。返回 9.0。

4. 回到 dist 函数，得到了 sq(x1-x2) = 9，同理 sq(y1-y2) =16。返回结果 5.0。

5. 回到 main 函数，得到了 dist(x2, y2, x3, y3) 的结果是 5.0。得到最终答案。



质数筛

例 7.2（洛谷 P5736）

输入 $n(n \leq 100)$ 个不大于 100000 的整数。要求全部储存在数组中，去除掉不是质数的数字，依次输出剩余的质数。

```
5
3 4 5 6 7
```

```
3 5 7
```

判断质数如何判断呢？

能不能编写一个函数，输入一个数字，就返回这个数字是不是一个质数呢？

质数筛

分析：我们之前介绍了如何判断质数——但可以把判断质数这一部分独立成一个函数，使主程序就更加清楚明。代码如下：

```
#include <iostream>
using namespace std;
int a[100], n;
bool is_prime(int x) {
    if (x == 0 || x == 1) return 0;
    //0和1都不是质数，需要特判
    for (int i = 2; i * i <= x; i++)
    //枚举1到sqrtx，来判断x是否为质数
        if (x % i == 0)
            return 0;
    return 1;//若是质数返回1，否则返回0
}
```

```
int main() {
    cin >> n;
    for (int i = 0; i < n; i++)
        cin >> a[i];
    for (int i = 0; i < n; i++)
        if (is_prime(a[i]))
            cout << a[i] << " ";
    cout << endl;
    return 0;
}
```

依次将 a[i] 喂给 is_prime()。进入函数后，喂进的数字就成为了函数内的 x 变量的值。根据质数判断条件，返回 1 或 0。

质数筛

这里的 a 数组和 n 变量定义在了主程序外面，是全局变量。

```
#include <iostream>
using namespace std;
int a[100], n;
bool is_prime(int x) {
    if (x == 0 || x == 1) return 0; //0和1都不是质数，需要特判
    for (int i = 2; i * i <= x; i++) //枚举1到sqrtx，来判断x是否为质数
        if (x % i == 0) return 0;
    return 1;//若是质数返回1，否则返回0
}
```

所有函数都可以访问这个变量（除非函数中定义了同名变量，或者函数的参数表中有这个变量名）。

全局变量会自动进行初始化操作，全部都会成为 0。

闰年展示

例 7.3（洛谷 5837）

输入 $x, y(1582 \leq x < y \leq 3000)$，输出 $[x, y]$ 区间中闰年个数，并在下一行输出所有闰年年份数字，使用空格隔开。

```
#include <iostream>
using namespace std;
int x, y, ans[500], cnt;
int main() {
    cin >> x >> y;
    for (int i = x; i <= y; i++)
        if (!(i % 400) || !(i % 4) && i % 100)
            // 之前章节介绍过的闰年判断
            ans[cnt++] = i; // 等同于ans[cnt]=i,cnt++;
    cout << cnt << endl;
    for (int i = 0; i < cnt; i++)
        cout << ans[i] << " ";
    cout << endl;
    return 0;
}
```

这个程序写起来很简单！

闰年展示

复杂的程序可分割成几个相对独立部分，便于编写与调试。

```cpp
#include <iostream>
using namespace std;
void init(); // 定义读入部分
void doit();// 定义处理部分
void output();// 定义输出部分

int x, y, ans[500], cnt;

int main() {
    init(); // 读入部分
    doit(); // 处理部分
    output(); // 输出部分
    return 0;
}
```

```cpp
void init() {
    cin >> x >> y;
}
void doit() {
    for (int i = x; i <= y; i++)
        if (!(i%400) || !(i%4) && i%100)
            ans[cnt++] = i;
}
void output() {
    cout << cnt << endl;
    for (int i = 0; i < cnt; i++)
        cout << ans[i] << " ";
    cout << endl;
}
```

