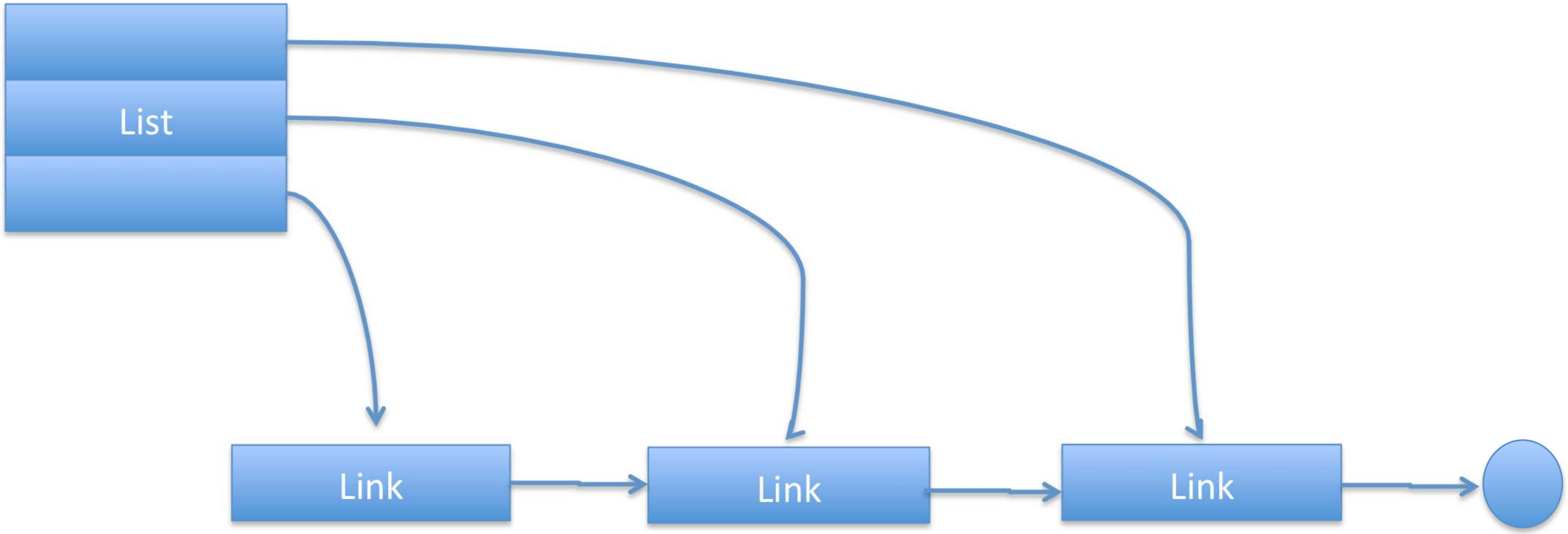
Object Oriented C

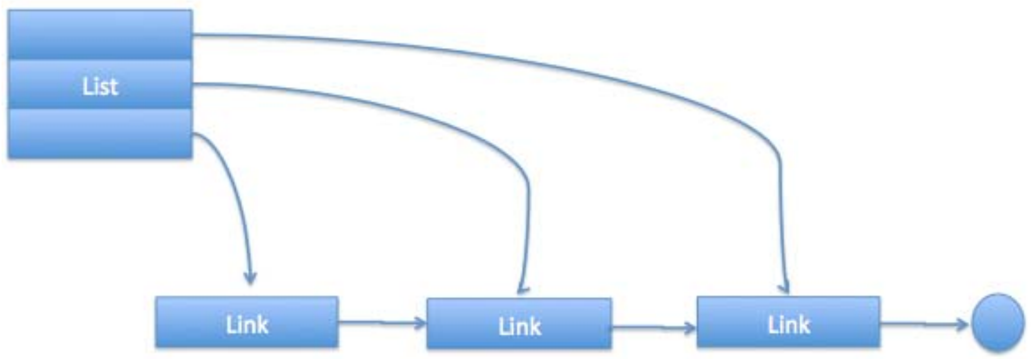
CpSc102 - Fall 2010

The List



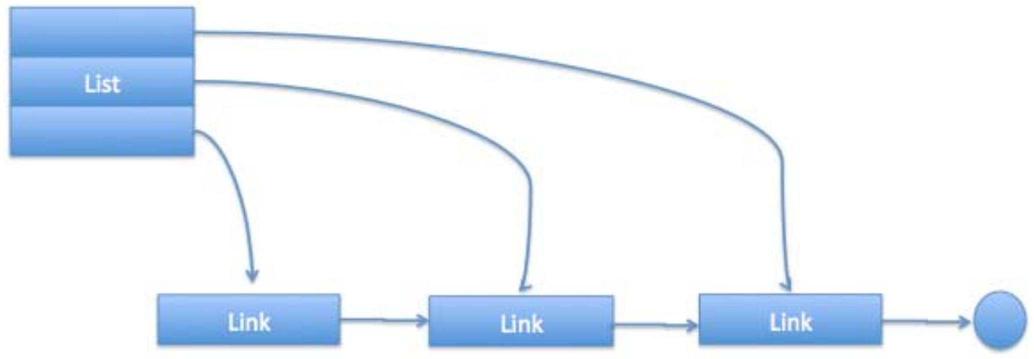
List Structure

link_t	*first
link_t	*last
link_t	*current

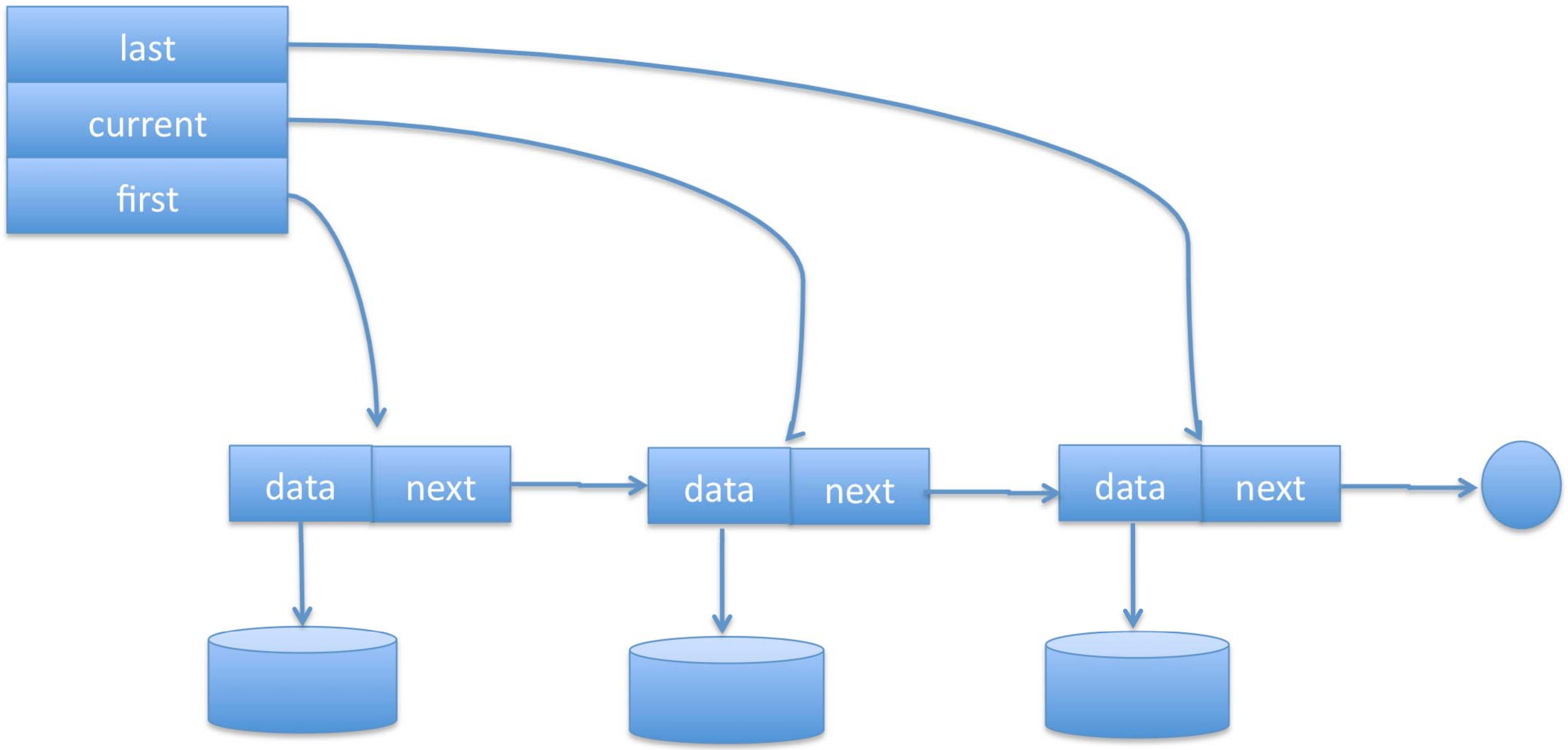


Link Structure

void	*data
link_t	*next

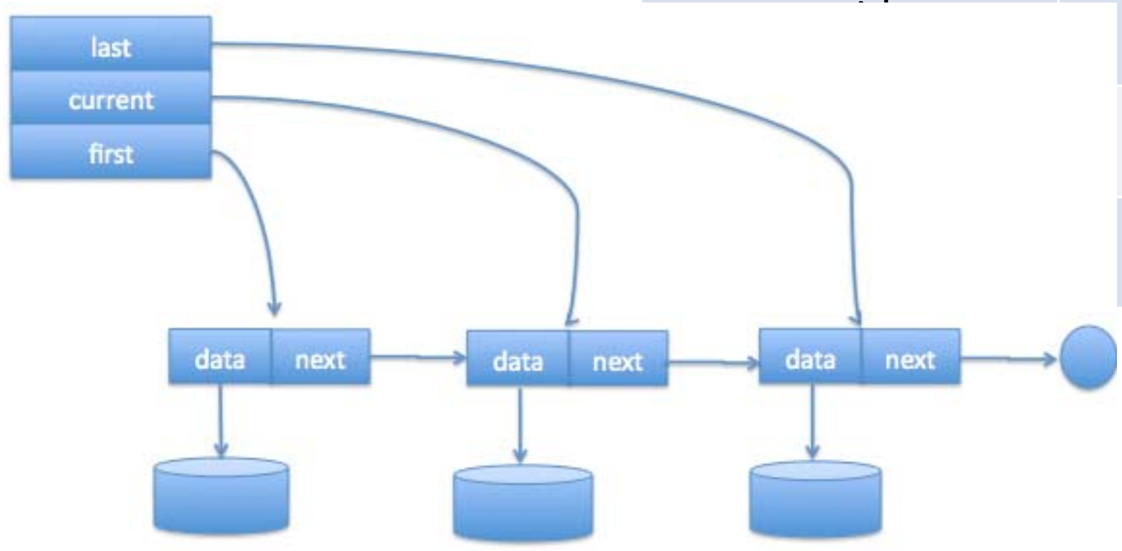


The List

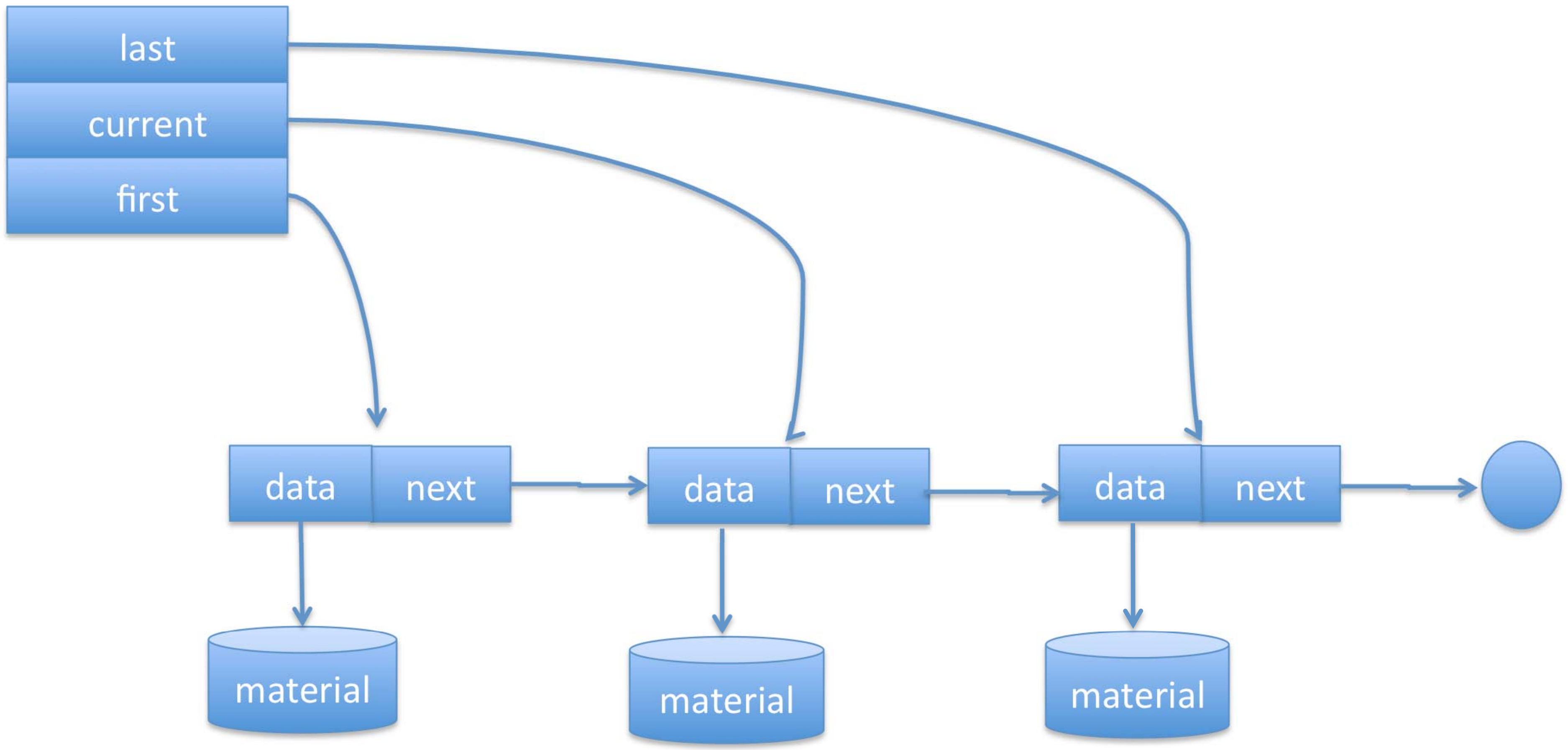


List Management Functions

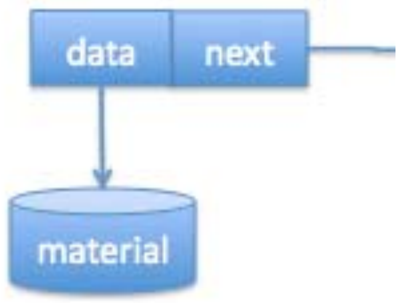
list_t*	list_init()
void	list_add(list_t*, void *)
void	list_del_front_link(list_t *)
void	list_del(list_t*)
void	list_reset(list_t*)
	list_not_end(list_t*)
	list_next_link(list_t*)
	list_get_data(list_t)



The Material List



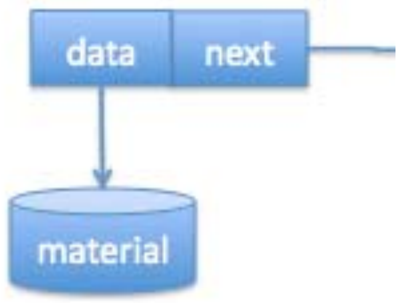
Material Structure



```
material green
{
   ambient 0 6 0
   diffuse   0 7 0
  specular 1 1 1
}
```

