深 圳 大 学 实 验 报 告

课程名称：面向对象程序设计

实验项目名称：实验二 指针的深入浅出

学院：医学院

专业：生物医学工程

指导教师：李乔亮、邓云

报告人：陈焕鑫 学号：**2016222042** 班级：生工 2 班

实验时间：**2018.9.26**

实验报告提交时间：**2018.10.14**

教务部制

实验题目：

熟练掌握 C 语言中的指针的使用，了解结构体中变量的内存分布

实验内容：

实验课堂内容：运行 vctest 工程，调试跟踪每一步的内存变化，将理论推导结果与 vc 运行结果进行对照，验证。

实验报告要求：对本 vc 工程的每一步，逐条解释其输出，从指针与内存的角度说明为什么，画出内存分布图

实验环境与程序代码：

实验环境：win7 系统下的 Visual C++ 6.0

```
// testp.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"

#include "stdlib.h"
int main(int argc, char* argv[])
{
  printf("/*------------------ please run all test and say why  -------------------- */\n");
  {



  printf("/*------------------ test  1-------------------- */\n");

  int i = 0;



  int * p = &i;

  *p = 0x02;

  printf(" i = 0x%x\n" , i );

  *p = 0x0102;

  printf(" i = 0x%x\n" , i );
  printf(" p = 0x%x, *p = 0x%x, i = 0x%x\n" ,p, *p , i);
```

```
printf("/*------------------ exam  1-------------------- */\n");

int * p1;
p1 = p;
* p1 = 0x3;

printf(" p1 = 0x%x, *p1 = 0x%x, i = 0x%x\n" ,p1, *p1,  i );
printf(" p = 0x%x, *p = 0x%x, i = 0x%x\n" ,p, *p , i);
}

{

printf("/*------------------ test  2-------------------- */\n");

int i = 0x01020304;

char * p1 = (char *)&i;

*p1++ = 0x3;

printf(" i = 0x%x\n" , i );

printf("/*------------------ exam  2-------------------- */\n");


unsigned char *pc;
pc = (unsigned char *) &i;

*pc++ = 0x90;
*pc++ = 0x03;

printf(" i = 0x%x\n" , i );
printf(" pc[0] = 0x%x, pc[1] = 0x%x\n" , pc[0],pc[1] );

}


{

printf("/*------------------ test  3-------------------- */\n");


struct node{
```

```
  int a;

  int b;
};

struct node n;

int i = 0x02;

int * p1 = (int *)&i;

n.a = 0x01;
n.b = * p1;
printf(" n.a = 0x%x n.b = 0x%x\n" , n.a, n.b );

int * pa = & n.a;
int * pb = & n.b;

*pa = 0x5;
*pb = *pa + 1;
printf(" n.a = 0x%x n.b = 0x%x\n" , n.a, n.b );

printf("/*------------------ exam  3-------------------- */\n");

*pa++ = 0x7;
*pa++ = 0x8;
printf(" n.a = 0x%x n.b = 0x%x\n" , n.a, n.b );
*pb++ = 0x9;
*pb++ = 0xa;
printf(" n.a = 0x%x n.b = 0x%x\n" , n.a, n.b );

}




  {
printf("/*------------------ ,( -------------------- */\n");

struct node{
  char a;
  int b;
  short c;
};
struct node n;
```

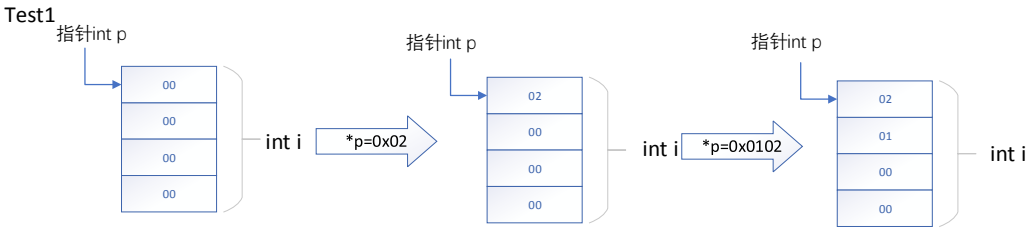
```
    n.a = 0x12345;
    n.b = 0x23456;
    n.c = 0x34567;



    * ++p = 1;
    * ++p = 10;
    * p++ = 20;
    printf(" n.a = 0x%x n.b = 0x%x n.c = 0x%x\n" , n.a, n.b, n.c );

    }



    /*------------------ END test  -------------------- */
    printf("\n\n\t\tYou are the best now!\n\n\n");

    return 0;
}
```

