学号	2021144301	姓名	王大殿	实验日期	
实验名称	指针与数组（1）				

实验目的：

1．掌握指针也变量地址的关系，掌握指针*与&运算基本方法；

2．掌握指针和一维数组间的关系，掌握用指针变量引用一维数组元素的方法。

实验任务、步骤与结果：

任务 1：求圆的周长与面积

题目描述

任务编制一个计算圆的面积与周长的程序，要求从键盘输入圆的半径，打印输出圆的周长与面积。

代码：

```c
#include<stdio.h>
#include<stdlib.h>
int getaver(int *p);
int main(){
        int a[10];
        printf("Input 10 number:\n");
        for(int i=0; i<10; i++){
                scanf("%d",&a[i]);
        }
        int aver=getaver(a);
        printf("Average is %d\n",aver);
        return 0;
}
int getaver(int *p){
        int sum=0;
        for(int i=0; i<10; i++){
                sum+=*p;
                p++;
        }
        return sum/10;
}
```

运行结果

```
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验8$
Input 10 number:
1 2 3 4 5 6 7 8 9 10
Average is 5
```

任务 2：数字交换位置

题目描述

函数 void max_min_value (int *number,int n)实现将一个数组中最小数与第一个数对换，最大数与最后一个数对换。

输入

2 行：

第 1 行，n(n<20)

第 2 行，输出数组中 n 个元素

输入

1 行：

交换后的数组，每个元素之间用空格分开

```c
#include<stdio.h>
void max_min_value(int *number, int n);
int main(){
        int n;
        printf("Enter a number:\n");
        scanf("%d",&n);
        int a[n];
        printf("Enter %d number of array:\n",n);
        for(int i=0; i<n; i++){
                scanf("%d",&a[i]);
        }
        max_min_value(a,n);
        printf("Result is: \n");
        for(int i=0; i<n; i++){
                printf("%d ",a[i]);
        }
        printf("\n");
        return 0;
}
void max_min_value(int *number, int n){
        int max, min, max_n=0, min_n=0;
        int *temp, swap;
        max=min=*number;
        temp=number;
        for(int i=0; i<n; i++){
                if(*temp>max){
                        max=*temp;
                        max_n=i;
                }else if(*temp<min){
                        min=*temp;
                        min_n=i;
                }
                temp++;
        }
        //交换数值
        *(number+min_n)=*number;
        *number=min;
        *(number+max_n)=*(number+n-1);
        *(number+n-1)=max;
}
```

运行结果：

```
daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/
Enter a number:
4
Enter 4 number of array:
3 2 5 4
Result is:
2 3 4 5
```

任务 3：数字向后移动

题目描述

有 n 个整数，编写函数 int move_n(int* a,int n,int m)，使前面各数顺序向后移 m 个位置，即最后 m 个数变成最前面 m 个数。

输入

3 行：

第 1 行，n

第 2 行，m

第 3 行，n 个整数（中间以空格隔开）

输入

1 行：

变序后的 n 个整数

```c
#include<stdio.h>
int move_n(int *a, int n, int m);
int main(){
        int n, m;
        printf("Enter a number n:\n");
        scanf("%d",&n);
        printf("Enter a number of m:\n");
        scanf("%d",&m);
        int a[n];
        for(int i=0; i<n; i++){
                a[i]=i+1;
        }
        move_n(a,n,m);
        for(int i=0; i<n; i++){
                printf("%d ",a[i]);
        }
        printf("\n");
        return 0;
}
int move_n(int *a, int n, int m){
        int index=0;
        for(int i=0; i<n; i++){
                index=(i+m)%(n);
                *(a+index)=i+1;
        }
        return 0;
}
```

运行结果：

```
daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud
Enter a number n:
6
Enter a number of m:
2
5 6 1 2 3 4
daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud
Enter a number n:
8
Enter a number of m:
5
4 5 6 7 8 1 2 3
```

任务 4：数字反序排列

题目描述

将数组 a 中 n 个整数按相反的顺序存放。编写函数 void　invert(int *p,int n)实现以上功能。

输入

2 行：

第 1 行，输入数组 a 的长度 n；

第 2 行，输入数组 a。

输出

1 行：

反序后的数组 a。

```c
#include<stdio.h>
void invert(int *p, int n);
int main(){
        int n;
        printf("Enter the length of array:\n");
        scanf("%d",&n);
        int a[n];
        printf("Input the number of array:\n");
        for(int i=0; i<n; i++){
                scanf("%d",&a[i]);
        }
        invert(a,n);
        for(int i=0; i<n; i++){
                printf("%d ",a[i]);
        }
        printf("\n");
        return 0;
}
void invert(int *p, int n){
        int mid=n/2;
        int swap;
        for(int i=0; i<mid; i++){
                swap=*(p+i);
                *(p+i)=*(p+n-i-1);
                *(p+n-i-1)=swap;
        }
}
```

运行结果：

```
daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud
Enter the length of array:
4
Input the number of array:
6 8 3 7
7 3 8 6
daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud
Enter the length of array:
5
Input the number of array:
7 6 3 4 8
8 4 3 6 7
```