int	cookie
char	name[16]
drgb_t	ambient
drgb_t	diffuse
drgb_t	specular

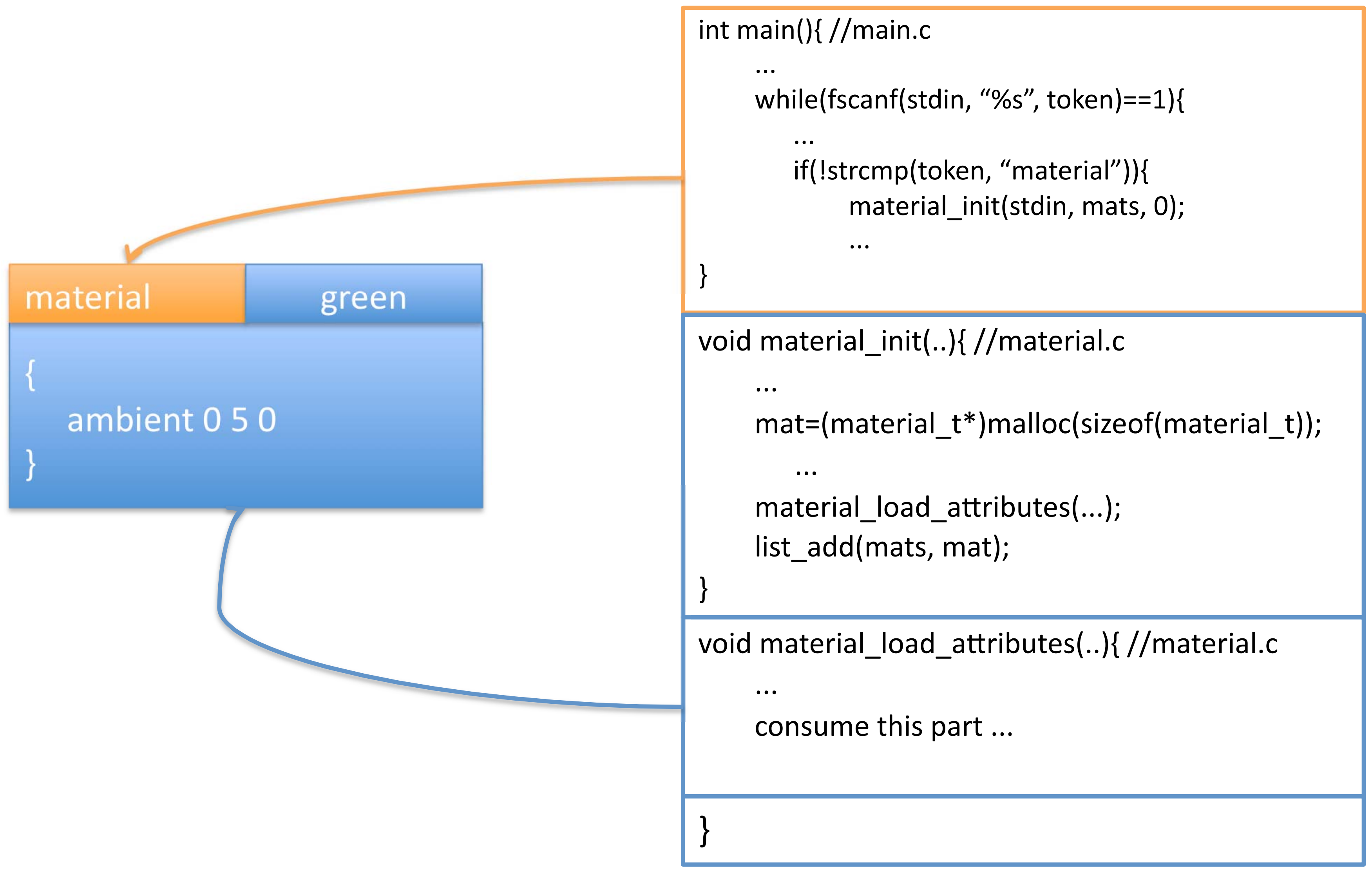
Material Functions



```
material green
{
   ambient 0 6 0
   diffuse   0 7 0
  specular 1 1 1
}
```

void	material_init(FILE*, list_t*, int)
void	material_load_attributes(FILE*, material_t*)
material_t*	material_getbyname(list_t*, char*)
void	material_print(material_t*, FILE*)
void	material_list_print(list_t*, FILE*)
char*	material_getname(material_t*)
void	material_getamb(material_t*, drgb_t)
void	material_getdiff(material_t*, drgb_t)
void	material_getspec(material_t*, drgb_t)

Material Input



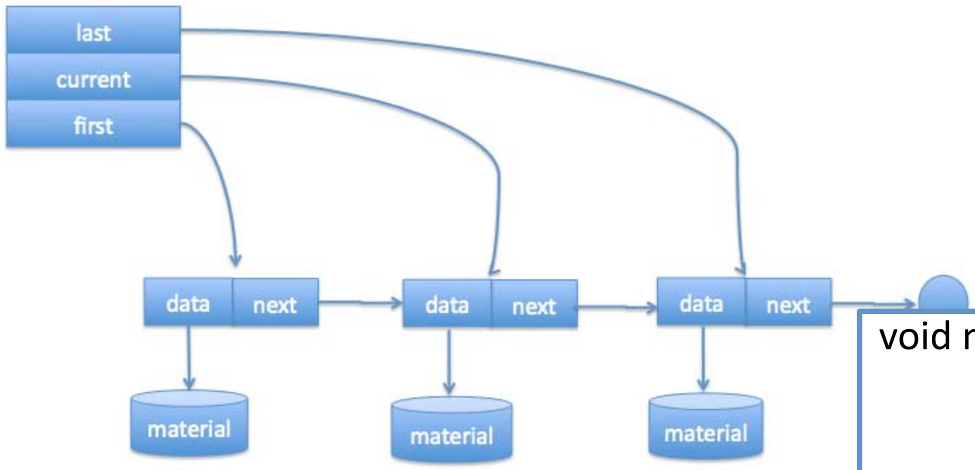
```
material    green
{
    ambient 0 5 0
}
```

```
int main(){ //main.c
        ...
        while(fscanf(stdin, “%s”, token)==1){
            ...
            if(!strcmp(token, “material”)){
                material_init(stdin, mats, 0);
                ...
}
```

```
void material_init(..){ //material.c
        ...
        mat=(material_t*)malloc(sizeof(material_t));
            ...
        material_load_attributes(...);
        list_add(mats, mat);
}
```

```
void material_load_attributes(..){ //material.c
        ...
        consume this part ...

}
```

Material Output



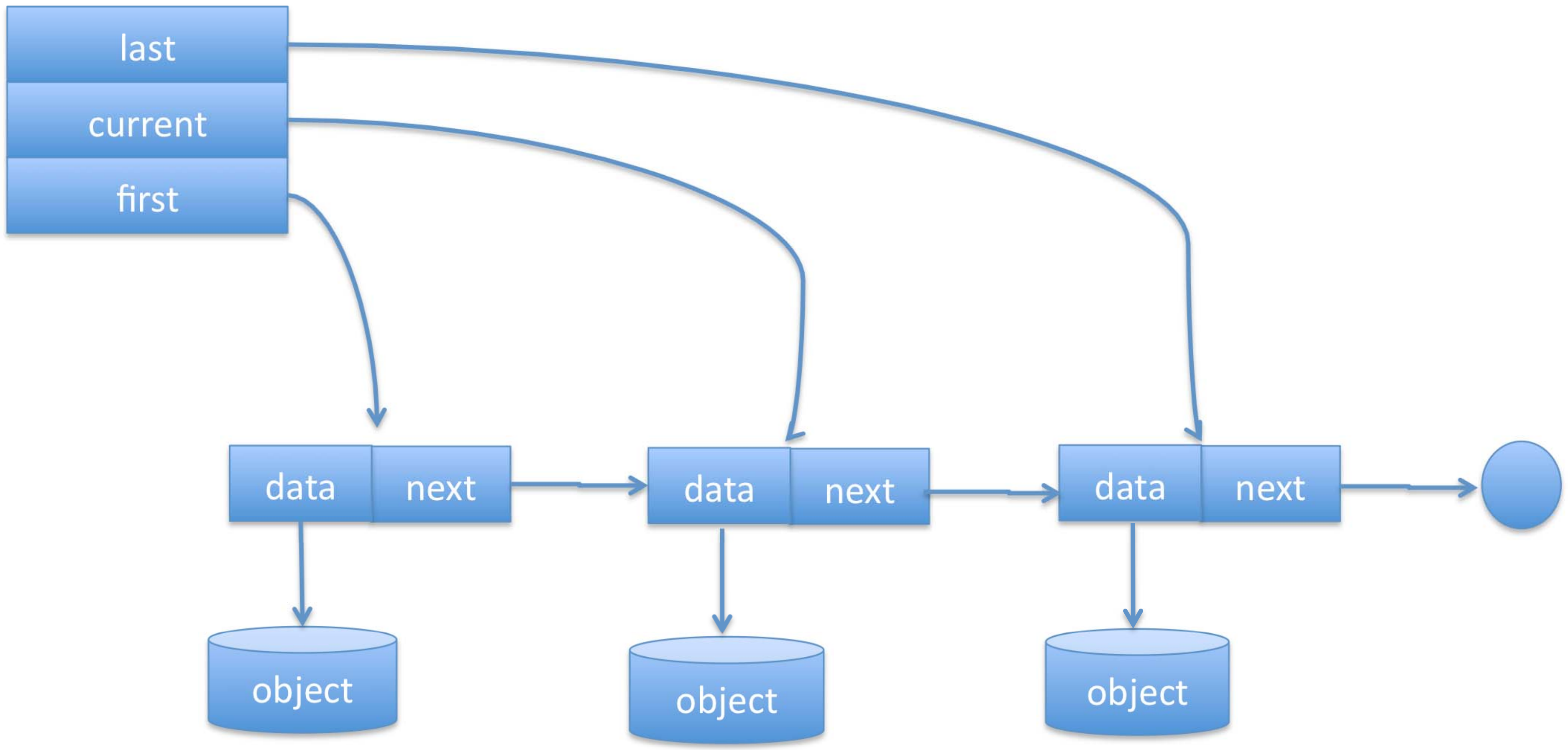
```
int main(...){ //main.c
    //load materials and objects
    ...
    //print out all materials
    material_list_print(mats, stdout);
        ...
```

```
void material_list_print(...){ //material.c
     while(list_not_end(mats){
          mat=(material_t*)list_get_data(mats);
          material_print(mat, out);
          list_next_link(mats);
     }
}
```

