

太戈编程
etiger.vip

信奥算法

函数复习

自定义函数

自己定义函数功能

参数+函数体+返回值

绝对值函数

函数
返回值
类型

函数
名称

参数
列表

int abs(int y)

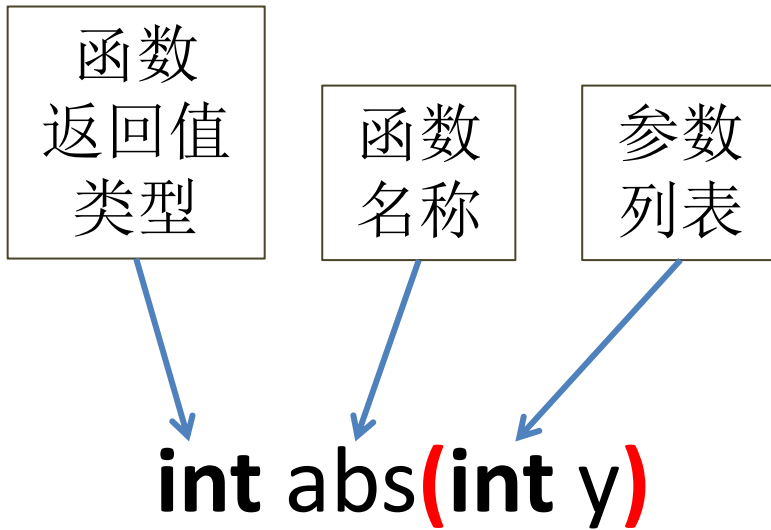
avg: average 平均值
定义一个函数avg计算3个整数的平均值

double avg(int x, int y, int z)

定义abs函数:
参数y为一个整数
返回一个整数答案
代表y的绝对值

abs:
absolute value 绝对值

绝对值函数



定义abs函数:
参数y为一个整数
返回一个整数答案
代表y的绝对值

abs:
absolute value绝对值

isPrime: Prime质数
(是质数吗)
定义一个函数isPrime判断
数字n是不是质数

bool isPrime(int n)

绝对值函数

```
3 int abs(int y){  
4     if(y>0) return y;  
5     else return -y;  
6 }
```

return操作将一个
答案值返回到调用
abs函数的位置

若y大于0, 将y返回

否则,将-y作为答案返回

函数可以有很多参数，但只有1个返回值！
函数运行到return就会结束运行

执行顺序

```
3  int abs(int y){
4      if(y>0) return y;
5      return -y;
6  }
7  int main(){
8      int n;
9      cin>>n;
10     cout<<abs(n);
11     return 0;
}
```

函数只有1个返回值！
函数运行到return就会
结束运行！

输入10

7, 8, 9, 10, 3, 4, 11

执行顺序

```
3  int abs(int y){
4      if(y>0) return y;
5      return -y;
6  }
7  int main(){
8      int n;
9      cin>>n;
10     cout<<abs(n);
11     return 0;
}
```

输入-5
7, 8, 9, 10, 3, 4, 5, 11

内存地址

```

1  #include<iostream>
2  using namespace std;
3  int main(){
4      int x,y,z;
5      cout<<&x<<endl;
6      cout<<&y<<endl;
7      cout<<&z<<endl;
8      return 0;
9  }

```

请同学运行程序
结果和老师一样吗?

0x提示后面
是十六进制

f是	e是	c是
15	14	12

0x22fe4c

0x22fe48

0x22fe44

输出内存地址

十六进制表示

取地址运算符 &

& 翻译为：取地址

&x 翻译为：x的地址

数组地址2

```
1  #include<iostream>
2  using namespace std;
3  int main(){
4      int f[3];
5      cout<<&f[0]<<endl;      0x22fe40
6      cout<<&f[1]<<endl;      0x22fe44
7      cout<<&f[2]<<endl;      0x22fe48
8      cout<<f<<endl;          0x22fe40
9      cout<<f+1<<endl;        0x22fe44
10     cout<<f+2<<endl;        0x22fe48
11     return 0;
12 }
```

太乙编程
www.etiger.vip

数组和地址

```
int f[3];
```

f &f[0] 数组名f记录数组0号元素的内存地址

f+1 &f[1] f+1记录数组1号元素的内存地址

f+2 &f[2] f+2记录数组2号元素的内存地址

指针

专门用于存储内存地址

引用运算符*

***** 翻译为：指向的内容

***p** 翻译为：p指向的变量值

int *p; 翻译为：定义p为指向整数的指针

p = &x; 翻译为：将x的地址赋值给p
也就是：让p指向了x

p代表pointer指针

请预测输出结果

观察老师的输出结果

```
int x=8;
```

```
int *p;
```

```
p = &x;
```

```
cout<< *p <<endl;
```

```
cout<< x <<endl;
```

```
cout<< &x <<endl;
```

```
cout<< p <<endl;
```

