# qsort（C库函数）

## 描述

C 库函数 **void qsort(void \*base, size\_t nitems, size\_t size, int (\*compare)(const void \*, const void \*)),** 对数组进行排序。二分法，时间复杂度为：O(n\*logn).不稳定排序。

## 声明

Void qsort(void \*base, size\_t nitems, size\_t size, int (\*compare)(const void \*, const void \*))

## 参数

**base** -- 指向要排序的数组的第一个元素的指针，类型强制转换为 void\* 指针。

**nitems** – 由base指向的数组中元素的个数。

**size** – 数组中每个元素的大小，以字节为单位。

**compare** – 用来比较两个元素的函数。如果compare返回值小于0，第一个元素会排在第二个元素之前，如果compare返回值大于0，第一个元素会排在第二个元素后面，如果两个值相等，则不能确定。（其中有两个元素做参数，参数列表是两个空指针，在使用过程中，要指向不同类型的元素，需要转换为相应的类型。

## 返回值

无返回值，base数组并未声明为const 故直接改变base值。

## 实例

1./\*对int型数组排序\*/

int num[100];

int cmp\_int(const void \*\_a, const void \*\_b){

int \*a = (int \*)\_a;

int \*b = (int \*)\_b;

return \*a - \*b;

}

qsort(num, 100, sizeof(num[0]), cmp\_int);

2./\*对char型数组排序\*/

int word[100];

int cmp\_word(const void \*\_a, const void \*\_b){

char \*a = (char \*)\_a;

char \*b = (char \*)\_b;

return \*a - \*b;

}

qsort(word, 100, sizeof(word[0]), cmp\_char);

3./\*对double型数组排序\*/

double in[100];

int cmp\_double(const void\* \_a , const void\* \_b)　　//参数格式固定

{

double\* a = (double\*)\_a; //强制类型转换

double\* b = (double\*)\_b;

return \*a > \*b ? 1 : -1;　 //特别注意

//在对浮点或者double型的一定要用三目运算符

//因为要是使用像整型那样相减的话，如果是两个很接近的数则可能返回一个很小的小数（大于-1，小于1）

//而cmp的返回值是int型，因此会将这个小数返回0，系统认为是相等，失去了本来存在的大小关系

}

qsort(in,100,sizeof(in[0]),cmp\_double);

4./\*对字符串进行排序\*/

char word[100][10];

int cmp\_string(const void\* \_a , const void\* \_b)　　//参数格式固定

{

char\* a = (char\*)\_a;　　//强制类型转换

char\* b = (char\*)\_b;

return strcmp(a,b);

}

qsort(word,100,sizeof(word[0]),cmp\_string);

# rand（C库函数）

## 描述

C 库函数 int rand(void) 返回一个范围在 0 到 RAND\_MAX 之间的伪随机数。RAND\_MAX 是一个常量，它的默认值在不同的实现中会有所不同，但是值至少是 32767。

## 声明

int rand(void)

## 参数

**NA**

## 返回值

该函数返回一个范围在 0 到 RAND\_MAX 之间的整数值。

## 实例

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

int main()

{

int i, n;

time\_t t;

n = 5;

/\* 初始化随机数发生器 \*/

srand((unsigned) time(&t));

/\* 输出 0 到 49 之间的 5 个随机数 \*/

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

printf("%d\n", rand() % 50);

}

return(0);

}0