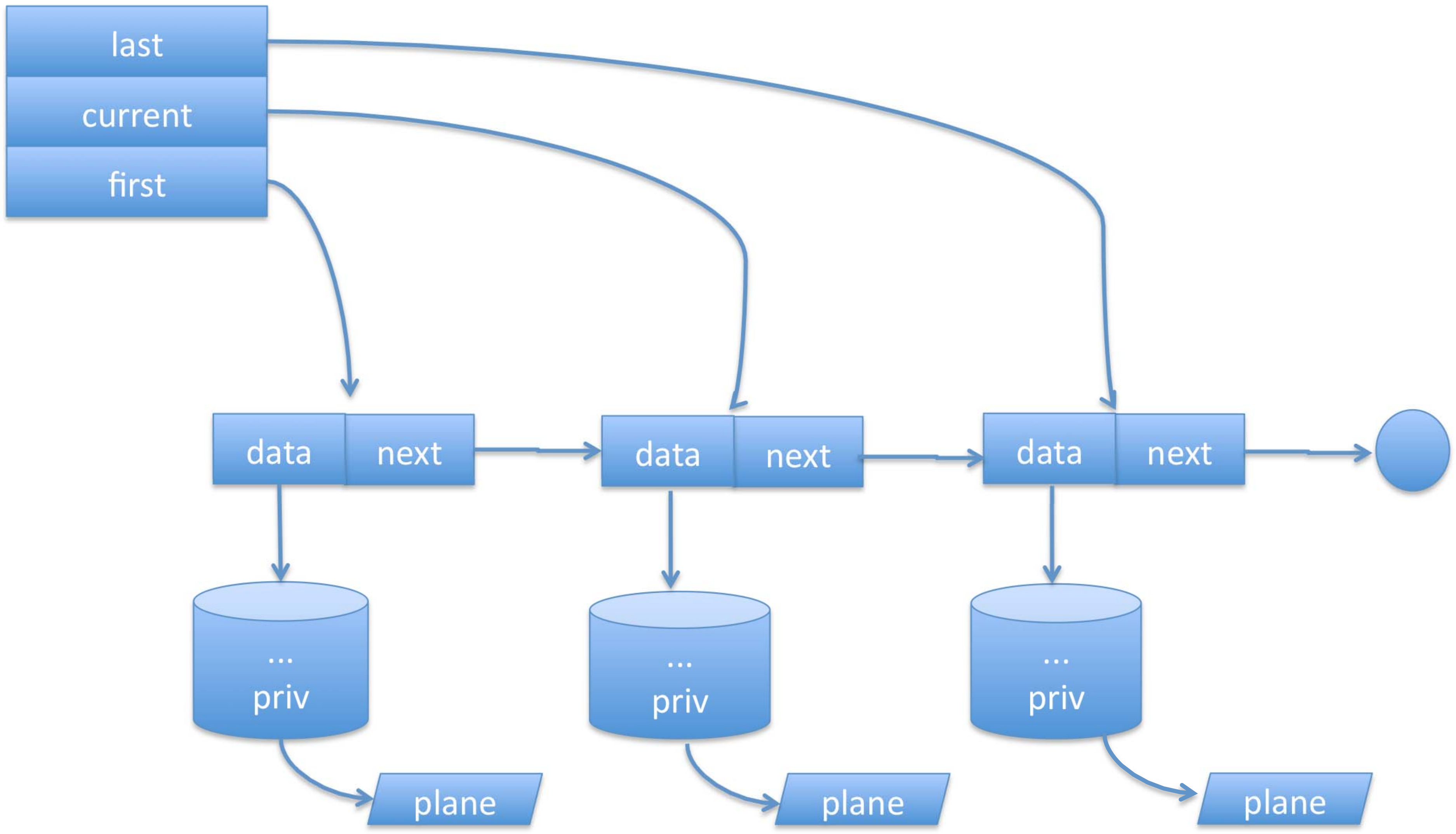
```
material    green
{
    ambient 0 5 0
}
```

```
void material_print(...){ //material.c
    fprintf(out, "material %s\n", mat->name);
    fprintf(out,"{\n");
    if(pix_nonzero(mat->ambient))
       pix_print(out, "ambient", mat->ambient);
    ...
}
```

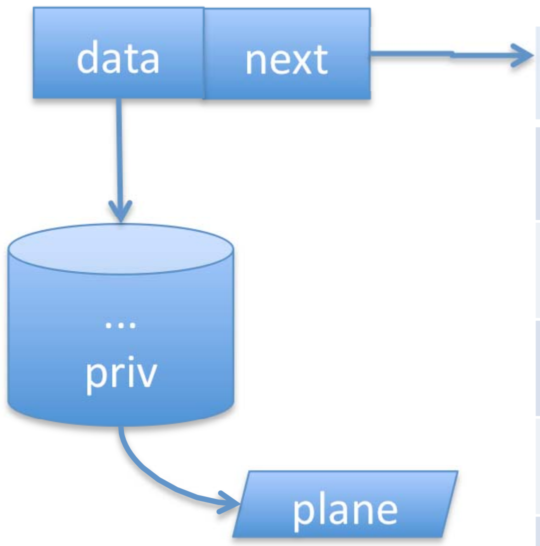
The Object List



The Object List

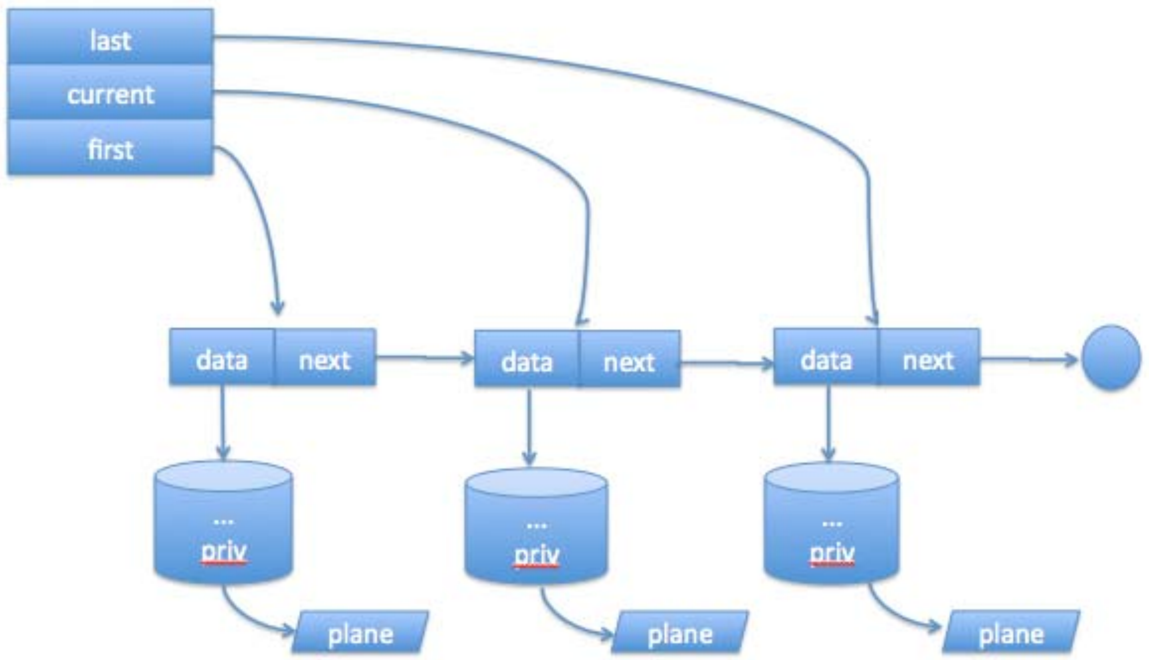


Object Structure



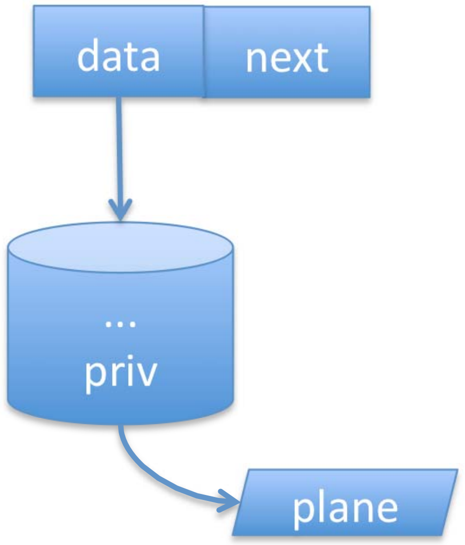
int	cookie
char	type[16]
char	name[16]
material_t*	mat
void	(*printer)(object_t*, FILE*)
double	(*hits)(object_t*, vec_t, vec_t)
void	(*ambient)(material_t*, drgb_t)
void	(*diffuse)(material_t*, drgb_t)
void	(*specular)(material_t*, drgb_t)
void*	priv
vec_t	last_hit
vec_t	last_normal

