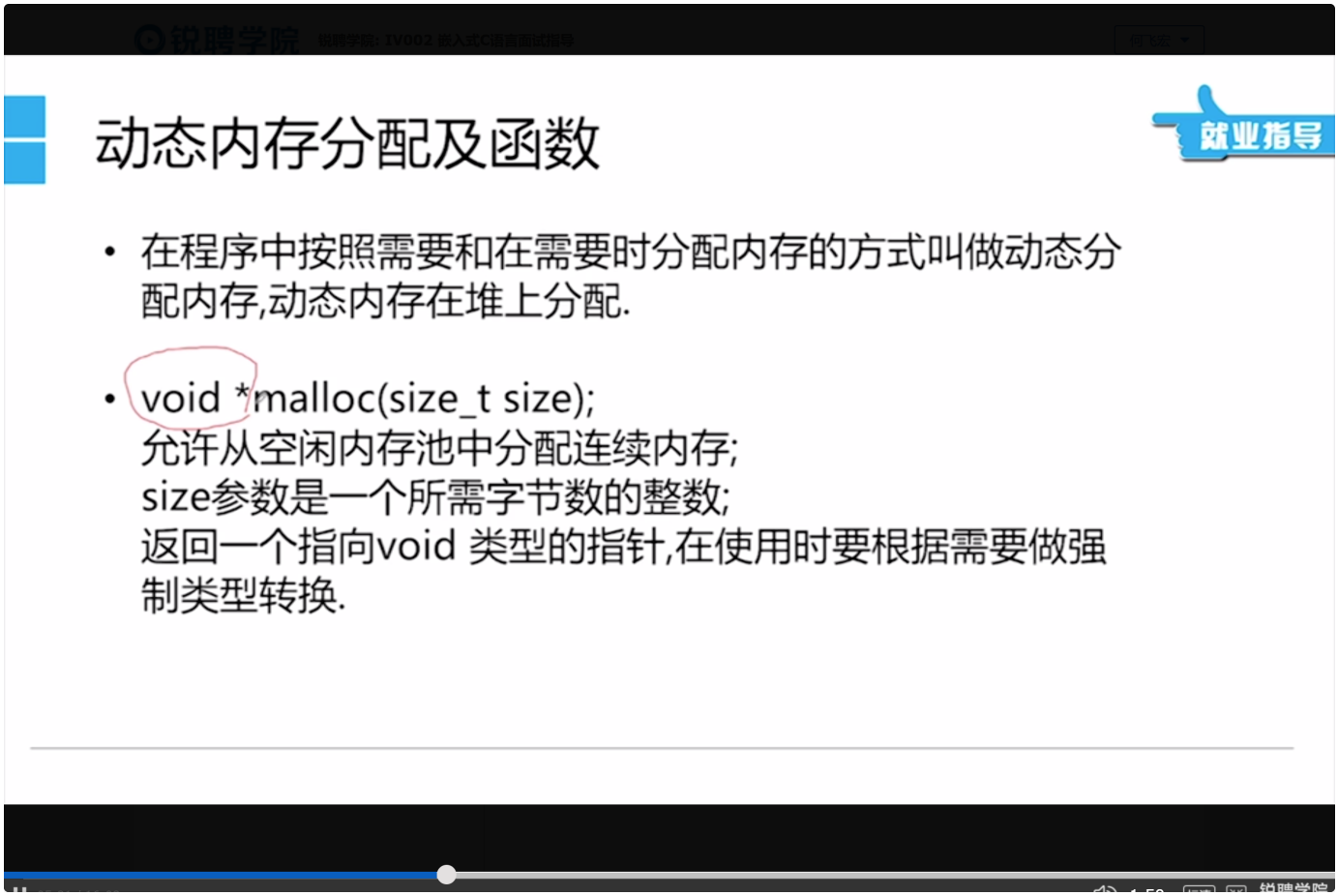
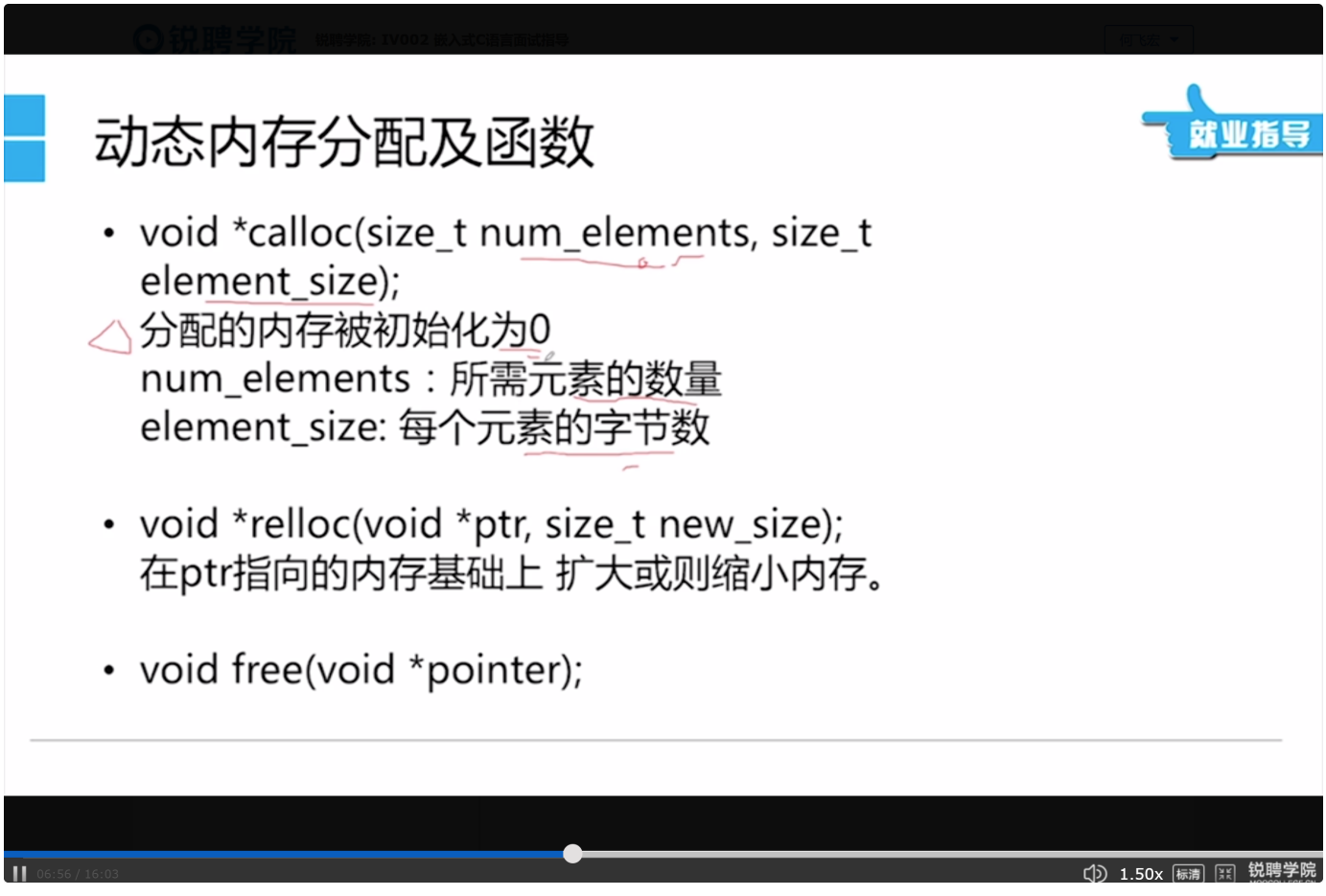