任务 5：约瑟夫问题

题目描述

n 个人围成一圈，顺序排号。从第 1 个人开始报数（从 1 到 m）,凡报到 m 的人退出圈子，问最后留下的是原来第几号。用函数和指针实现以上功能，返回数组中最后留下的元素序号。

输入

1 行：

n 和 m，中间使用空格分开，分别表示总人数和该出局的报数。

输入

输出最后留下者的序号 x,输出格式为：NO.x

```c
#include<stdio.h>
int Josef(int *a, int n, int m);
int main(){
        int n, m;
        printf("Enter n and m:\n");
        scanf("%d%d",&n,&m);
        int a[n];
        for(int i=0; i<n; i++){
                a[i]=i+1;
        }
        int No;
        No=Josef(a,n,m);
        printf("NO.%d\n",No);
        return 0;
}
int Josef(int *a, int n, int m){
        int count=0, k=0, j=0, i=0;
        //count用于统计0的个数，k用于表示当前序号，j用于累加每一个非0项，i正常累计用于控制循环进行
        while(count<n-1){
                k=i%n;
                if(*(a+k)!=0){
                //遇到非0则j+1
                        j++;
                        if(j%m==0){
                        //j%m代表非零项每过m次判断一次
                                *(a+k)=0;
                                count++;
                        }
                }
                i++;
        }
        for(int h=0; h<n; h++){
                if(*(a+h)!=0){
                        return h+1;
                }
        }
        return 0;
}
```

运行结果：

题目描述
有一数组 a[N]，需要对其局元素进行从高到低排序，请编写函数 localsort(int *start ,int len)，实现数组指定位置开始后的 len 个元素从高到低排序。N<100，start+len<&a[N-1]。
输入
2 行：
第 1 行,n,b,l，n 表示数组元素个数，n<=N,b 表示开始排序的下标，1<b<=n,l 表示从开始下标起排序元素长度,1<=b+l<=n；
第 2 行,n 个数组元素。
输出
1 行，局部排序后的数组

【示例 1】
输入
10 2 5
1 2 3 4 5 6 7 8 9 10
输出
1 6 5 4 3 2 7 8 9 10

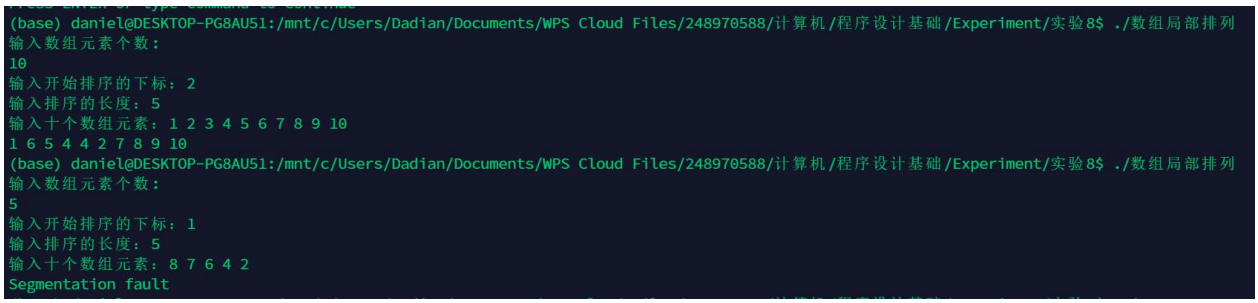
输入
5 1 5
6 8 7 4 2
输出
8 7 6 4 2

代码：

```
#include<stdio.h>
typedef int status;
#define OK 1
#define False 0
status localsort(int *start, int len);
int main(){
```

```
29              int a[100];
30              int n, b, l;
31              printf("输入数组元素个数:\n");
32              scanf("%d",&n);
33              printf("输入开始排序的下标：");
34              scanf("%d",&b);
35              printf("输入排序的长度：");
36              scanf("%d",&l);
37              printf("输入十个数组元素：");
38              for(int i=0; i<n; i++){
39                      scanf("%d",&a[i]);
40              }
41              localsort(&a[b-1],l);
42              for(int i=0; i<n; i++){
43                      printf("%d ",a[i]);
44              }
45              printf("\n");
46              return 0;
47 }
48 status localsort(int *start, int len){
49              int *tmp;
50              tmp=start;
51              if(!tmp){
52                      return False;
53              }
54              for(int i=0; i<len-1; i++){
55                      int index;
56                      int num=*(tmp+i);
57                      for(int j=i+1; j<len; j++){
58                              if(*(tmp+j)>num){
59                                      num=*(tmp+j);
60                                      index=j;
61                              }
62                      }
63                      *(tmp+index)=*(tmp+i);
64                      *(tmp+i)=num;
65              }
66              return OK;
67 }
```

运行结果：



结果错误

发现问题与分析：在数值交换程序中，一开始直接用*number 完成循环动作，然后想把 max 值给*number 时却无法正确输出。出现了 2000 多这样离谱的数值，初步怀疑是地址溢出。后来在程序中设了一个*temp，把 number 地址赋给 temp，赋值比较过程由 temp 进行。最后的结果再赋给*number。

备注：

说明：

1、 各项记录应完备详尽。

2、 各栏目如果长度不够可以自行调整大小，但应注意排版的美观。

3、 可将实现过程的代码、运行效果等内容以帖图方式放在相应栏目。