Object Functions



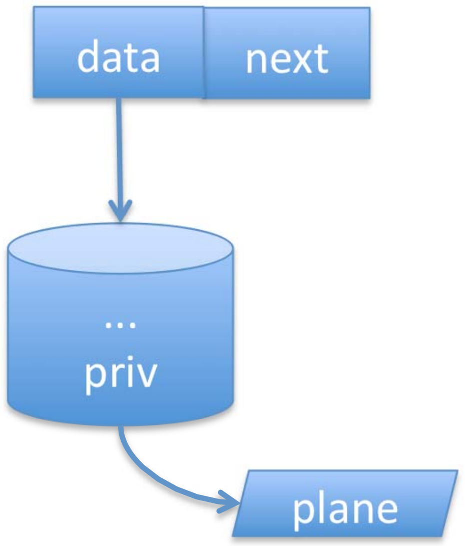
void	object_init(FILE*, list_t*, list_t*)
double	object_no_hit(object_t*, vec_t, vec_t)
void	object_list_print(list_t*, FILE*)
void	object_print(object_*, FILE*)
char*	object_getname(objec_t*)

Plane Structure



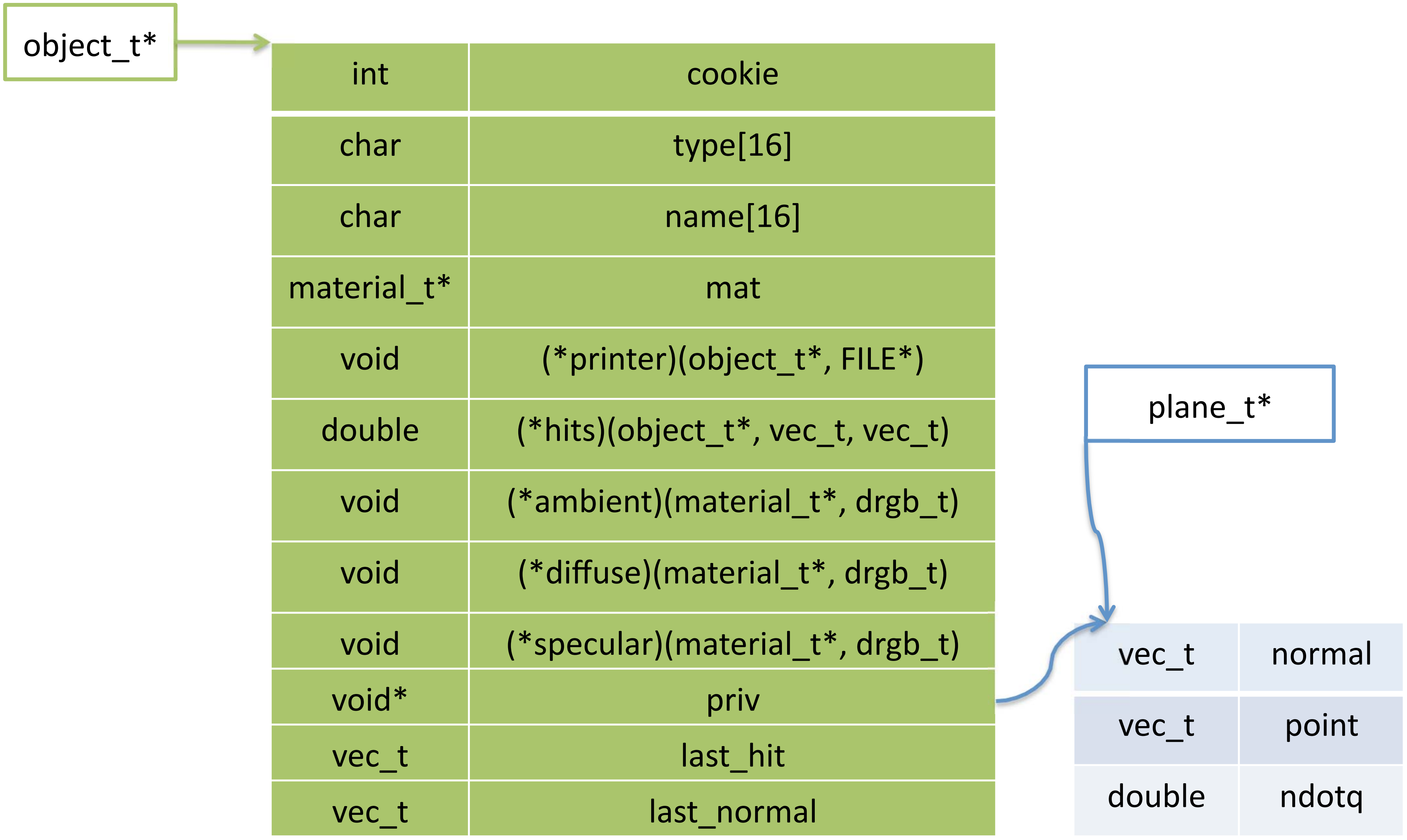
vec_t	normal
vec_t	point
double	ndotq

Plane Functions



void	plane_init(FILE*, list_t*, list_t*, int)
void	plane_print(object_t*, FILE*)
double	plane_hits(object_t*, vec_t, vec_t)

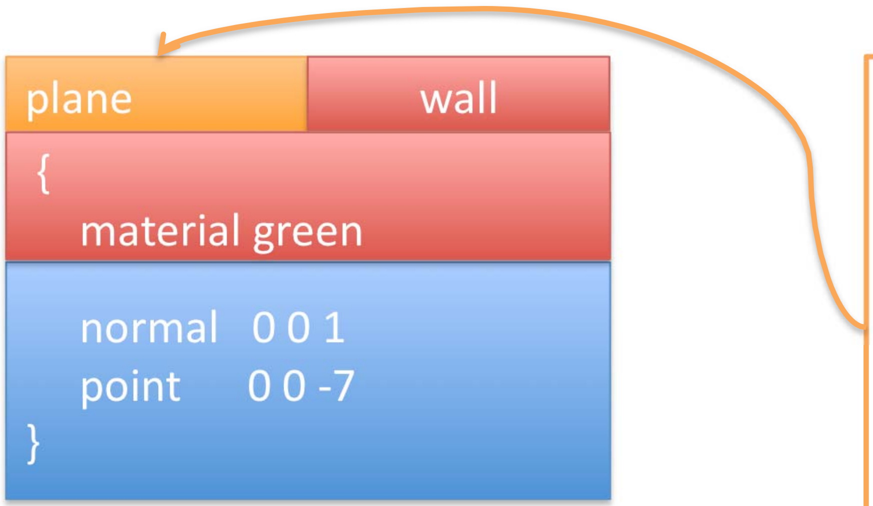
Disjoint Memory



int	cookie
char	type[16]
char	name[16]
material_t*	mat
void	(*printer)(object_t*, FILE*)
double	(*hits)(object_t*, vec_t, vec_t)
void	(*ambient)(material_t*, drgb_t)
void	(*diffuse)(material_t*, drgb_t)
void	(*specular)(material_t*, drgb_t)
void*	priv
vec_t	last_hit
vec_t	last_normal

