

实验 12 指针及应用

一、实验目的

- (1) 通过实验进一步掌握指针的概念，会定义和使用指针变量；
- (2) 能正确使用数组的指针和指向数组的指针变量；
- (3) 能正确使用字符串的指针和指向字符串的指针变量；

二、实验内容和步骤

I.基础部分

- (1) 给定程序 c12-1-1.c，请分析程序的运行结果，回答问题。

```
/* c12-1-1.c */
#include <stdio.h>
int main()
{
    int a,*p;
    p=&a;
    *p=10;
    printf("%d    %d\n",p,*p);
    return 0;
}
```

- ① 程序 c12-1.c 中，很多个语句中都出现了*p。

语句：“int a,*p;”和语句：“*p=10;”中，*p 分别代表什么含义？

- ② 语句：

```
printf("%d    %d\n",p,*p);
```

中，第一项和第二项分别输出什么值？

- (2) 给定程序 c12-1-2.c，请分析程序的运行结果，回答问题。

```
/* c12-1-2.c */
#include <stdio.h>
#include<string.h>
int main()
{
    char *s1;
    char s2[20];
    s1="abcde";
    puts(s1);
    strcpy(s2,"ABCDE");
    puts(s2);
    return 0;
}
```

}

- ① 为什么对 s1 采用直接赋值,而 s2 采用 strcpy 函数赋值? 可以通过语句:s2="ABCDE"给 s2 赋值吗?
- ② 对 s1 和 s2,可以分别采用 gets(s1)、gets(s2)从键盘输入字符串吗? 为什么?
- ③ 给定程序 c12-1-3.c,程序的运行结果如图 12-1 所示,请分析程序的运行结果,回答问题。

```
*****弘扬社会主义核心价值观*****
富强 民主 文明 和谐
自由 平等 公正 法制
爱国 敬业 诚信 友善
```

图 12-1 程序 c12-1-3.c 的运行结果

```
/* c12-1-3.c */
#include <stdio.h>
int main()
{
    char concept[12][20]={"富强","民主","文明","和谐","自由","平等","公正","法制","爱国","敬业","诚信","友善"};
    int i;
    printf("*****弘扬社会主义核心价值观*****\n");
    for(i=0;i<12;i++)
    {
        printf("    %s",concept[i]);
        if((i+1)%4==0) printf("\n");
    }
    return 0;
}
```

- ① 可以将数组的定义 char concept[12][10]改为 char *concept[12],程序的其他部分不变,运行结果也不变。char concept[12][20]和 char *concept[12]这 2 种定义有何区别?
- ② 还可以有其他的定义方式吗?

II.提高部分

1、程序填空

说明:程序有多个空(1)、(2)……需要补充完整。请将程序中的__ (1) __、__ (2) __……删除后,在相应的位置填入正确答案并调试直到得到正确结果为止。

注意:不要随意改动程序,不得增行或删行,也不得更改程序的结构!

(1) 给定程序的功能是:从键盘输入一个字符串,将其中的大写字母转换成小写字母,然后输出。程序运行结果如图 12-2 所示,请把程序补充完整。

```
please input string:
ABcdEFgh12sd
abcdefghijkl2sd
```

图 12-2 程序的运行结果

① 程序 c12-2-1.c 通过字符数组下标变量实现运行结果。

```
/* c12-2-1.c */
#include <stdio.h>
int main()
{
    char s[20];
    int i;
    printf("please input string:\n");
    gets(s);
    for(i=0;__(1)__;i++)
        if(__(2)__) s[i]=s[i]+'a'-'A';
    puts(s);
    return 0;
}
```

② 程序 c12-2-2.c 通过字符指针变量实现运行结果。

```
/* c12-2-2.c */
#include <stdio.h>
int main()
{
    char s[20];
    char *p;
    printf("please input string:\n");
    scanf("%s",s); /*注意用 scanf()输入和 gets()输入的区别*/
    p=s;
    while(*p!='\0')
    {
        if(*p>='A'&&*p<='Z') __(1)__;
        p++;
    }
    __(2)__;
    while(*p!='\0')
    {
        putchar(__(3)__);
        __(4)__;
    }
    printf("\n");
    return 0;
}
```

(2) 给定程序的功能是：分别采用下标法、数组名法和指针法访问数组元素，求出 10 个数中的最大值。程序运行结果如图 12-3 所示，请把程序补充完整。

```
please input array a:
10 20 90 50 70 80 40 60 70 30
MAX=90
```

图 12-3 程序的运行结果

- ① 程序 c12-2-3.c 通过数组下标变量实现运行结果。

```
/* c12-2-3.c */
#include <stdio.h>
int amax(int a[],int n);
int main()
{
    int a[10],i;
    printf("please input array a:\n");
    for (i=0;i<10;i++)
        scanf ("%d", &a[i]);
    printf ("MAX=%d\n",__(1)__);
    return 0;
}
__(2)__ amax(int a[],int n)
{
    int max,i;
    max=a[0];
    for (i=1;i<n;i++)
        if (max<a[i]) __(3)__;
    return max;
}
```

