

Application of Arrays

An array type describes a contiguously allocated nonempty set of objects with a particular member object type, called the element type. Array types are characterized by their element type and by the number of elements in the array. An array type is said to be derived from its element type, and if its element type is T, the array type is sometimes called “array of T”. The construction of an array type from an element type is called “array type derivation”.

C99, 6.2.5

Prototype原型

- Where do We Put the Function

```
void sum(int a, int b) {  
    sum = 0;  
    for ( int i=a; i<=b; i++ ) {  
        sum += i;  
    }  
    printf("%d\n", sum);  
}  
sum(1, 10);  
sum(20, 30);  
sum(35, 45);
```

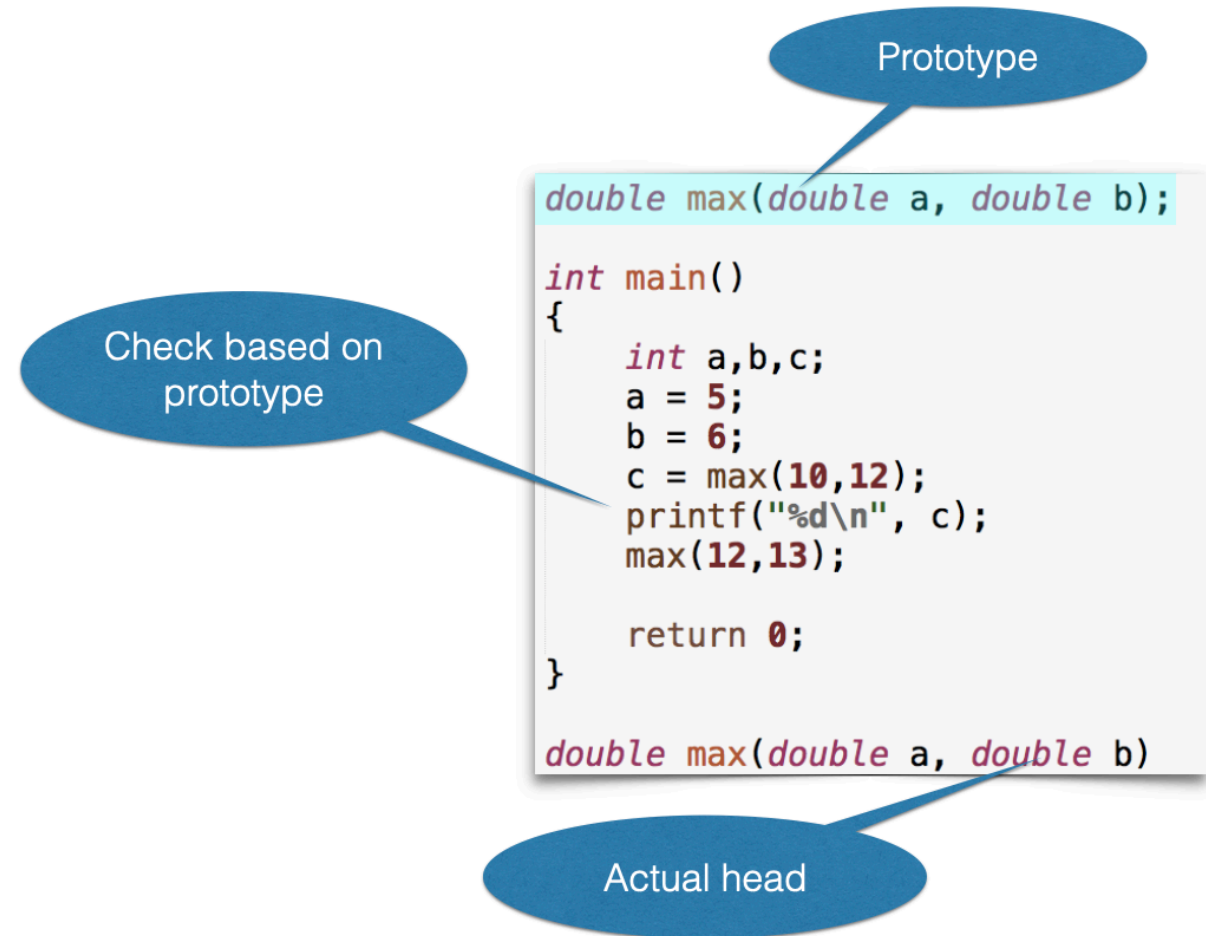
- The reason we have to put `sum()` before `main()` is:
 - The compiler parses code line by line
 - It needs to know how `sum()` looks like at the line `sum(1,10)`
 - How it looks like means the name, the parameter list and the return type
 - So it can check whether your calling is correct