vec_t	normal
vec_t	point
double	ndotq

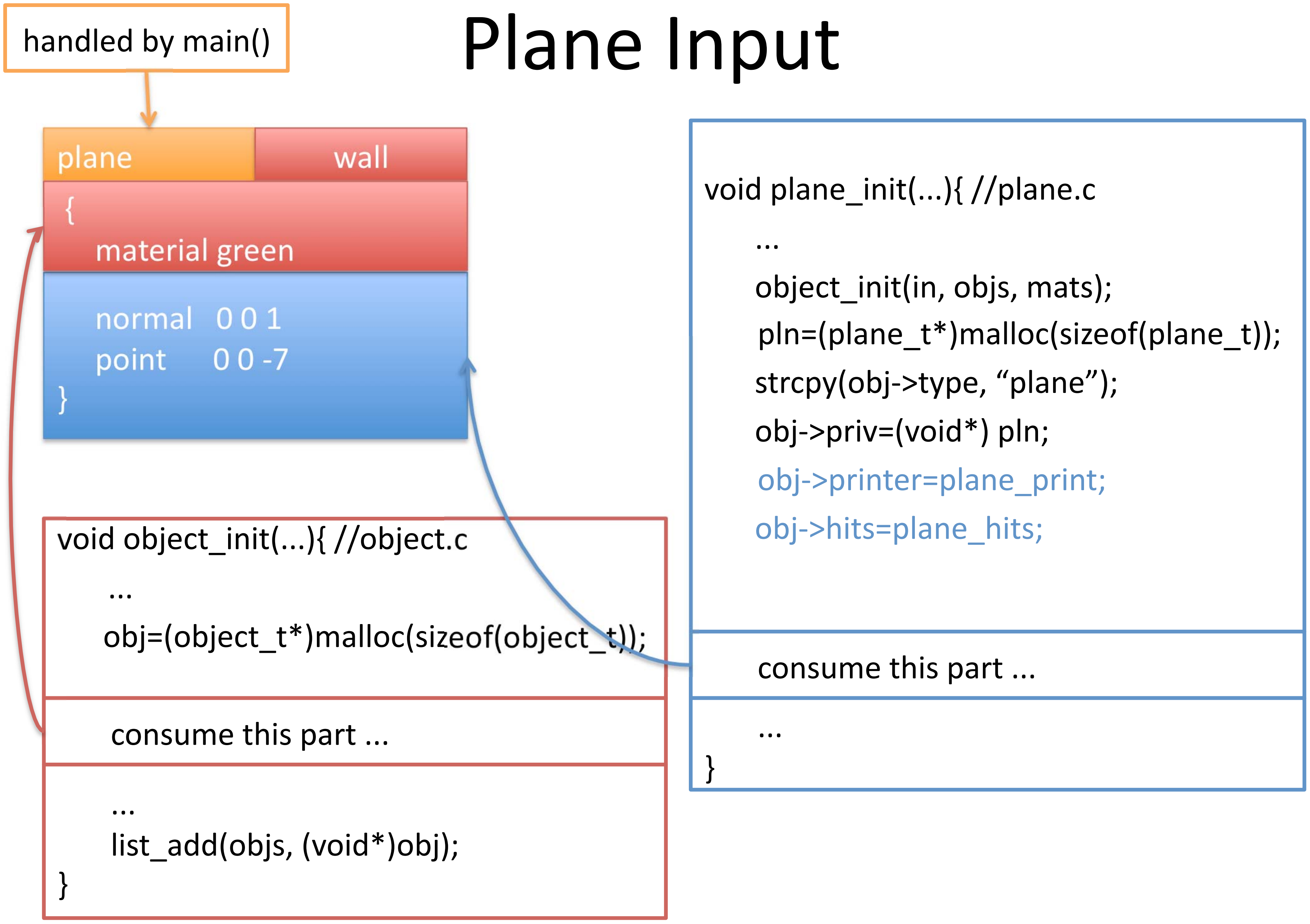
Plane Input



```
plane                wall
{
    material green
    normal   0 0 1
    point    0 0 -7
}
```

```
int main(){ //main.c
    ...
    while(fscanf(stdin, "%s", token)==1){
        ...
        if(!strcmp(token, "plane")){
            plane_init(stdin, objs, mats, 0);
            ...
}
```

Plane Input

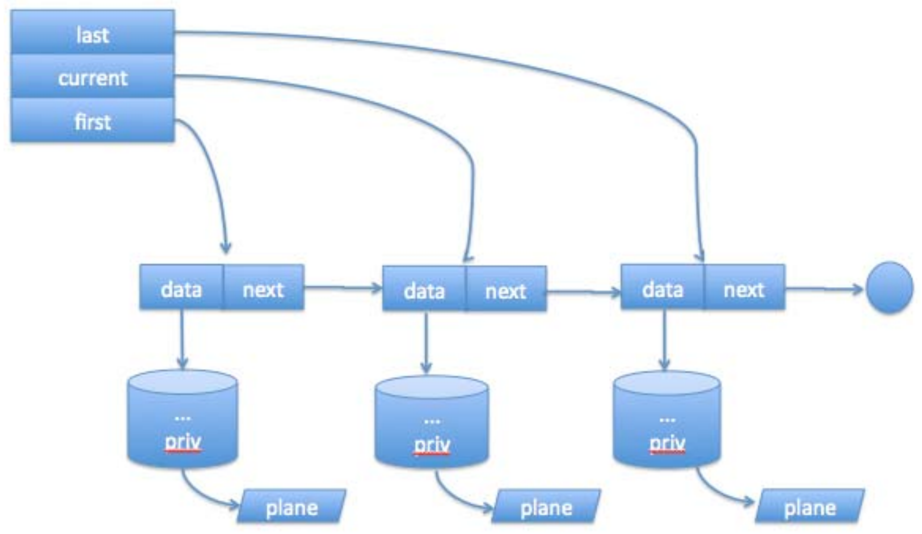


```
plane    wall
{
    material green
    normal   0 0 1
    point    0 0 -7
}
```

```
void object_init(...){ //object.c
    ...
    obj=(object_t*)malloc(sizeof(object_t));

    consume this part ...

    ...
    list_add(objs, (void*)obj);
}
```

```
void plane_init(...){ //plane.c
    ...
    object_init(in, objs, mats);
    pln=(plane_t*)malloc(sizeof(plane_t));
    strcpy(obj->type, “plane”);
    obj->priv=(void*) pln;
    obj->printer=plane_print;
    obj->hits=plane_hits;

    consume this part ...

    ...
}
```

Plane Output



```
int main(...){ //main.c
       //load materials and objects
      ...
      //print out all materials
      object_list_print(objs, stdout);
            ...
```

```
void object_list_print(...){ /object.c
        while(list_not_end(objs){
                obj=(object_t*)list_get_data(objs);
                obj->printer(obj, out);
                list_next_link(objs);
        }
}
```

plane	wall
{ material green	
normal 0 0 1 point 0 0 -7 }	

What is obj->printer?

obj->printer==plane_print!!!

Plane output

```
plane		wall
{
	material green
	normal	0 0 1
	point	0 0 -7
}
```

```
void plane_print(...){ //plane.c
	object_print(obj, out);
	plane_t *pln=(plane_t*)obj->priv;
	pix_print(out, “normal”, pln->normal);
	pix_print(out, “point”, pln->point);
}
```

```
void object_print(...){ //object.c
	fprintf(out, “%s %s\n”, obj->type, obj->name);
	fprintf(out, “{\n”);
	fprintf(out, “ %s %s\n”, “material”, material_getname(obj->mat));
}
```

New Plane Structure

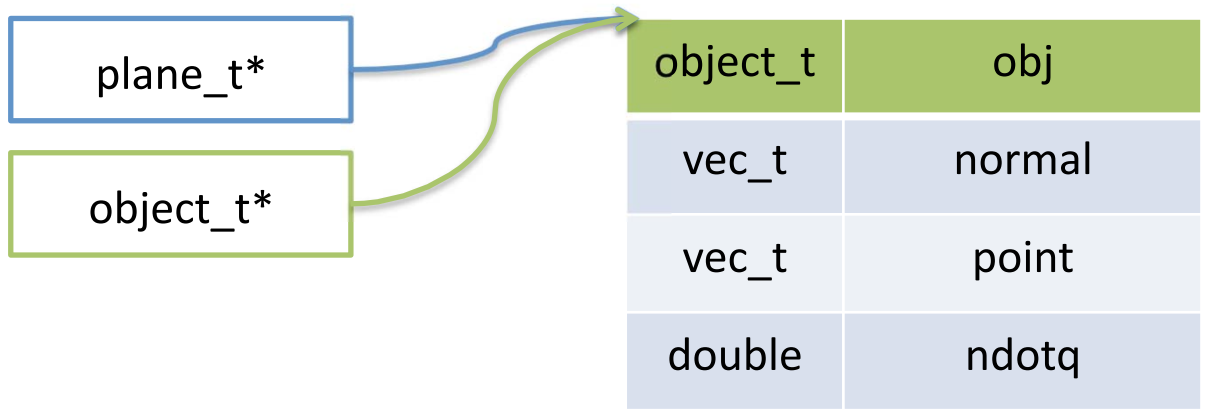
old!

```
typedef struct plane_type
{
   vec_t normal;
   vec_t point;
   double ndotq;
}plane_t;
```

new!

