函数的知识和内容以及递归函数

递归函数
1>>>>自己要使用自己
2>>>>一定要有一个结束基线

也就是：

一般情况下要有一个参数，且必须拥有返回值
要有一个基线情况：基线情况下返回值
要有一般情况：一般情况下递归>>>>自己调用自己的情况

如一个循环加法：

```
#include<stdio.h>

int a(int n)
{
if(n == 1)              //基线，在到达此处时结果开始固定：：基线情况
return 1;               //必须要有返回值
else                    //通常情况下：：一般情况
return (n*a(n-1));      //递归的过程
}

int main()
{
int ac = 4;
ac = a(ac);
printf("%d\n",ac);
return 0;
}
//数组的结构和形式

//数组的结构：数组名和数组元素+数组的下标
//定义数组的时候不能使用变量定义数组的大小
```

//数组的二维的长度不可以忽略

//向函数传递一维数组
/*数组名作为实参进行传递即可
函数原型的申明
int abs(int abs);

在传递数据的时候不需要输入数组的后面的方框
只要使用不带方框的数组名作为函数的实参即可
数组的名称代表数组的第一个元素的地址
所以其在传递的过程中实际上是将数组的首地址传递给被调函数

数组在做函数形参时，数组的长度可以不出现在数组名后面的方括号内。
通常用另一个整形实参来指定数组的长度。
例如：int average（int sore[], int n）

//使用排序算法对一个队列进行排序处理
//使用排序算法对一个队列进行排序处理

//在被调用的函数中改变形参元素值时，实参数组的元素值也会随之发生改变
：：：这是因为其传递的值是来自于同样的地址，共享同一段内存单元所造成的
*/
//排序的算法和数据的传递
//当形参被声明为二维数组时，可以省略第一位的长度的声明，而不可以省略第二维的长度声明
排序和擦找算法和数据的传递

```
int argv[maxsize] = {9,8,7,4,5,6};
int average(int var[])
{
        int i;
        int j;
        int temp;
```

```
        for(i = 0;i<maxsize-1;i++)
        {
                for(j = i+1;j<maxsize;j++)
                {
                if(var[j]>var[i])
                        {
                                temp = var[j];
                                var[j] = var[i];
                                var[i] = temp;
                        }
                }
        }

}

int  find(int a)
{
int mid = 0;
int min = 0;
int max = maxsize-1;
while(min<=max)
{
mid = (min + max)/2 ;
if(a>mid)
        min = mid+1;
else if(mid<mid)
        max = mid-1;
else
        return 23 ;
}
}
```

```
int main( )
{
        int j;
        int temp;
        average(argv);
        for(j = 0;j<maxsize;j++)
                {
                printf("%d ", argv[j]);
                }
                printf("\n");


        temp = find(4);
        printf("%d ", temp);
        return 0;
}
```