将整个算法分解成读入部分、处理部分、输出部分。

函数可在使用前先定义，写清楚函数返回值、函数名和参数列表。

void 的意思是这个函数没有返回值。

变量作用域与参数传递

变量是怎么在函数之间传递的呢？如果遇到两个名称一样的变量，会不会引发冲突呢？

请翻至课本 P102

歌唱比赛

例 7.4（洛谷 P5738）

$n(n \leq 100)$ 名同学参加歌唱比赛，并接受 $m(m \leq 20)$ 名评委的评分，评分范围是 0 到 10 分。

这名同学的得分就是这些评委给分中去掉一个最高分，去掉一个最低分，剩下 $m-2$ 个评分的平均数。

请问得分最高的同学分数是多少？保留 2 位小数。

```
7 6
4 7 2 6 10 7
0 5 0 10 3 10
2 6 8 4 3 6
6 3 6 7 5 8
5 9 3 3 8 1
5 9 9 3 2 0
5 8 0 4 1 10
```

```
6.00
```

歌唱比赛

使用 stat 函数处理给定数组评分信息，然后更新最高分同学分数。

```
#include <cstdio>
#include <algorithm>
using namespace std;
int s[25], n, m, maxsum;
void stat(int a[], int m) {
  int maxscore = 0, minscore = 10, sum = 0;
  for (int i = 0; i < m; i++) {
    maxscore = max(a[i], maxscore); minscore = min(a[i], minscore);
    sum += a[i];
  }
  maxsum = max(maxsum, sum-maxscore-minscore); // 记录剩下的n-2评分总和
}
int main() {
  scanf("%d%d", &n, &m);
  for (int i = 0; i < n; i++) {
    for (int j = 0; j < m; j++) scanf("%d", &s[j]);
    stat(s, m);
  }
  printf("%.2f", double(maxsum) / (m - 2)); // 最后再计算平均值
  return 0;
}
```

歌唱比赛

stat 函数的参数 int a[] 指接收 int 类型的数组名。

传递数组名作为参数只是传递数组在内存中的地址。

数组 a[] 其实就是全局变量 s[] 的别名，如果在函数中改变 a[] 中的一项值，s[] 也会相应地改变。

建议大数组（超过1000个元素）定义为全局变量

```
int s[25], n, m, maxsum;
void stat(int a[], int m) {
  int maxscore=0, minscore=10, sum=0;
  for (int i = 0; i < m; i++) {
    maxscore = max(a[i], maxscore);
    minscore = min(a[i], minscore);
    sum += a[i];
  }
  maxsum = max(maxsum, sum-maxscore-minscore);
}
```

```
int main() {
  /* 省略 */ stat(s, m); /* 省略 */
  printf("%.2f",double(maxsum)/(m-2));
  return 0;
}
```

stat 函数中的变量 m、maxscore、sum 等在函数内部定义和使用的，被称为局部变量。函数里的 m 变量和外面的 m 没有关系。

交换两个变量的值

例 7.5

输入两个整数变量 a 和 b，设计一个交换函数将其交换后再输出。可不能直接输出 b 和 a 偷懒哦。

提问：下面的写法有什么问题？

```
#include <iostream>
using namespace std;
void swap(int x, int y) {
    int t = x;
    x = y;
    y = t;
}
```

```
int main() {
    int a, b;
    cin >> a >> b;
    swap(a, b);
    cout << a << " " << b << endl;
    return 0;
}
```

函数起不到作用！

交换两个变量的值

因为在 swap 函数中，x 和 y 都是局部变量，不会影响到外面了

经过改正的函数如下：

```
#include <iostream>
using namespace std;
void swap(int &x, int &y) {
    int t = x;
    x = y;
    y = t;
}
```

```
int main() {
    int a, b;
    cin >> a >> b;
    swap(a, b);
    cout << a << " " << b << endl;
    return 0;
}
```