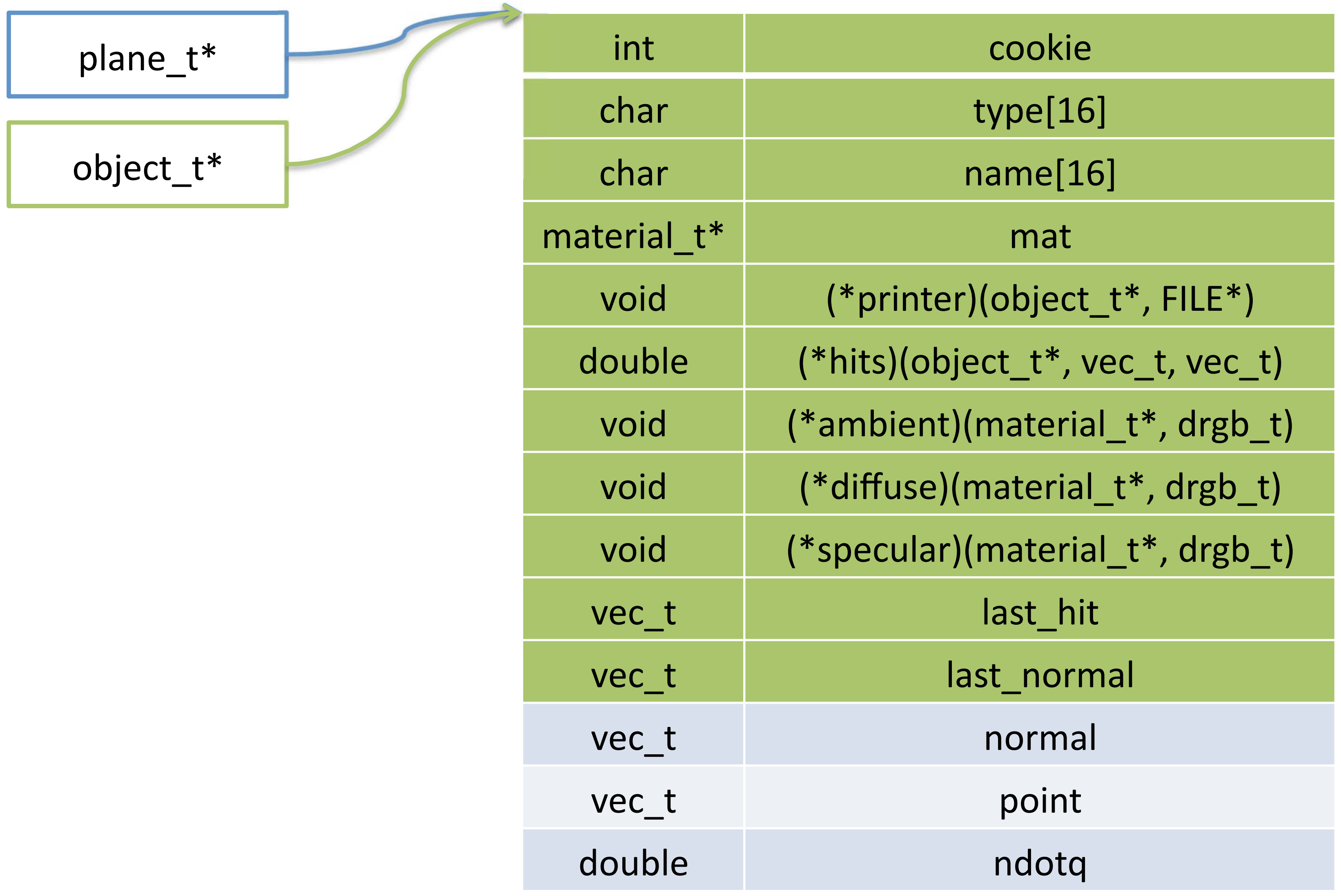
```
typedef struct plane_type
{
    object_t obj;
   vec_t normal;
   vec_t point;
   double ndotq;
}plane_t;
```

New Plane Structure



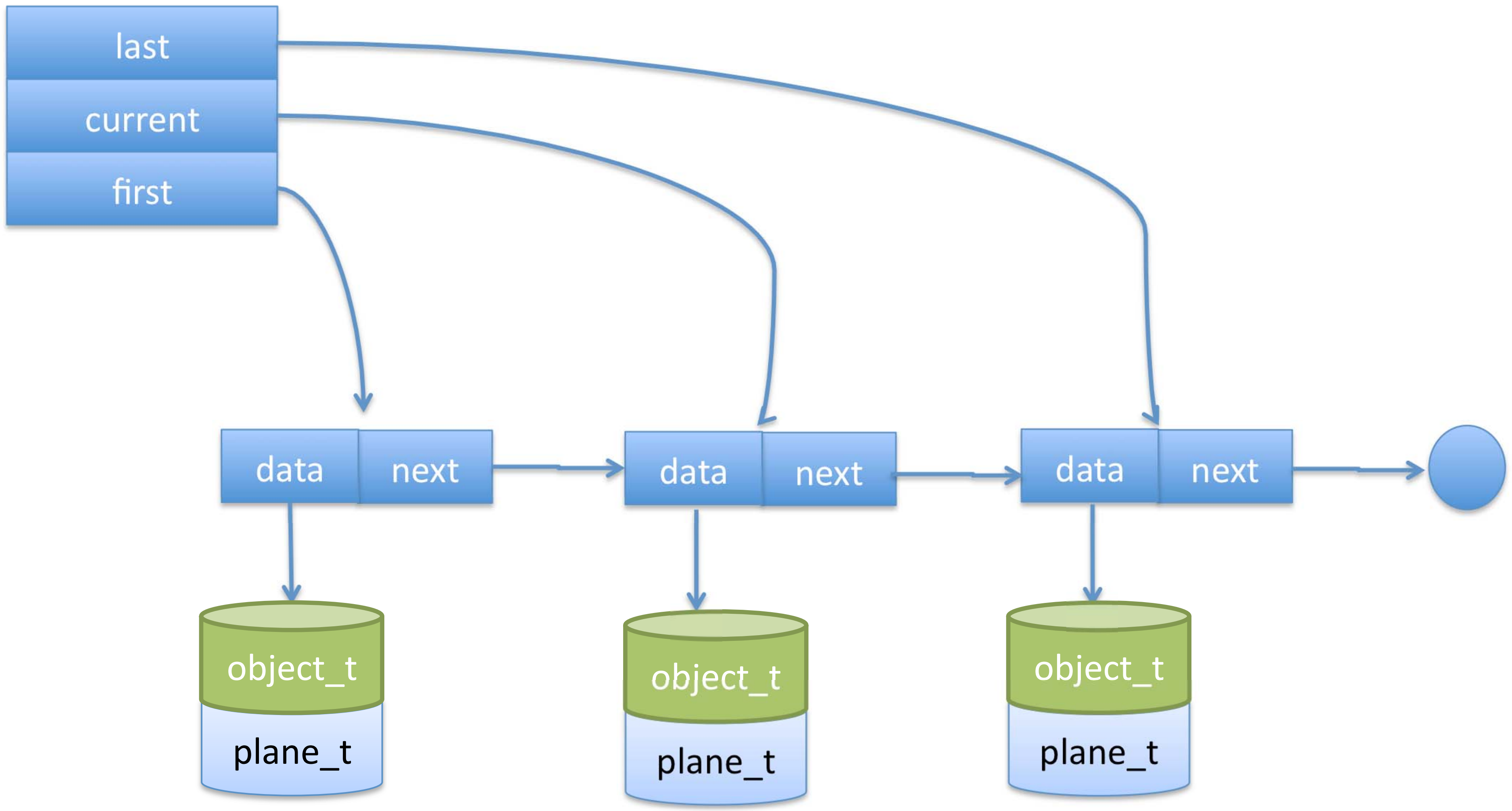
object_t	obj
vec_t	normal
vec_t	point
double	ndotq

Contiguous Memory

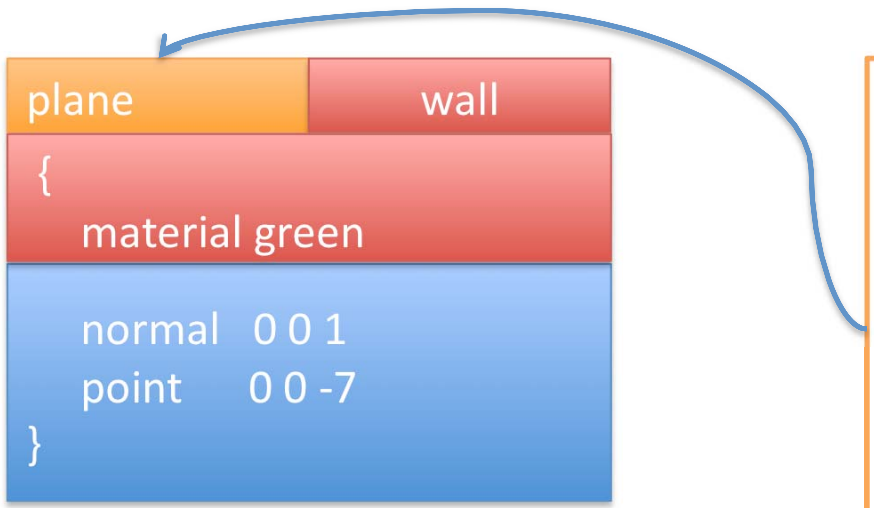


int	cookie
char	type[16]
char	name[16]
material_t*	mat
void	(*printer)(object_t*, FILE*)
double	(*hits)(object_t*, vec_t, vec_t)
void	(*ambient)(material_t*, drgb_t)
void	(*diffuse)(material_t*, drgb_t)
void	(*specular)(material_t*, drgb_t)
vec_t	last_hit
vec_t	last_normal
vec_t	normal
vec_t	point
double	ndotq