定义p是指针，指向整数

p赋值为x的地址

p指向x

输出p指向的变量的值

输出x的值

输出x的地址

输出p的值

就是x的地址

指针

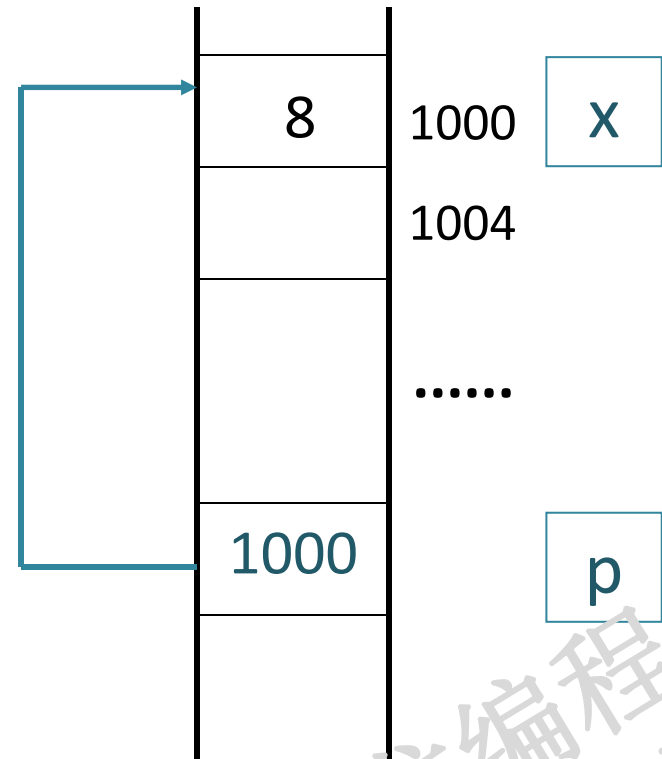
指针是**存储内存地址**的变量

```
int x=8;  
int *p;  
p = &x;
```

假设系统给x分配的地址是1000
那么p中保存的数据就是1000

两种访问方式:

- 直接访问变量x
- 间接访问指针p指向的变量内容



指针运算 &和*

```
int x=8;
```

```
int *p;
```

```
p = &x;
```

取地址运算符 & p赋值为x的地址

```
*p= 9;
```

引用运算符*

p指向的变量赋值为9

```
cout<<x<<endl;
```

等效于 x赋值为9

```

1  #include<iostream>
2  using namespace std;
3  int main() {
4      int x,y;
5      int *p;
6      x=3; y=4;
7      p=&x;
8      cout<<&x<<endl;
9      cout<<p<<endl;
10     cout<<*p<<endl;
11     *p=y+4;
12     cout<<x<<endl;
13     return 0;
14 }