在参数的变量名前面加上一个 & 符号，代表引用传参。

告知 swap 函数 a 和 b 的地址，故 x 是 a 别名，y 是 b 的别名，修改 x 和 y 的值，就会影响到 a 和 b。

回想一下 scanf 函数，如 scanf("%d",&n)，加上 & 是一样的道理。

递归函数

从前有座山，山里有座庙，庙里有个老和尚，老和尚给小和尚讲故事……

请翻至课本 P105

计算阶乘

例 7.7（洛谷 P5739）

求 $n!(n \leq 12)$，也就是 $1 \times 2 \times 3 \ldots \times n$。不允许使用循环语句。

分析：计算阶乘简单，按照之前的写法，你应该可以很快写出：

```
#include<iostream>
using namespace std;
int main() {
  int n, ans = 1;
  cin >> n;
  for(int i = 1; i <= n; i++)
    ans *= n;
  cout << ans << endl;
}
```

但不允许使用循环语句的话……

计算阶乘

假设 f(n) 代表 n 的阶乘，则阶乘可表示为 f(1)=1;f(n)=n*f(n-1)。

当想求 f(n) 时，就可以调用 f(n-1)
当想求 f(n-1) 时，就可以调用 f(n-2) …… 直到调用 f(1) 为止。

```
#include <iostream>
using namespace std;
int f(int x) {
    if (x == 1) return 1; //如果x为1，则返回1!=1
    return x * f(x - 1);
   //否则递归调用函数计算(x-1)!，并且将其乘上x返回，从而得到x!的结果
}
int main() {
    int n; cin >> n;
    cout << f(n) << endl; return 0;
}
```

像这样将规模大的问题转化成形式相同，但是规模更小的问题，被称为子问题。

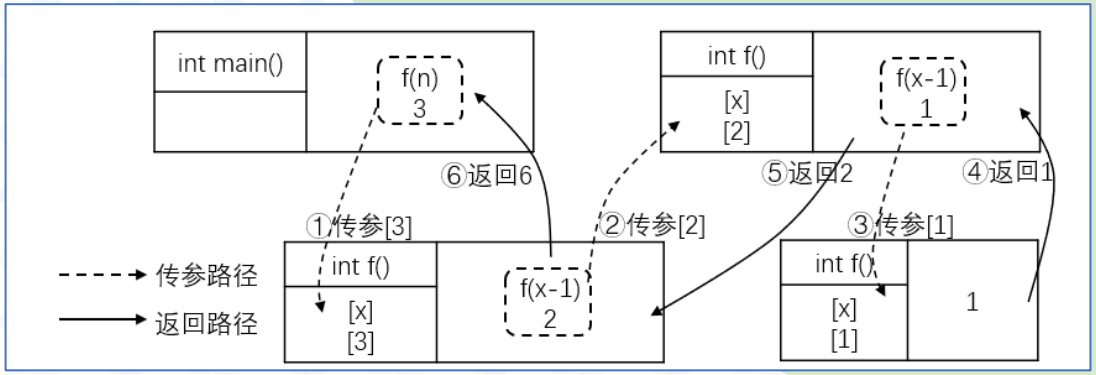
计算阶乘

一个函数不仅可以调用其他的函数，甚至还能调用自己，这种自己调用自己被称为递归。

```
cout << f(n) << endl;
```

```
int f(int x) {
  if (x == 1)
    return 1;
  return x * f(x - 1);
}
```