运行程序之后出现的结果如下图所示：

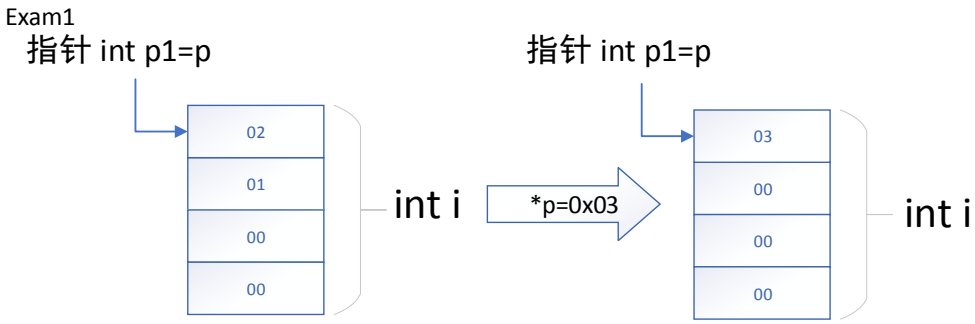
```
/*------------------ please run all test and say why  -------------------- */
/*------------------ test  1-------------------- */
 i = 0x2
 i = 0x102
 p = 0x18ff44, *p = 0x102, i = 0x102
/*------------------ exam  1-------------------- */
 p1 = 0x18ff44, *p1 = 0x3, i = 0x3
 p = 0x18ff44, *p = 0x3, i = 0x3
/*------------------ test  2-------------------- */
 i = 0x1020303
/*------------------ exam  2-------------------- */
 i = 0x1020390
 pc[0] = 0x2, pc[1] = 0x1
/*------------------ test  3-------------------- */
 n.a = 0x1 n.b = 0x2
 n.a = 0x5 n.b = 0x6
/*------------------ exam  3-------------------- */
 n.a = 0x7 n.b = 0x8
 n.a = 0x7 n.b = 0x9
/*------------------ 思考题-------------------- */
 n.a = 0x45 n.b = 0x1400 n.c = 0x4567


                You are the best now!
```

实验结果与分析：

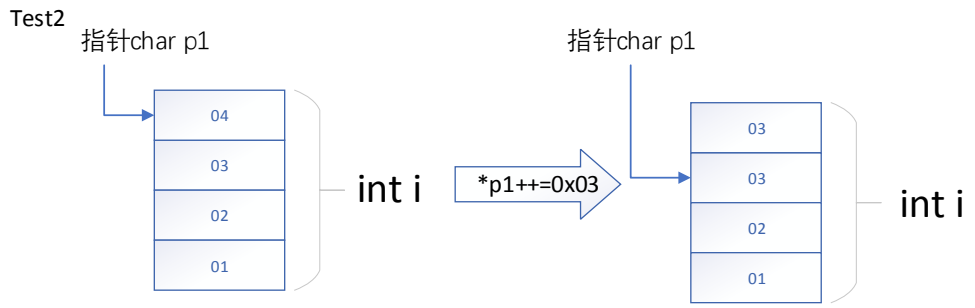
1、在 test 1 中，将 int 类型的指针指向了 i 的地址，然后通过指针运算符修改 i 的地址的值为 0x02，因此，打印输出的 i 的值为 0x2；之后使用指针运算符对 p 赋值 0x0102，然后打印输出 i= 0x0102，p 是 i 的地址，*p 的值与 i 的值相等。