```

p是指向整数的指针变量

p赋值为x的地址	p指向x
输出x的地址	
输出p	也就是输出x的地址
输出p指向的变量，也就是x	
p指向的变量x 赋值为y+4	

思考：为什么x变量的值最后变为8

指针演示

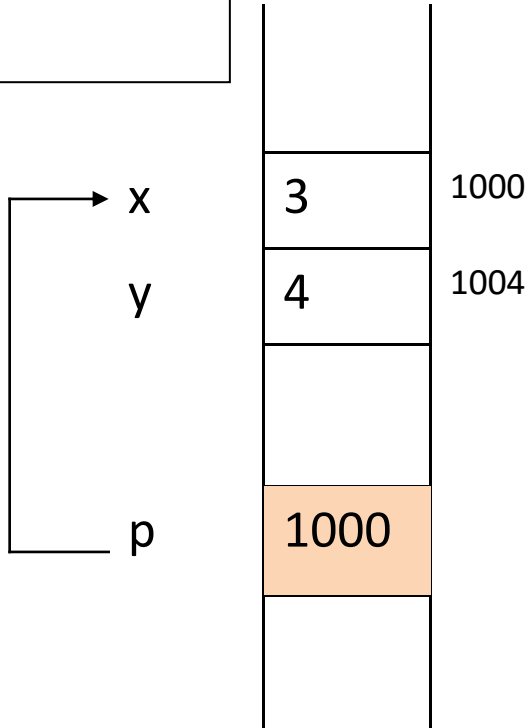
样例代码

```
int x,y,*p;
```

```
x=3;
```

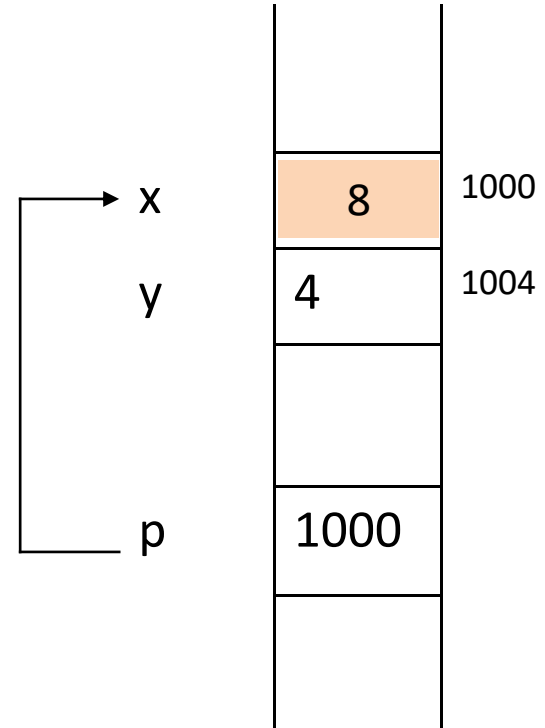
```
y=4;
```

```
p=&x;
```



执行代码

```
*p=y+4;
```



指针和数组

请预测
输出结果

```
1  #include<iostream>
2  #include<string>
3  using namespace std;
4  string x[3]={"ha","wa","la"};
5  int main(){
6      cout<<x<<endl;
7      cout<<x+1<<endl;
8      cout<<x+2<<endl;
9      cout<<*x<<endl;
10     cout<<*(x+1)<<endl;
11     cout<<*(x+2)<<endl;
12     cout<<(x+1)[0]<<endl;
13     cout<<(x+1)[1]<<endl;
14     return 0;
15 }
```

存储
内存
地址

数组名x
也是指针

x+1
也是指针

x+2
也是指针

请翻译第5-9行
并预测输出结果

数组名x
也是指针

将x数组1号元素的地址赋值给p

```
1  #include<iostream>
2  using namespace std;
3  int x[4]={5,6,7,8};
4  int main(){
5      int *p;
6      p=x+1;
7      cout<<p[0]<<endl;      6
8      cout<<p[1]<<endl;      7
9      cout<<p[2]<<endl;      8
10     return 0;
11 }
```

p[0] 相当于 *(p+0)

指针变量



数组名

都记录内存地址

顺序表

顺序表：数组实现

2	3	5	7
---	---	---	---	-------

存储要点

一段连续内存地址

顺序表算地址

定义数组 `int a[100];`
已知 `a[0]` 地址为 5000,
计算下列变量的地址

每个 `int` 变量

`int` 数组的每个元素

4 字节

`a[0]`

5000

`a[1]`

5004

`a[8]`

`a[50]`

`a[99]`

给定随机编号

顺序表可以

$O(1)$ 时间定位

$O(1)$ 存取元素

基本数据类型

类型	占字节Byte	占比特bit	取值范围
int	4	32	$-2^{31} \sim 2^{31}-1$
long long	8	64	$-2^{63} \sim 2^{63}-1$
double	8	64	15位有效数字
char	1	8	$-128 \sim 127$
bool	1	8	0或1

存储器的最小单位是字节
最小的存储单位是比特

1个字节=8个比特

基本数据类型

1个字节=8个比特

int变量的范围是 $-2^{31} \sim 2^{31}-1$ ，能表示 2^{32} 个数

long long变量的范围是什么？能表示几个数

878

读题确认题目大意

6	string type;	// 字符串type表示类型名称
7	int n;	// 整数n表示数组元素个数
8	int addr;	// 整数addr表示数组起始地址
9	int q;	// 整数q表示询问的地址
10	int nByte;	// 整数nByte表示每个元素占几个字节
11	cin>>type>>n>>addr>>q;	
12		// 如果类型名称为char
13		// 每个元素占1个字节
14		// 否则
15		// 每个元素占4个字节
16		// 整数id表示查询元素的编号
17		// 如果编号小于零或编号大于等于n
18		// 输出错误, 换行
19		// 否则
20		// 输出编号, 换行

数组最值

max_element()

```
1  #include<iostream>
2  #include<algorithm>
3  using namespace std;
4  int f[5]={3,1,5,1,2};
5  int main(){
6      cout<<max_element(f, f+5)<<endl;
7      cout<<max_element(f, f+5)-f<<endl;
8      cout<<*max_element(f, f+5)<<endl;
9      return 0;
10 }
```