输入：3　输出：6



1. 进入 main 函数，cout << f(n) << endl; 调用 f(3)。

2. 进入第 1 层 f 函数，x=3。要求 f(3-1)，传递参数 [2] 给 f 函数，调用 f(2)。

3. 进入第 2 层 f 函数，x=2。要求 f(2-1)，传递参数 [1] 给 f 函数，调用 f(1)。

4. 进入第 3 层 f 函数，x=1，返回 1，回到第 2 层 f 函数，得到 f(2-1)=1，继续运算。

5. 返回结果 2*1=2，返回到第 1 层 f 函数，得到 f(3-2) 就是 2，继续进行运算。

6. 返回结果 3*2=6，返回到 main 函数，得到 f(3) 的值是 6，输出这个结果。

计算阶乘

虽然递归函数可自己调用自己，但是不能无限制调用下去，所以必须要设置递归终止条件。本例的递归终止条件是 f(1)=1。

请读者听听下面的故事，进一步了解递归。

从前有座山，山中有座庙，
庙里有个老和尚，老和尚在和小和尚讲故事：“
从前有座山，山中有座庙，
庙里有个老和尚，老和尚在和小和尚讲故事：“
从前有座山，山中有座庙，
庙里有个老和尚，老和尚在和小和尚讲故事：“
从前有座山，山中有座庙，
庙里有个老和尚，老和尚在和小和尚讲故事：“
太困了不讲了不讲了
”，于是都回去睡觉了。
”，于是都回去睡觉了。
”，于是都回去睡觉了。
”，于是都回去睡觉了。

到达终止条件！

```
void 讲故事() {
  if (困了) return;
  讲故事();
  回去睡觉;
}
```

可以抽象成这个样子！
同学们理解了吗？

递归函数有点类似于剥洋葱，一层套着一层，直到掰到最里层。

赦免战俘

例 7.8（洛谷 P5461）

现有 $2^n \times 2^n (n \leq 10)$ 名战俘站成正方形方阵。决定赦免一些俘虏。

将正方形矩阵均分 4 个更小的正方形矩阵，每个更小的矩阵的边长是原矩阵的一半。其中左上角那个矩阵的所有战俘都被赦免。

剩下 3 个小矩阵中，每一个矩阵继续分为 4 个更小的矩阵，然后通过同样方式赦免战俘……直到矩阵无法再分下去为止。

给出 n，请输出每名战俘的命运

其中 0 代表被赦免，1 代表不被赦免。

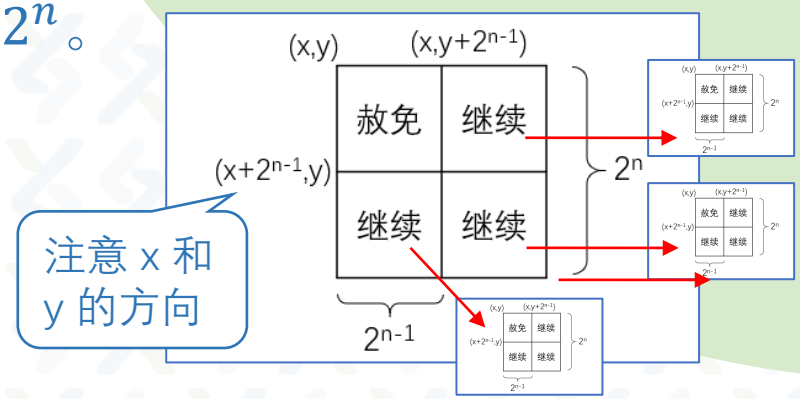
例如当 n=3 时，输出如右边所示：

```
0 0 0 0 0 0 0 1
0 0 0 0 0 0 1 1
0 0 0 0 0 1 0 1
0 0 0 0 1 1 1 1
0 0 0 1 0 0 0 1
0 0 1 1 0 0 1 1
0 1 0 1 0 1 0 1
1 1 1 1 1 1 1 1
```