锐聘学院
动态内存分配及函数
就业指导
• 在程序中按照需要和在需要时分配内存的方式叫做动态分配内存,动态内存在堆上分配.
• void *malloc(size_t size);
允许从空闲内存池中分配连续内存;
size参数是一个所需字节数的整数;
返回一个指向void 类型的指针,在使用时要根据需要做强制类型转换.


锐聘学院
动态内存分配及函数
就业指导
• void *calloc(size_t num_elements, size_t element_size);
分配的内存被初始化为0
num_elements：所需元素的数量
element_size: 每个元素的字节数
• void *relloc(void *ptr, size_t new_size);
在ptr指向的内存基础上 扩大或则缩小内存。
• void free(void *pointer);
06:56 / 16:03
1.50x
标清
锐聘学院

函数指针

```
int Func(int x);   /*声明一个函数*/
int (*p) (int x);  /*定义一个函数指针*/
p = Func;          /*将Func函数的首地址赋给指针变量p*/
```

函数名本身就是该函数的地址

因此赋值时函数 Func 不带括号，也不带参数。由于函数名 Func 代表函数的首地址，因此经过赋值以后，指针变量 p 就指向函数 Func() 代码的首地址了。

此后，便可以使用这个函数指针(p)来调用对应的函数(Func)了。

```
# include <stdio.h>
int Max(int, int);  //函数声明
int main(void)
{
    int(*p)(int, int);  //定义一个函数指针
    int a, b, c;
    p = Max;  //把函数Max赋给指针变量p, 使p指向Max函数
    printf("please enter a and b:");
    scanf("%d%d", &a, &b);
    c = (*p)(a, b);  //通过函数指针调用Max函数
    printf(" a = %d\n b = %d\n max = %d\n", a, b, c);
    return 0;
}
int Max(int x, int y)  //定义Max函数
{
    int z;
    if (x > y)
    {
        z = x;
    }
    else
    {
        z = y;
    }
    return z;
}
```

使用时像程序那样即可：

- 函数指针 + 参数列表

避免产生野指针

- 野指针的成因
  指针变量没有被初始化。
  任何指针变量刚被创建时不会自动成为NULL指针，它的缺省值是随机的，它会乱指一气。所以，指针变量在创建的同时应当被初始化，要么将指针设置为NULL，要么让它指向合法的内存。
  char *p = NULL;
  char *str = (char *) malloc(100);
  指针p被free或者delete之后，没有置为NULL，让人误以为p是个合法的指针。