算法库

地址

编号

数值

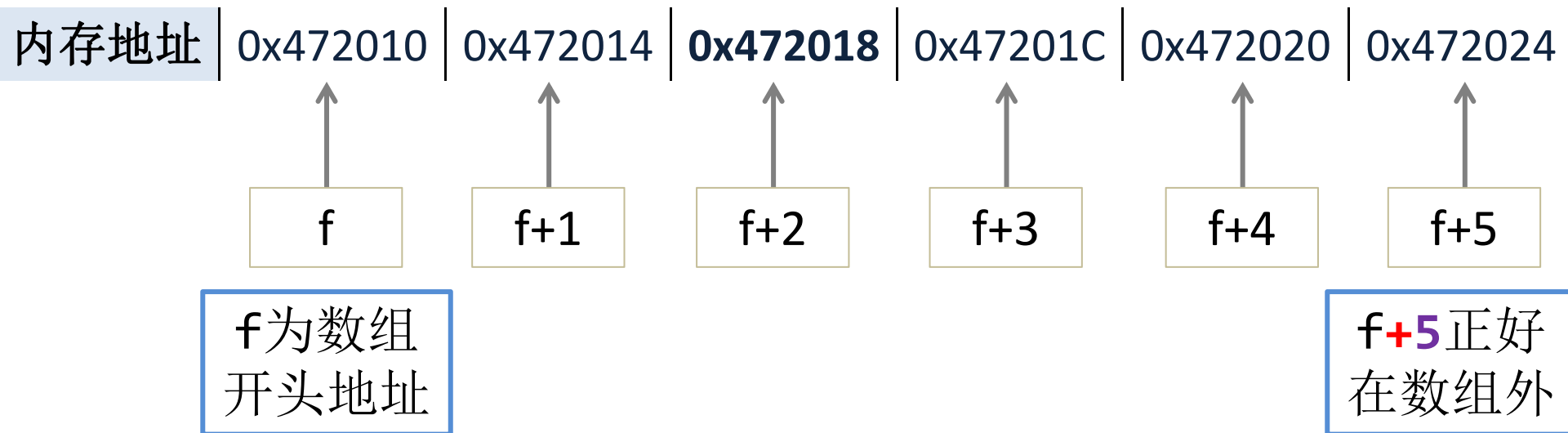
max
最大

element
元素

数组的内存地址

```
int f[5]={3,1,5,1,2};
```

数组编号	0	1	2	3	4	
数组元素值	3	1	5	1	2	



数组最值的地址

```
int f[5]={3,1,5,1,2};
```

数组编号	0	1	2	3	4	
数组元素值	3	1	5	1	2	







内存地址	0x472010	0x472014	0x472018	0x47201C	0x472020	0x472024
	↑	↑	↑	↑	↑	↑
	f	f+1	f+2	f+3	f+4	f+5

`max_element(f, f+5)`
寻找从地址f开始, 在地址f+5之前
最大值的第一个地址
返回值为内存地址**0x472018**

数组最值的编号

```
int f[5]={3,1,5,1,2};
```

数组编号	0	1	2	3	4	
数组元素值	3	1	5	1	2	







内存地址	0x472010	0x472014	0x472018	0x47201C	0x472020	0x472024
						
	f	f+1	f+2	f+3	f+4	f+5

$\text{max_element}(f, f+5) - f$
是两个地址 $f+2$ 和 f 的距离
结果为整数 2, 代表数组编号

数组最值的数值

```
int f[5]={3,1,5,1,2};
```

数组编号	0	1	2	3	4	
数组元素值	3	1	5	1	2	

内存地址	0x472010	0x472014	0x472018	0x47201C	0x472020	0x472024
						
	f	f+1	f+2	f+3	f+4	f+5

`*max_element(f, f+5)`
对应最大数值
星号*能取出地址内存放的数值

数组求最值

`min_element(f, f+n)`

翻译为：寻找从地址f开始, 在地址f+n之前
n个数里最小值的第一个地址

`min_element(f, f+n)`
`min_element(f, f+n)-f`
`*min_element(f, f+n)`

地址
编号
数值

时间复杂度 $O(n)$
实现方法还是逐个打擂台比大小

1147

算法步骤

输入年数 n 和团员数量 k

共循环 n 次

输入第 i 人的颜值 $x[i]$

共循环 $n-k+1$ 次：枚举起点编号 i

输出 $x[i]$ 开始的 k 个数字里的最小值

时间复杂度 $O(n*k)$

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  const int N=200009;    //定义整数常量N赋值为200009
4  int x[N];              //定义数组x包含N个整数,x[i]表示第i人的颜值
5  int main(){
6      freopen("xjj.in","r",stdin);
7      freopen("xjj.out","w",stdout);
8          //定义整数变量n,表示共有几年
9          //定义整数变量k,表示团员数量
10         //输入年数n和团员数量k
11         //for循环:计数器i从1自增到n,共循环n次
12         //输入第i人的颜值x[i]
13         //for循环:计数器i从1自增到n-k+1,共循环n-k+1次
14         //定义整数变量ans,表示x[i]开始的k个数字里的最小值
15         //输出ans,空格
16     }
17     return 0;          //主函数结束,返回零
18 }

```

太戈编程

1797

878

1147

只要60分

拓展题

1447,676