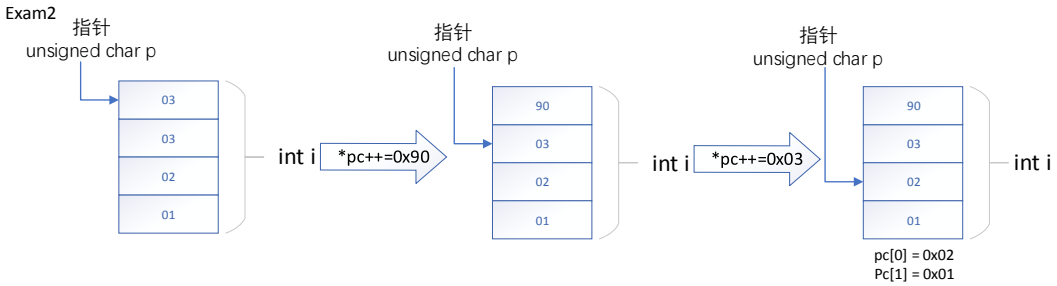
2、在 exam 1 中，首先声明了一个 int 类型的指针 p1，将 p 的地址赋给 p1，这样，因为 p 指向 i 的地址，所以 p1 也同样指向 i 的地址。使用指针运算符对 p1 进行操作，修改 i 的值为 0x3，打印输出的 p1 与 p 的值都是 i 的地址，*p1，*p，i 的值是相等的，都为 0x3



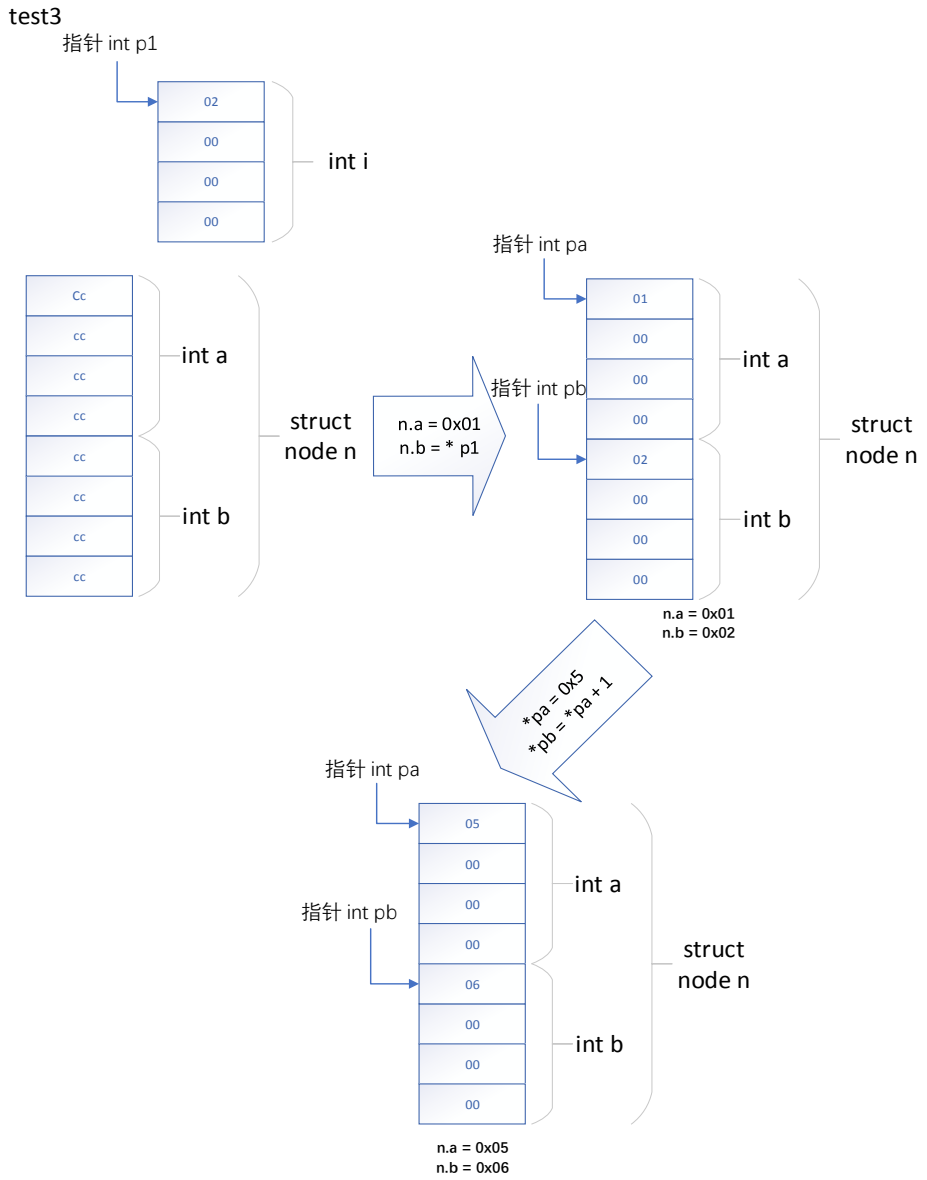
3、在 test 2 中，int 类型的 i 的值为 0x01020304，使用 char 类型的指针 p1 指向 i 的地址时，由于 int 占 4 个字节，而 char 只占一个字节，所以发生了截断，*p1=0x04。"*p1++ = 0x3",这个语句先执行"*p1=0x3"，在对指针 p1 加 1。所以 i 的最低一个字节被修改为 0x03 之后，p1 指向了 i 的第 2 个字节。因此，i 的值变成了 0x01020303。