- ② 程序 c12-2-4.c 通过数组名实现运行结果。

```
/* c12-2-4.c */
#include <stdio.h>
int amax(int *a,int n);
int main()
{
    int a[10],i;
    printf("please input array a:\n");
    for (i=0;i<10;i++)
        scanf ("%d",a+i);
    printf ("MAX=%d\n",amax(a,10));
    return 0;
}
```

/*下列函数中不能出现诸如 a[i]之类的下标变量*/

```
int amax(int *a,int n)
{
```

```

int max,i;
max=__ (1)__;
for (i=1;i<n;i++)
    if (__ (2)__) max=*(a+i);
return max;
}

```

③ 程序 c12-2-5.c 通过指针变量实现运行结果。

```

/* c12-2-5.c */
#include <stdio.h>
int amax(int *a,int n);
int main()
{
    int a[10];
    int *p;
    printf("please input array a:\n");
    for (p=a;p<a+10;p++)
        scanf ("%d",__ (1)__);
    printf ("MAX=%d\n",amax(a,10));
    return 0;
}

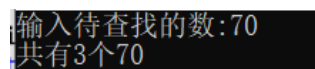
int amax(int *a,int n)
{
    int max;
    int *p;
    max=*a;
    for (p=__ (2)__;p<a+n;p++)
        __ (3)__;
    return max;
}

```

2、程序改错

注意：程序中的其它地方不要随意改动，不得增行或删行，也不得更改程序的结构！

(1) 给定程序 c12-2-6.c 的功能是：下列程序实现顺序查找，并能统计待查找的数的个数。下列程序有 2 处错误，请改正程序中的错误。程序正确运行输出结果如图 12-4 所示。



```

输入待查找的数:70
共有3个70

```

图 12-4 程序 c12-2-6.c 的运行结果示例

```

/* c12-2-6.c */
#include<stdio.h>
#define N 10

```

```

int main()
{
    int a[N]={ 70,50,70,90,80,65,70,50,90,65};
    int *p,x,count=0;
    printf("输入待查找的数:");
    scanf("%d",&x);
    for(p=a;p<=a+N;p++)
        if(x==p)    count++;
    printf("共有%d 个%d\n",count,x);
    return 0;
}

```

(2) 给定程序 c12-2-7.c 中的 cmpstr 函数实现 2 个字符串相比较。当 $s1=s2$ 时，函数返回值为 0，如果 $s1 \neq s2$ ，则返回它们二者第一个不相同字符的 ASCII 码差值(如“BOY”与“BAD”的第二个字母不相同，'O' 与 'A' 的 ASCII 之差为 $79-65=14$)。下列 cmpstr 函数中有 2 个错误，请改正程序中的错误。程序正确运行输出结果如图 12-5 所示。

```

please input string s1:BOY
string s1 is:BOY
please input string s2:BAD
string s2 is:BAD
s1>s2

```

```

please input string s1:ABCD
string s1 is:ABCD
please input string s2:ABCD
string s2 is:ABCD
s1==s2

```

图 12-5 程序 c12-2-7.c 的运行结果示例

```

/* c12-2-7.c */
#include<stdio.h>
int cmpstr(char *s1,char*s2)
{
    char *p1,*p2;
    p1=s1;
    p2=s2;
    while(*p1!=0 || *p2!=0)
    {
        if(*p1==*p2)
        {
            p1++;
            p2++;
        }
        else
            break;
    }
    return p1-p2;
}

int main()

```

```

{
    char s1[80],s2[80];
    int c;
    printf("please input string s1:");
    gets(s1);
    printf("string s1 is:%s\n",s1);
    printf("please input string s2:");
    gets(s2);
    printf("string s2 is:%s\n",s2);
    c=strcmp(s1,s2);
    if(c>0) printf("s1>s2\n");
    else
        if(c<0)
            printf("s1<s2\n");
        else
            printf("s1==s2\n");
    return 0;
}

```

3、程序设计

(1) 编写程序 **c12-2-8.c**，程序实现的功能是：统计一个字符串中大写字母、小写字母、数字和其他字符的个数，用指针实现。

(2) 编写程序 **c12-2-9.c**，程序实现的功能是：假设有 20 个英文姓名，将姓名按升序排序。

III.拓展部分:通过指针实现单链表

(1) 编写程序 **c12-3.c**，实现单链表的相关操作。

假设有如下定义的单链表结点：

```

struct List
{
    int num;
    char name[20];
    struct List *next;
};

```

编程实现如下功能：

① 创建单链表，函数原型为：

```
struct List * H_creat_list();
```

函数返回单链表的头结点地址。

② 输出单链表，函数原型为：

```
void print(struct List *head);
```

其中，head 为单链表的头指针。

③ 在一个单链表中指定的位置 k 插入一个结点，函数原型为：

```
struct List *insert_Node_k(struct List *head, int num ,char name[],int k);
```

其中，head 为单链表的头指针，num 和 name[]为待插入的学号和姓名，函数返回插入后的

单链表的头结点的指针。

④ 删除单链表中学号为 x 的结点

`struct List *Del_Node_x(struct List *head,int x);`

其中，`head` 为单链表的头指针，函数返回删除后的单链表的头结点的指针。