赦免战俘

每次处理一个大矩阵，将这个矩阵分成四部分——左上角不处理，右上角、左下角、右下角矩阵使用相同方式处理。

函数 cal(x,y,n) 就是递归函数，x 和 y 表示正处理的矩阵左上角坐标 (x,y)；n 表示这个矩阵的边长是 2^n。



矩阵左上角坐标 (x,y)，边长 2^n，可求出剩下三个方块左上角坐标，并且进行下一层的处理。

当 n=0 时矩阵无法再分割，那么这个战俘不能赦免，杀了。

当 n 不是 0 的时候，矩形可分割，确定好剩下三个矩形的左上角坐标，继续递归执行下去。

赦免战俘

需要定义足够大的数组 a 记录每名战俘是否赦免，2^n 不超过 1024。

这个数组可以定义在函数外面作为全局变量，自动初始化为 0。

```cpp
#include <cstdio>
using namespace std;
int a[1050][1050], n;
void cal(int x, int y, int n) {
  if (n == 0) a[x][y] = 1;
  else {
    cal(x + (1 << n - 1), y, n - 1); // 右上方矩阵
    cal(x, y + (1 << n - 1), n - 1); // 左下角矩阵
    cal(x + (1 << n-1), y + (1 << n-1), n - 1); // 右下角
  }
}
int main() {
  int n; scanf("%d", &n); cal(0, 0, n);
  for (int i = 0; i < 1 << n; i++)
    for (int j = 0; j < 1 << n; j++)
      printf("%d%c", a[i][j], j == (1 << n) - 1 ? '\n' : ' ');
  return 0;
}
```

1<<n-1 就是 2^n，
<< 叫做左移运算符，
优先级很低

结构体的使用

为什么 string 类型使用起来这么方便呢？这么复杂的东西竟然可以直接赋值和拷贝等操作!

请翻至课本 P108

结构体

存储批量数据，比如说某位考生的信息，可以考虑使用数组。

但数组只能存同样数据类型的信息，不能同时存姓名、成绩等。

	a[x][0]	a[x][1]	a[x][2]
a[0]	118	129	135
a[1]	96	70	92
a[2]	124	148	146

←数组

结构体→

a[0]	a[1]	a[2]
no: 11451	no: 41919	no:81028
name: kkk	name: cz	name: bfqwq
age: 25.8	age: 17.2	age: 16.9
score: 114	score: 145	score: 81

而 string 类型字符串可存很多信息（字符数组数据、长度等），作为一个整体处理，可以赋值，可以作为参数直接传递给函数。

本节介绍的结构体可将一些不同类型的信息聚合成整体。

结构体

C++ 的结构体是由一系列有相同或不同类型的数据构成的数据集合。例如一名学生有姓名（字符串），有成绩（整数）。

结构体可以认为是一种变量类型，定义的一般形式如下：

```
struct 类型名 {
  数据类型1 成员变量1;
  数据类型2 成员变量2;
} [结构体变量名];
// 结构体变量名是可选的

struct 已经定义过的类型名 结构体变量名;
//"struct" 也可以不加
```

可以在定义结构体时同时定义变量，也可之后定义。

结构体可以定义为数组，可以整体赋值，也可以访问结构体成员。

最厉害的学生

例 7.9（洛谷 P5740）

有 $N(N \leq 1000)$ 名同学参加考试，获得了每名同学的信息：

姓名（不超过 8 个字符的字符串，没有空格）、语文、数学、英语成绩（均为不超过 150 的自然数）。

总分最高的学生就是最厉害的，请输出最厉害的学生各项信息（姓名、各科成绩）。

如果有多个总分相同的学生，输出靠前的那位。

```
3
senpai 114 51 4
lxl 114 10 23
fafa 51 42 60
```

```
senpai 114 51 4
```

最厉害的学生

学生信息使用结构体存储。可用于存储一名学生信息。

每次比较当前总分最大的答案，和枚举到的学生的总分。如果后者更大，就把当前学生的结构体赋值给答案的结构体，打擂台！

```
#include <iostream>
#include <string>
using namespace std;
struct student {
  string name;
  int chinese, math, english;   //开一个结构体记录每个学生的信息
} a, ans;
int main() {
  int n; cin >> n;
  for (int i = 1; i <= n; i++) {
    cin >> a.name >> a.chinese >> a.math >> a.english;
    if (a.chinese+a.math+a.english > ans.chinese+ans.math+ans.english)
      ans = a; //比较两个结构体的大小，如果这个更大那就用这个来更新答案
  }
  cout << ans.name<<" "<<ans.chinese<<" "<<ans.math<<" "<<ans.english<<endl;
  return 0;
}
```