If the compiler does not know

- How about we put the function after the `main()` ?
- It will assume all the parameters and return type are `int`
- What if that is not the case...
- In recent compilers, this is simply forbidden.

Prototype

- A function head, ending by `;`, is the prototype of that function
- It tells the compiler how the function looks like
 - name
 - parameter list (number and types)
 - return type
- It is usually somewhere before the function nowadays
- The names of the parameters are not required in the prototype



Lab 1

PTA 6-1 判断一个数字是否在一个正整数中

Calling a function

- If the called function has parameters, the calling must provide values of the right number and right types
- The values are expression, including:
 - literal
 - variable
 - return value
 - expression

What if type mismatch?

- It is the biggest leak in C that type may not match at calling
- The compiler always converts that quietly. But it may not the way you expected
- Later languages, C++/Java are more strict on it

What is to be passed?

- Can we swap a and b like this?

```
int swap(int a, int b) {  
    int t = a;  
    a = b;  
    b = t;  
}
```

- Only value can be passed in the C

Passing by value

- Each calling of a function has its own memory space, where parameters are in. That space is isolated from other functions
- At calling, the values provided are used to initialize the parameters. It is not the variables that are passed to
- In the past, the parameters in the function head are called “former parameter形式参数”, while the values at the calling are called “real parameter实际参数”
- These terms are easily to be misunderstood that it is the “real” be passed into the function to replace the “former” one, rather than the values. So we suggest do not call them like this.
- They are parameters and values.

Hash

Digits counting

Write a program to get input of arbitrary number of integers within $[0,9]$, count number of each digit, and output the numbers. A -1 is used to indicate the end of the input.

Least prefix

Given a non-empty array of N integers A , please find the smallest integer P such that all the numbers in A are in the subarray $A[0..P]$.

- Input
 - A list of N non-negative integers less than 1000000. N is no larger than 1000000.
- Output
 - The smallest integer P .

- Sample Input

2 2 1 0 1

- Sample Output

3

Lab 1

least prefix

Contains duplicate

Given an array of integers, find if the array contains any duplicates. Your function should return true if any value appears at least twice in the array, and it should return false if every element is distinct.

```
bool containsDuplicate(int nums[], int numsSize) {  
    bool ret = false;  
    for ( int i=0; i<numsSize; i++ ) {  
        for ( int j = i+1; j<numsSize; j++ ) {  
            if ( nums[i] == nums[j] ) {  
                ret = true;  
                goto RET;  
            }  
        }  
    }  
RET:  
    return ret;  
}
```

How to improve?

- The code above has a $O(n^2)$
- Sort
 - The best sort has a $O(n \cdot \log n)$
- Hash
 - The best hash has a $O(n)$

找画笔

豆豆对数字的执着，让他在理科领域游刃有余，但他近乎疯狂的投入也使父母有些担心，为了让孩子能够全面发展，决定拓宽他的学习领域，正好家旁边有个绘画培训中心就给豆豆报了名，学习绘画的第一天就让豆豆产生了浓厚的兴趣，还主动要求买了很多很多的画笔，画笔有多种颜色，豆豆有一个习惯就是同种颜色的画笔就买两支，一支备用，就这样总共攒了 N 支画笔(N 是偶数且 $1 < N < 10^6$)。

可是数字的敏感无孔不入，豆豆脑里蹦出了一个奇怪的问题：如果蒙上眼任意拿走一支画笔，分析剩下的 $N - 1$ 支画笔，能否找出拿走了哪种颜色？

Lab 2

PTA 7-2去掉重复的数据

Simple hash

- A hash function is any function that can be used to map data of arbitrary size to data of fixed size. The values returned by a hash function are called hash values, hash codes, hash sums, or simply hashes. One use is a data structure called a hash table, widely used in computer software for rapid data lookup.
- A simple hash function is to use the number itself as the hash value.