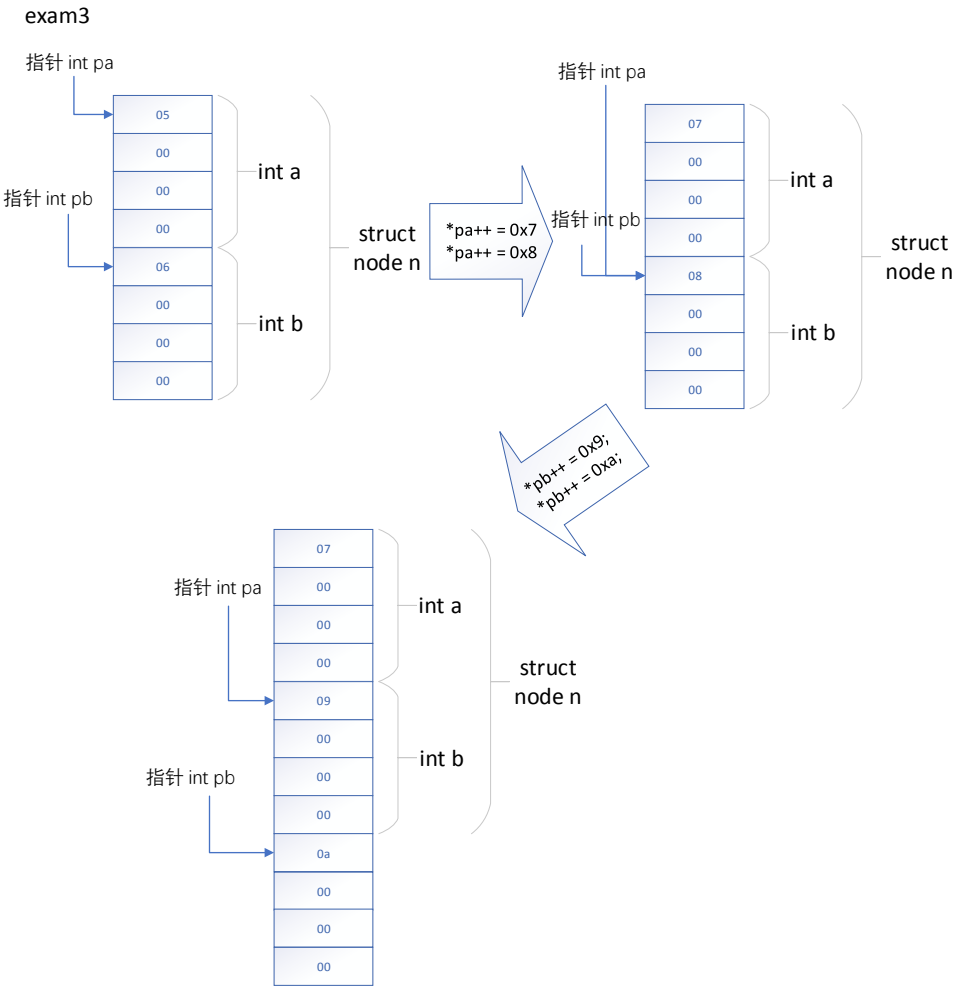
4、在 exam 2 中，将 i 的地址转换成了 unsigned char 类型的指针，并将其赋值给了 unsigned char 类型的指针变量 pc，因此*pc 的值为 0x04。"*pc++ = 0x90;"这个语句先将 pc 原本指向的地址的内容（也就是 i 的第一个字节）赋值为 0x90，此时 i 变成了 0x01020390，然后 pc 指向 i 的第二个字节。"*pc++ = 0x3;"这个语句又将 i 的第二位赋值为 0x03，然后指针加 1。至此，打印输出的值 i 为 0x01020390，pc[0]的值为 0x02，pc[1]的值为 0x01。



