

- we have a vector library
- we have a list library
- I'd like to use both to read in & store a list of materials — see (P. 21) in notes example materials.

```
material green
{
  ambient 0 5 0
}
```

```
material yellow
{
  diffuse 4 4 0
  ambient 5 4 0
  specular 1 1 1
}
```

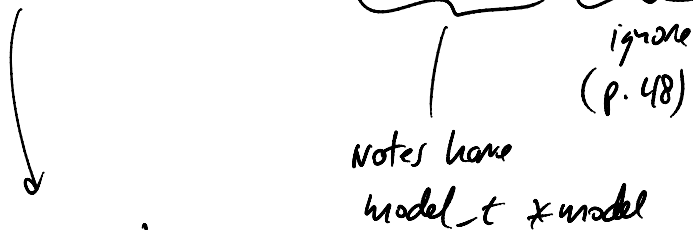
```
in material.h
#ifndef NAME_LEN
#define NAME_LEN 16
#endif
```

- our material data type (p. 47) `#define MAT_COOKIE 32456123`

```
typedef struct material_type
{
  int cookie; // like a magic number — distinct id
  char name [NAME_LEN];
  drgb_t ambient;
  drgb_t diffuse;
  drgb_t specular;
} material_t // from pixel.h a double [3] just like vec_t
```

- we also want material functions (methods):

```
material_init(FILE *in, list_t *list, int attributes)
```



idea is to:

1. malloc() a material\_t struct
2. load material attributes (read FILE \*in) `material_t *mat; mat = (material_t *) malloc(sizeof(material_t));`

```

2. load material attributes material_t *mat;
   (read FILE *in) mat = (material_t *) malloc(sizeof(material_t));
   assert(mat != NULL);
   (calls material_load_attributes(in, mat);

```

```

3. add material mat to list
   list_add(list, (void *)mat);

```

```

material_load_attributes(FILE *in, material_t *mat)
{

```

```

// parse file as
material name \n
{

```

```

    ambient r g b\n
    diffuse r g b\n
    specular r g b\n
}

```

```

    fscanf(in, "%s", mat->name);
    while((c = fgetc(in)) != EOF && c != '\n');

```

```

    while((c = fgetc(in)) != EOF && c != '{');

```

```

    // acquire
    fscanf(in, "%s", attrname);

```

```

    char attrname[NAME_LEN]

```

```

    if(!strcmp(attrname, "ambient"))
    "    "
    "    "
    "    "

```

```

    vrc_read(in, mat->ambient);

```

put into loop that reads next char

loops while that char != '}'

```

while((c = fgetc(in)) != EOF && c != '}')
{
    fputc(m, c);
}

```

other stuff:

material\_get\_by\_name

material\_list\_print

objects of lab3:  
need in mat.txt

∴ write it out (copy)

---

get more stuffs

material - get ambient (material + x mat,  
drght + dot)

↑

P.51 " get diffize

" get speckel

fill in this  
double r g b dot  
with ambient r g b

---

these are known as accessors