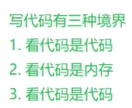
1.

int main()
{
int a[5] = {1, 2, 3, 4, 5};
int \*ptr = (int \*)(&a + 1); //&a+1指向5后面 跳过整个数组
printf("%d,%d", \*(a+1), \*(ptr - 1));
return 0;
}//2 5



刚开始我看山是山、看海是海

后来我看山不是山、看海不是海

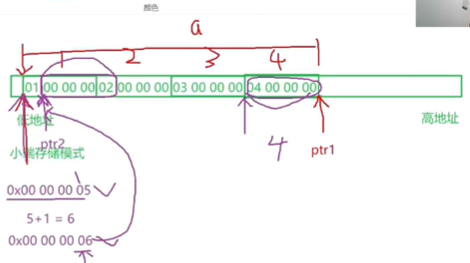
最后啊我看山还是山、看海还是海

2.

struct Test
{
int Num;
char\* pcName;
short sDate;
char cha[2];
short sBa[4];
}\*p;
//假设p的值为0x100000 如下表达式的值是多少
//已知 结构体Test类型的变量大小是20个字节
int main()
{
p = (struct Test)0x100000;
printf("%p\n", p+0x1); 0x100000 + 20
printf("%p\n", (unsigned long)p+0x1); //10485767 + 1 = 10485767十进制
printf("%p\n", (unsigned int\*)p+0x1); //0x100000 + 4
}
//0x00100014 0x00100001 0x00100004

3.

int main()
{
int a[4] = {1,2,3,4}; //小端模式 01000000 020000000 03000000 04000000
int \*ptr1 = (int \*)(&a + 1); //0x4
int \*ptr2 = (int \*)((int)a + 1); //0x20000000
printf("%x, %x", ptr1[-1], \*ptr2);
return 0;
}

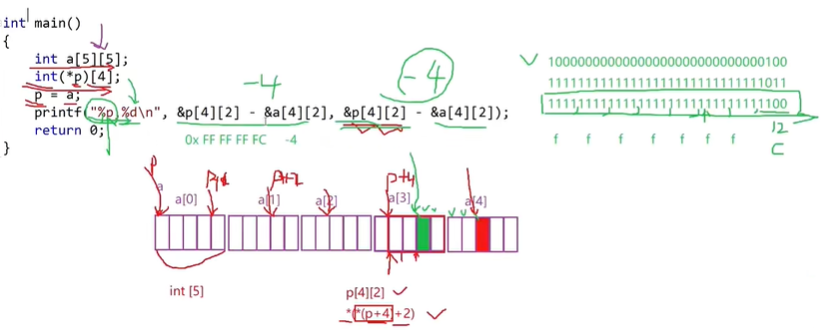


4.

int main()
{
int a[3][2] = { (0,1), (2,3), (4,5)}; //逗号表达式 {1，3，5}
int \*p;
p = a[0];
printf("%d", p[0]);
return 0;
}//1

5.

int main()
{
int a[5][5];
int (\*p)[4];
p = a;
printf("p, %d" &p[4][2] - &a{4}[2], &p[4][2] - &a[4][2]);
return 0;
} 0xff ff ff fc \ -4



6.

int main()
{
int aa[2][5] = {1,2,3,4,5,6,7,8,9,10};
int \*ptr1 = (int \*)(&aa + 1); //跳过整个二维数组
int \*ptr2 = (int \*)(\*(aa+1)); //首元素也就是第一行的地址+1 跳到第二行首元素
printf("%d,%d", \*(ptr1 - 1), \*(ptr2 - 1));
return 0;
}//10 5

7.

int main()
{
char \*a[] = {"work", "at", "alibaba"}; //将首字符的地址放到a中
char \*\*pa = a;
pa++;
printf("%s\n", \*pa);
return 0;
} // at

8.

int main()
{
char \*c[] = {"ENTE", "NEW", "POINT", "FIRST"};
char\*\*cp[] = {c+3, c+2, c+1, c};
char\*\*\*cpp = cp;
printf("%s\n", \*\*++cpp); //point
printf("%s\n", \*--\*++cpp + 3);
printf("%s\n", \*cpp[-2] + 3);
printf("%s\n", cpp[-1][-1] + 1);
return 0;
}