5、在 test 3 中，结构体 node 类型的变量 n 中的两个 int 类型的成员变量连续占 8 个字节的内存空间（每个 int 类型的变量占 4 个字节）。p1 指针指向了 i，i 的值为 0x02，那么*p1 的值为 0x02。之后将 n 的成员变量 a 的值修改为 0x01，b 的值修改为*p1 的值，也就是 0x02。屏幕上显示的是"n.a = 0x01 n.b = 0x02"。使用 int 类型的指针 pa 和 pb 分别指向了成员变量 a 和 b 的地址，使用指针运算符操作 pa 修改成员变量 a 的值为 0x5，成员变量 b 的值为*pa 的值加 1，等于 0x6。所以屏幕上显示的是"n.a = 0x5 n.b = 0x6"。

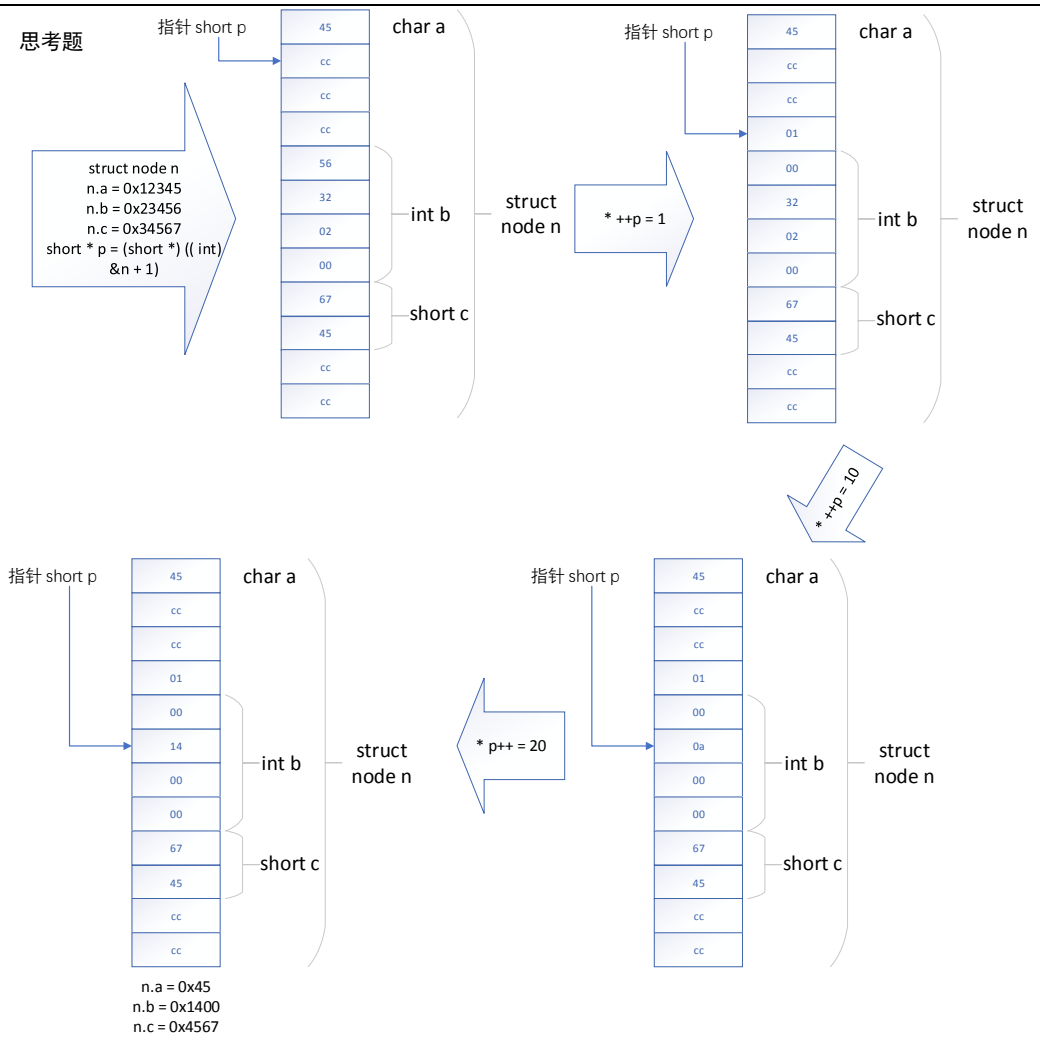


6、在 exam 3 中，"*pa++ = 0x07"先将成员变量 a 的值修改为 0x7，再将指针指向了成员变量 b。"*pa++ = 0x8"先将成员变量 b 的值修改为 0x8，然后将指针 pa 指向了 b 后面的内存单元。此时，打印出的信息为"n.a = 0x7 n.b = 0x8"。"*pb++ = 0x9"，先将 pb 所指向的成员变量 b 的值修改为 0x9，再将 pb 指针指向变量 b 后面的内存单元，之后"*pb++ = 0xa"对 b 后面的内存单元进行了修改，不影响 n 的变量值。此时，屏幕打印输出"n.a = 0x7 n.b = 0x9"。



