

printf("p = %d\n",p);

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

printf("q = %d\n",q);

printf("\*q = %d\n",\*q);

printf("\*\*q = %d\n",\*\*q);

printf("m = %d\n",m);

printf("\*m = %d\n",\*m);

printf("\*\*m = %d\n",\*\*m);

printf("\*\*\*m = %d\n",\*\*\*m);

结果：

p = 6618636

\*p = 5q = 6618624

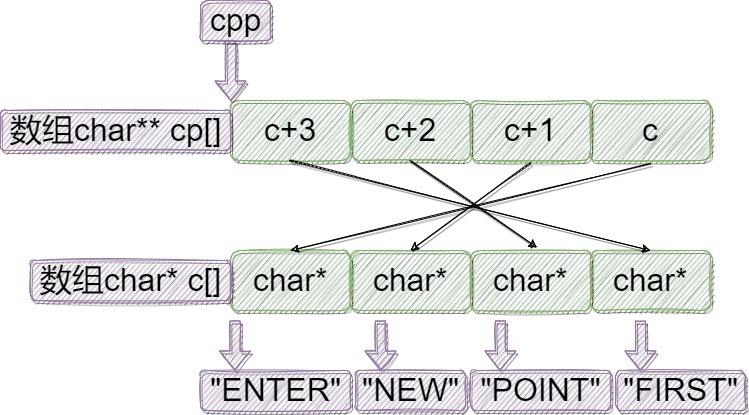
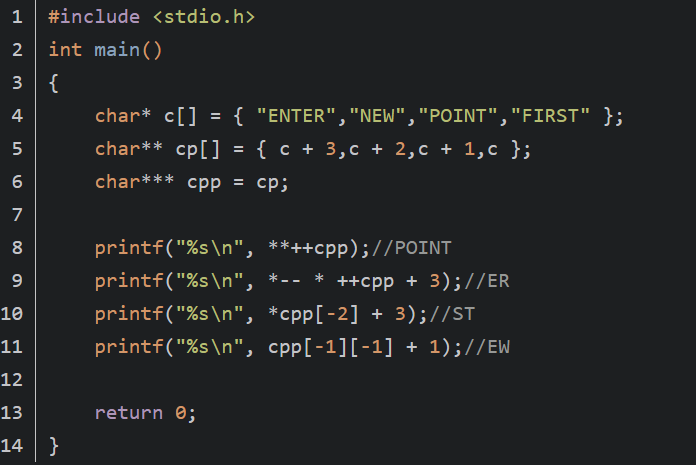
\*q = 6618636

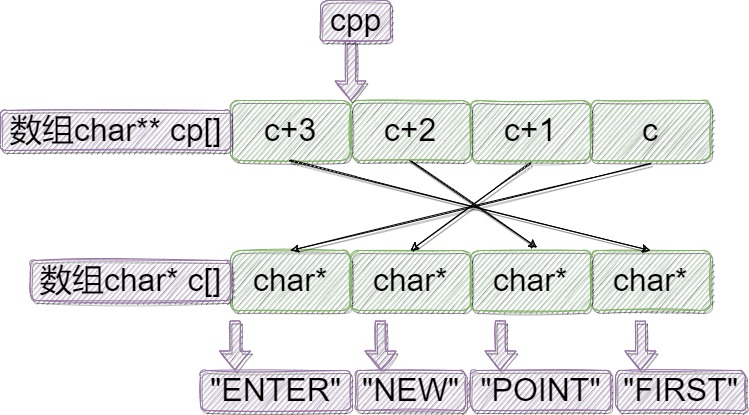
\*\*q = 5m = 6618616

\*m = 6618624

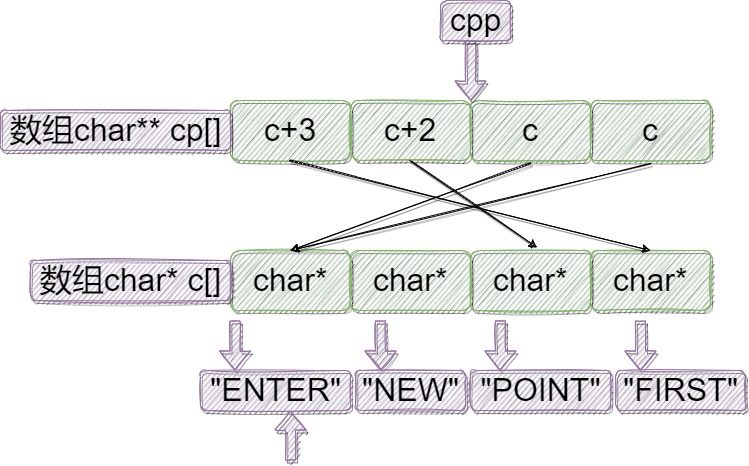
\*\*m = 6618636

\*\*\*m = 5

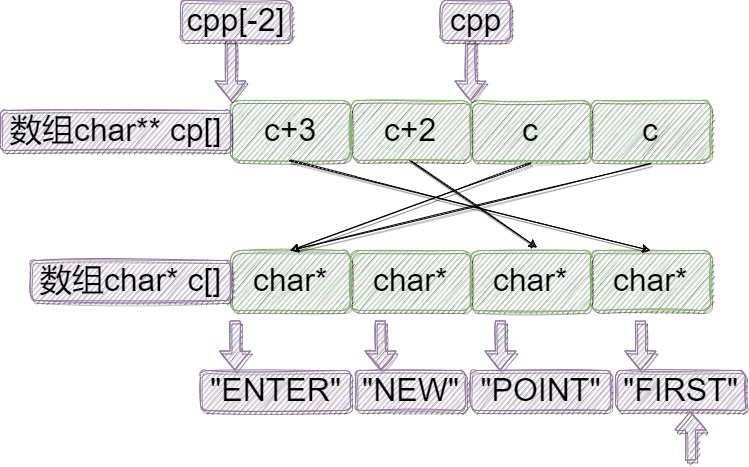


printf("%s\n", \*\*++cpp)，首先cpp的地址+1，再连续两次解引用，第一次解引用找到c+1，再次解引用找到c[3]，即"POINT"的地址，以%s打印即为POINT。

printf("%s\n", \*-- \* ++cpp + 3)，首先cpp的地址+1，再解引用找到的是cp[3],再将cp[3]中的地址-1，再解引用找到c[0],再加3表示指向ENTER的指针右移三个字节，即为ER



printf("%s\n", \*cpp[-2] + 3)，首先cpp[-2]可以看成\*(cpp-2)，即cp[0],再次解引用，即为指向FIRST的指针，再加3，变为指向S的指针，以%s打印，即为ST



printf("%s\n", cpp[-1][-1] + 1)，可以转化为\*(\*(cpp-1)-1)+1,即为EW，注意此处的cpp和cp[1]的值是不改变的。