哈希排序

- 对数组的元素按从小到大进行排序。
- 输入有两行，第一行有一个整数 n ($5 \leq n \leq 200000$), 第二行有 n 个小于等于100000的整数。输出更新后的数组。

- 样例输入

```
8 1 2 3 8 7 4 5 10
```

- 样例输出

```
1 2 3 4 5 7 8 10
```

Lab 3

PTA 7-3 期末分数排序

Experiment from W3

- Write your code to calculate factorial
- Using `int`, what is the maxim $n!$ your code can get? How do you figure out the maxim `n`?
- The uplimit of `int` is $2^{31} - 1 = 2147483647$
- `int` 类型的数值范围有限，不能直接处理大整数

棋盘放米

- 发明chess的人把棋盘献给国王，国王很高兴，说要赏赐他。发明者提出的条件是：在棋盘的第一个格子放 1 粒米，第二个格子放 2 粒，第三个放 4 粒，依次每个格子翻倍。国王一口答应，可是很快发现，米粒数量大到难以想象，已经远远超过仓库能存放的数量
- 如何计算和表达 $1 + 2 + 4 + \dots + 2^{63}$
- The uplimit of `int` is $2^{31} - 1 = 2147483647$
- `int` 类型的数值范围有限，不能直接处理大整数
- 用 `int` 数组表达一个数的每一位数字，模仿人类的计算过程实现大数的计算

大数表示方法

- 用一个 `int` 数组来保存大数的每一位
- 用一个 `int` 表示它的位数
 - 用一个结构体把这两个数据放在一起
- 例如: `12345` 存为 `a[] = {5,4,3,2,1}`
- 低位在前存储
 - i. 便于对齐
 - ii. 计算是从低位向高位进行的

大数加法

```
  9876
+   789
-----
```

- 从个位开始逐位相加
- 保存进位 carry
- 直到最高位

Pseudocode

```
function add(a, b, c, len):  
    carry ← 0  
    对 i 从 0 到 len-1:  
        sum ← a[i] + b[i] + carry  
        c[i] ← sum % 10  
        carry ← sum / 10  
    如果 carry > 0:  
        c[len] ← carry
```

Quick check

请根据伪代码写出C语言程序

大数乘法思路

- 模仿竖式乘法：一个数的每一位依次与另一个数的每一位相乘
- 部分积存入结果数组并累加，注意处理进位
- $a[i] \times b[j] \rightarrow$ 加到结果 $c[i+j]$ 上

```
  123
×   45
-----
  615   (123×5)
 492    (123×4, 向左错一位)
-----
 5535
```

$$123 \times 5 = 615$$

$$123 \times 4 = 492, \text{ 左移一位} \rightarrow 4920$$

$$\text{结果} = 615 + 4920 = 5535$$

Pseudocode

```
function multiply(a, lena, b, lenb, c):  
    对 i 从 0 到 lena-1:  
        carry ← 0  
        对 j 从 0 到 lenb-1:  
            sum ← c[i+j] + a[i] * b[j] + carry  
            c[i+j] ← sum % 10  
            carry ← sum / 10  
        c[i+lenb] ← c[i+lenb] + carry
```

作业提示

设计一个完整的程序，用上述加法和乘法函数计算

$$\sum_{i=0}^{63} 2^i$$

Polynomial

In mathematics, a polynomial is an expression made of variables (indeterminates) and coefficients, combined only with addition, subtraction, multiplication, and nonnegative integer exponents. For example, $x^2 - 4x + 7$ is a polynomial in x .