7、在思考题中，结构体 node 中每个成员变量分别是 char、int、short 型，都占用不同大小的内存空间，但是每个成员变量的首地址都是类型所占字节数的整数倍。而整个结构体的长度将会是成员变量中占用字节数最多的变量类型字节数的整数倍。成员变量之间的空字节将有系统自动使用 cc 填充。所以，当执行到语句 n.a = 0x12345; n.b = 0x23456; n.c = 0x34567; 时，此时的成员变量 a = 0x45; b = 023456; c = 0x4567; 语句 short *p = (short *)((int)&n + 1);将使的 short 型的指针 p 指向结构体 n 首地址偏移一个字节的位置。* ++p = 1，指针先移动一个字节，指向了偏离首地址 2 个字节的位置，再进行赋值，使得该位置的两个字节分别填充 0x01 和 0x00。* ++p = 10，同理。此时，指针将偏离首地址 5 个字节数。执行 *p++ = 20 语句时，先进行赋值，将先前的值覆盖掉，再移动指针。所以，最后打印出来的结果将会是，n.a = 0x45, n.b = 0x1400, n.c = 0x4567.

思考题


指针 short p
char a
int b
short c
struct node n
struct node n
n.a = 0x12345
n.b = 0x23456
n.c = 0x34567
short * p = (short *) ((int) &n + 1)
* ++p = 1
* ++p = 10
* p++ = 20
n.a = 0x45
n.b = 0x1400
n.c = 0x4567

指导教师批阅意见：

成绩评定：

指导教师签字：

年　　月　　日

备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。