a.name 可以访问 a 结构体变量的 name 成员

结构体变量可以直接整体赋值

旗鼓相当的对手 - 加强版

例 7.10（洛谷 P5741）

现有 $N(N \leq 1000)$ 名同学参加考试，知道每名同学的信息：

姓名（不超过 8 个字符的字符串，没有空格）、语文、数学、英语成绩（均为不超过 150 的自然数）。

如果某对学生 <i,j> 的每一科成绩的分差都不大于 5，且总分分差不大于 10，那么这对学生就是“旗鼓相当的对手”。

现在我们想知道这些同学中，哪些是“旗鼓相当的对手”？请输出他们的姓名和成绩。

```
3
senpai 114 51 4
lxl 114 10 23
fafa 51 42 60
```

```
senpai 114 51 4
```

旗鼓相当的对手 - 加强版

之前使用多维数组实现过类似的题目，但由于还要记录学生的名字，所以要使用结构体。

结构体也能批量定义，就跟数组一样，用于存储大量学生的信息。

```
struct student {
  string name; //开结构体记录学生信息
  int chinese, math, english;
};
struct student a[MAXN];
```

```
bool check(int x, int y, int z) {
  //检查两个数x,y的差是否不超过z
  return x <= y + z && y <= x + z;
}
```

```
for (int i = 1; i <= n; i++) //枚举第一个学生i
  for (int j = i + 1; j <= n; j++) //枚举第二个学生j
    if (check(a[i].chinese, a[j].chinese, 5)
      && check(a[i].math, a[j].math, 5)
      && check(a[i].english, a[j].english, 5)
      && check(a[i].chinese + a[i].math + a[i].english,
          a[j].chinese + a[j].math + a[j].english, 10)
        ){
        cout << a[i].name << " " << a[j].name << endl;
    }
```

结构体也能作为函数的参数或者返回值，请读者自己尝试实验。

祝贺大家

我们介绍完了入门部分的全部内容。

虽然只介绍了 C++ 的基本内容，但是已经足以应对大多数的算法竞赛所要求的语言知识了。

编程绝不是靠读书或听课就能搞懂的，必须要亲自上机实践。

课后习题非常重要，请尽力完成。

尽可能多的去完成洛谷“入门难度”的题。

课后习题与实验

学而时习之，不亦说乎。学而不思则罔，思而不学则殆。——孔子

请翻至课本 P95

课后拓展

习题 7.2

观察以下程序，那些变量是局部变量？哪些是全局变量？

如果输入数据是 1 2 时，应该输出什么呢？

```
#include <iostream>
#include <string>
using namespace std;
const int MAXN = 10000;
int a, b, c, array[MAXN];
bool check(int x, int y, int z) {
    return x <= y + z && y <= x + z;
}
void add(int a, int b) {
    c = a + b;
}
int multi() {
    return a * b;
}
```

```
int main() {
    int c = 3;
    cin >> a >> b;
    add(c, a);
    b = multi();
    if (check(a, b, c))
        cout << a;
    else
        cout << c;
    return 0;
}
```

课后拓展

习题 7.2

哪些变量是局部变量（黄）？哪些是全局变量（绿）？

如果输入数据是 1 2 时，应该输出什么呢？输出 1

```
#include <iostream>
#include <string>
using namespace std;
const int MAXN = 10000;
int a, b, c, array[MAXN];
bool check(int x, int y, int z) {
    return x <= y + z && y <= x + z;
}
void add(int a, int b) {
    c = a + b;
}
int multi() {
    return a * b;
}
```

```
int main() {
    int c = 3;
    cin >> a >> b;
    add(c, a);
    b = multi();
    if (check(a, b, c))
        cout << a;
    else
        cout << c;
    return 0;
}
```