Polynomials appear widely in mathematics and science. They are used to form equations that model many problems, to define functions in physics, chemistry, and economics, and in calculus and numerical analysis to approximate other functions. In higher mathematics, they also lead to concepts such as polynomial rings and algebraic varieties.

Polynomial equation

A polynomial in a single indeterminate x can always be written (or rewritten) in the form

$$a_n x^n + a_{n-1} x^{n-1} + \cdots + a_2 x^2 + a_1 x + a_0,$$

where a_0, \dots, a_n are constants that are called the coefficients of the polynomial, and x is the indeterminate. The word "indeterminate" means that x represents no particular value, although any value may be substituted for it. The mapping that associates the result of this substitution to the substituted value is a function, called a polynomial function.

This can be expressed more concisely by using summation notation:

$$\sum_{k=0}^n a_k x^k$$

Polynomial evaluation

- Given a polynomial, such as $f(x) = 2x^2 + 3x - 2$, calculate $f(x)$ given different values of x .
- Given the number of terms (times) of the polynomial and the coefficients of each term (time), and the x value, find the value of the polynomial

5

3

2 3 -2

(In order, this means that x has a value of 5, a polynomial of the 3rd power, and coefficients of the 2nd, 1st, and 0th powers)

- It makes sense to store it in an array

Expressing polynomials with array

```
double x;  
int N;  
scanf("%lf %d", &x, &N);  
double poly[N];  
for ( int i=0; i<N; i++ ) {  
    scanf("%lf", &poly[i]);  
}
```

Iterating the array to compute the polynomial

```
double sum = 0;
int power = N-1;
for ( int i =0; i<N; i++ ) {
    double k = 1;
    for ( int j=0; j<power; j++ ) {
        k *= x;
    }
    sum += k*poly[i];
    power--;
}
```

- The x^n, x^{n-1}, \dots, x^0 were computed sequentially

How to speed it up?

- How about simply do x^n once and do $\frac{k}{x}$ each round of the loop

How to speed it up?

Take an n th degree polynomial $f(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0$ rewrite it in the following form:

$$\begin{aligned}
 f(x) &= a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 \\
 &= (a_n x^{n-1} + a_{n-1} x^{n-2} + \dots + a_2 x + a_1) x + a_0 \\
 &= (a_n x^{n-2} + a_{n-1} x^{n-3} + \dots + a_3 x + a_2) x + a_1) x + a_0 \\
 &= (\dots ((a_n x + a_{n-1}) x + a_{n-2}) x + \dots + a_1) x + a_0 \\
 &= (\dots (((0x + a_n) x + a_{n-1}) x + a_{n-2}) x + \dots + a_1) x + a_0
 \end{aligned}$$

Algorithm

1. Make the result `sum=0`

2. Iterate over all coefficients `a` from the highest order

○ `sum += sum*x+a`

```
double sum = 0;
for ( int i=0; i<N; i++ ) {
    sum = sum*x;
    sum += poly[i];
}
```

- In this way, finding the value of the n th degree polynomial $f(x)$ is transformed into finding the value of n primary polynomials

秦九韶算法

- 该算法最早出现在汉朝（公元前202到公元220年），刘徽所注的《九章算术》中
- 宋淳祐四至七年（公元1244至1247），秦九韶在湖州为母亲守孝三年期间，把长期积累的数学知识和研究所得加以编辑，写成了举世闻名的数学巨著《数书九章》。书成后，并未出版。原稿几乎流失，书名也不确切。后历经宋、元，到明初，此书无人问津，直到明永乐年间，在解缙主编《永乐大典》时，记书名为《数学九章》。又经过一百多年，经王应麟抄录后，由王修改为《数书九章》
- 19世纪初，英国数学家威廉·乔治·霍纳重新“发现”并证明，西方称作霍纳算法（Horner's method、Horner scheme）。但是，19世纪英国传教士伟烈亚力 Alexander Wylie. (1815–1887) 最早对霍纳的发明权提出质疑，霍纳的算法程序和数字处理都远不及五百多年前的秦九韶有条理；秦九韶算法不仅在时间上早于霍纳，也比较成熟

Convolution of Vectors (向量的卷积)

The convolution of two vectors, u and v , represents the area of overlap(重叠) under the points as v slides(滑动) across u . Algebraically(代数上), convolution is the same operation as multiplying polynomials(多项式) whose coefficients(系数) are the elements of u and v . Let $m = \text{length}(u)$ and $n = \text{length}(v)$. Then w is a vector of length $m + n - 1$ whose k^{th} element is

$$w(k) = \sum_j u(j)v(k - j + 1)$$

The sum is over all the values of j that lead to legal subscripts(下标) for $u(j)$ and $v(k - j + 1)$. When $m = n$, this gives

$$w(1) = u(1) * v(1)$$

$$w(2) = u(1) * v(2) + u(2) * v(1)$$

$$w(3) = u(1) * v(3) + u(2) * v(2) + u(3) * v(1)$$

...

$$w(n) = u(1) * v(n) + u(2) * v(n - 1) + \dots + u(n) * v(1)$$

...

$$w(2 * n - 1) = u(n) * v(n)$$

Now a program is needed to reads two vectors in and prints the convolution of these two vectors.

Input Format:

The first line is a natural number M , stands for the dimension(维度) of the first vector.

After that, a line of M real numbers are the values of the first vector.

The third line is a natural number N , stands for the dimension of the second vector.

After that, a line of N real numbers are the values of the second vector.

$(M, N \leq 100)$

Output Format:

The $M + N - 1$ values of elements of the result vector. Each number has two figures for the decimal part. There should be only one space after each number.

Input Example:

```
3
1 1 1
7
1 1 0 0 0 1 1
```

Output Example:

```
1.00 2.00 2.00 1.00 0.00 1.00 2.00 2.00 1.00
```

查字典

- 在字典里，每个汉字都对应一个拼音，而字典是按照拼音字母顺序排列的。假设我们需要查找一个拼音首字母为 r 的字，通常会：
 - i. 翻开字典约一半的页数，查看该页的首字母是什么，假设正好翻到首字母为 m ；
 - ii. 由于在拼音字母表中 r 位于 m 之后，所以排除字典前半部分，查找范围缩小到后半部分；
 - iii. 不断重复步骤 i 和步骤 ii，直至找到拼音首字母为 r 的页码为止
- 这就是二分算法

理牌

- 先把牌尽量打乱，然后按顺序发给四位玩家。我们首先要把发到的牌整理好，一般的做法是：
 - i. 拿起一张牌，在已经理好的牌里找到它合适的位置；
 - ii. 将这张牌插入在合适的位置；
 - iii. 不断重复步骤 i 和步骤 ii，直到所有的牌都已经放好了位置
- 这就是插入排序算法

算法

- 算法是在有限步骤内解决特定问题的有效方法，它是由一系列明确的、可执行的指令组成的。一个算法通常具有以下特点：
 - 输入：算法有零个或多个输入；
 - 输出：算法至少有一个输出；
 - 有限性：算法必须在有限步骤内结束；
 - 确定性：每一步的含义都必须明确，没有歧义；
 - 有效性：每一步都可以在有限时间内完成。

数据结构

- 数据结构是带有一定逻辑关系的数据元素的集合，以及对这些数据进行操作的一组方法。它回答两个问题：
 - i. 数据是如何存放和组织的
 - ii. 我们如何在这些数据上进行操作（查找、插入、删除等）
- 常见的数据结构有：
 - 线性结构：数组（array）、链表（linked list）、栈（stack）、队列（queue）
 - 非线性结构：树（tree）、图（graph）、哈希表（hash table）

What we've learned today?

- Prototype of a function
- Simple Hash
- 大数算法
- 秦九韶算法
- 算法与数据结构的意义