总结

定义子程序

可以定义函数并且调用它，参数有实际参数和形式参数

可以将大程序拆成几个小的部分

变量作用域与参数传递

局部变量和全局变量，以及引用传参

递归函数

自己调用自己。可以将大问题分解成形式一样的小问题

结构体的使用

结构体可以集合不同数据类型为独立结构

也可以批量存储和作为函数的参数或者返回值

课后拓展

习题 7.1：设计以下的函数

1. 判断某个整数是不是完全平方数。
2. 给出三维空间的 2 对点，6 个变量，求出这两个点之间的距离。
3. 给出圆柱的半径和高，求出这个圆柱的体积。
4. 传入一个字符串，将这个字符串的所有空格去掉后返回处理好的字符串。
5. 给出一位 int 类型整数，计算它的每位数字的和。
6. 寻找指定数组中的平均数。

课后拓展

习题 7.3：质因数分解（洛谷 P1075，NOIP2012 普及组）

已知正整数 $n(n \leq 2 \times 10^9)$是两个不同的质数的乘积，试求出两者中较大的那个质数。

习题 7.4：单词覆盖还原（洛谷 P1321）

输入一个偶数 $N(N <= 10000)$，验证 4 到 N 所有偶数是否符合哥德巴赫猜想：任一大于 2 的偶数都可写成两个质数之和。如果一个数不止一种分法，则输出第一个加数相比其他分法最小的方案。

课后拓展

习题 7.5：回文质数（洛谷 P1217，USACO Training）

因为 151 既是一个质数又是一个回文数（从左到右和从右到左是看一样的），所以 151 是回文质数。写一个程序来找出范围 $[a,b](5 \leq a < b \leq 100{,}000{,}000)$(一亿)间的所有回文质数。

习题 7.6：集合求和（洛谷 P2415）

给定一个集合 S(|S|≤30)（即集合元素数量不超过 30），求出此集合所有子集元素之和。例如当集合是 {2,3} 时，子集包括 [] [2] [3] [2,3]，子集元素的和是 2+3+(2+3)=10。

课后拓展

习题 7.7：利用递归函数求解：

1. 不使用循环和数组，输入一串整数，然后将整数倒序输出。假设给出的整数串以 0 结尾。
2. 使用辗转相除法，求出两个给定的整数的最大公约数。
3. 求出斐波那契数列 fib(n) 的值，n≤10。更进一步，如果 n=20，你的递归函数可能会很慢，有什么办法改进效率吗？

习题 7.8：猴子吃桃（洛谷 P5743）

一只小猴买了若干个桃子。第一天他刚好吃了这些桃子的一半，又贪嘴多吃了一个；接下来的每一天它都会吃剩余的桃子的一半外加一个。第 n(n≤20) 天早上起来一看，只剩下 1 个桃子了。请问小猴买了几个桃子？

课后拓展

习题 7.9：培训（洛谷 P5744）

某培训机构的学员有如下信息：

1. 姓名（字符串）2. 年龄（周岁，整数）

3. 去年 NOIP 成绩（整数，且保证是 5 的倍数）

经过为期一年的培训，所有同学的成绩都有所提高，提升了 20%（当然 NOIP 满分是 600 分，不能超过这个得分）。

输入学员信息，请设计一个结构体储存这些学生信息，并设计一个函数模拟培训过程，其参数是这样的结构体类型，返回同样的结构体类型，并输出学员信息。

```
3
kkksc03 24 0
chen_zhe 14 400
nzhtl1477 18 590
```

```
kkksc03 25 0
chen_zhe 15 480
nzhtl1477 19 600
```

参考阅读材料

以下的内容限于课件篇幅未能详细阐述。如果学有余力，可自行翻阅课本作为扩展学习。

- P104 例 7.5： string 作为参数与返回值
- P110 例 7.10：结构体的成员函数和构造函数