New Object List



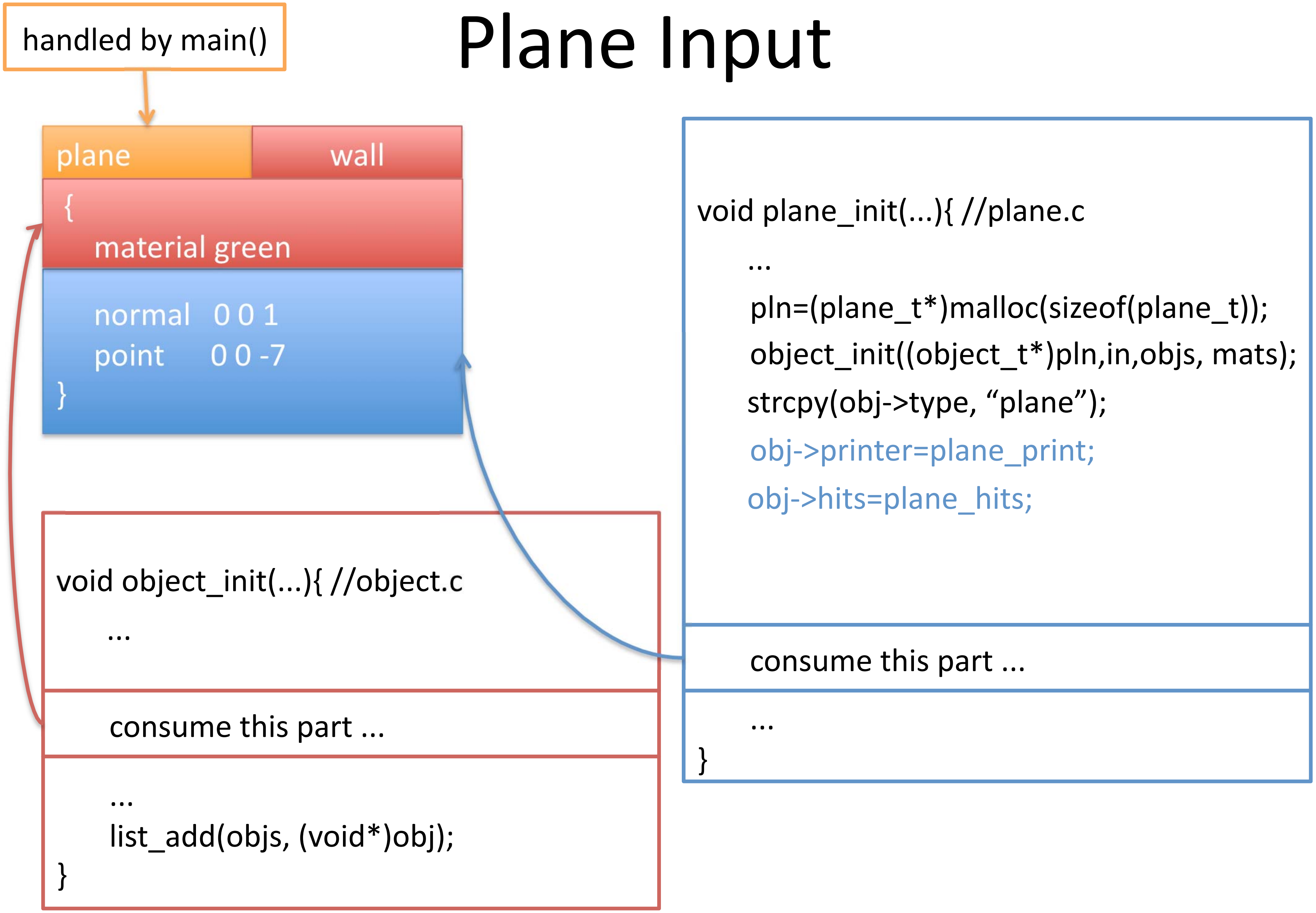
New Plane Input



```
plane          wall
{
    material green
    normal  0 0 1
    point   0 0 -7
}
```

```
int main(){ //main.c
    ...
    while(fscanf(stdin, “%s”, token)==1){
        ...
        if(!strcmp(token, “plane”)){
            plane_init(stdin, objs, mats, 0);
            ...
}
```

Plane Input

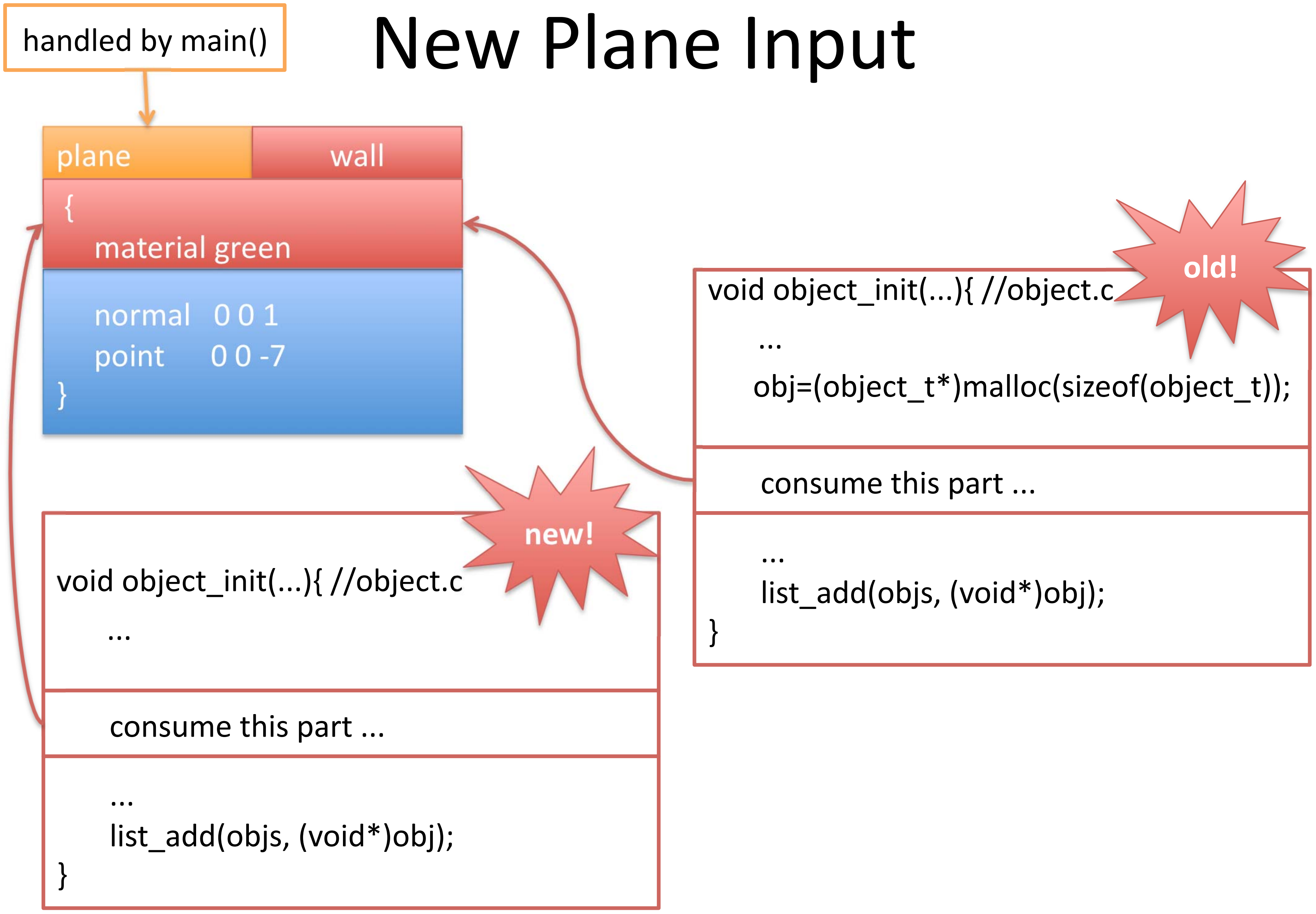


```
plane     wall
{
    material green
    normal   0 0 1
    point    0 0 -7
}
```

```
void object_init(...){ //object.c
     ...
     consume this part ...
     ...
     list_add(objs, (void*)obj);
}
```

```
void plane_init(...){ //plane.c
      ...
      pln=(plane_t*)malloc(sizeof(plane_t));
      object_init((object_t*)pln,in,objs, mats);
      strcpy(obj->type, “plane”);
      obj->printer=plane_print;
      obj->hits=plane_hits;
      consume this part ...
      ...
}
```

New Plane Input



handled by main()

```
plane	wall
{
	material green
	normal	0 0 1
	point	0 0 -7
}
```

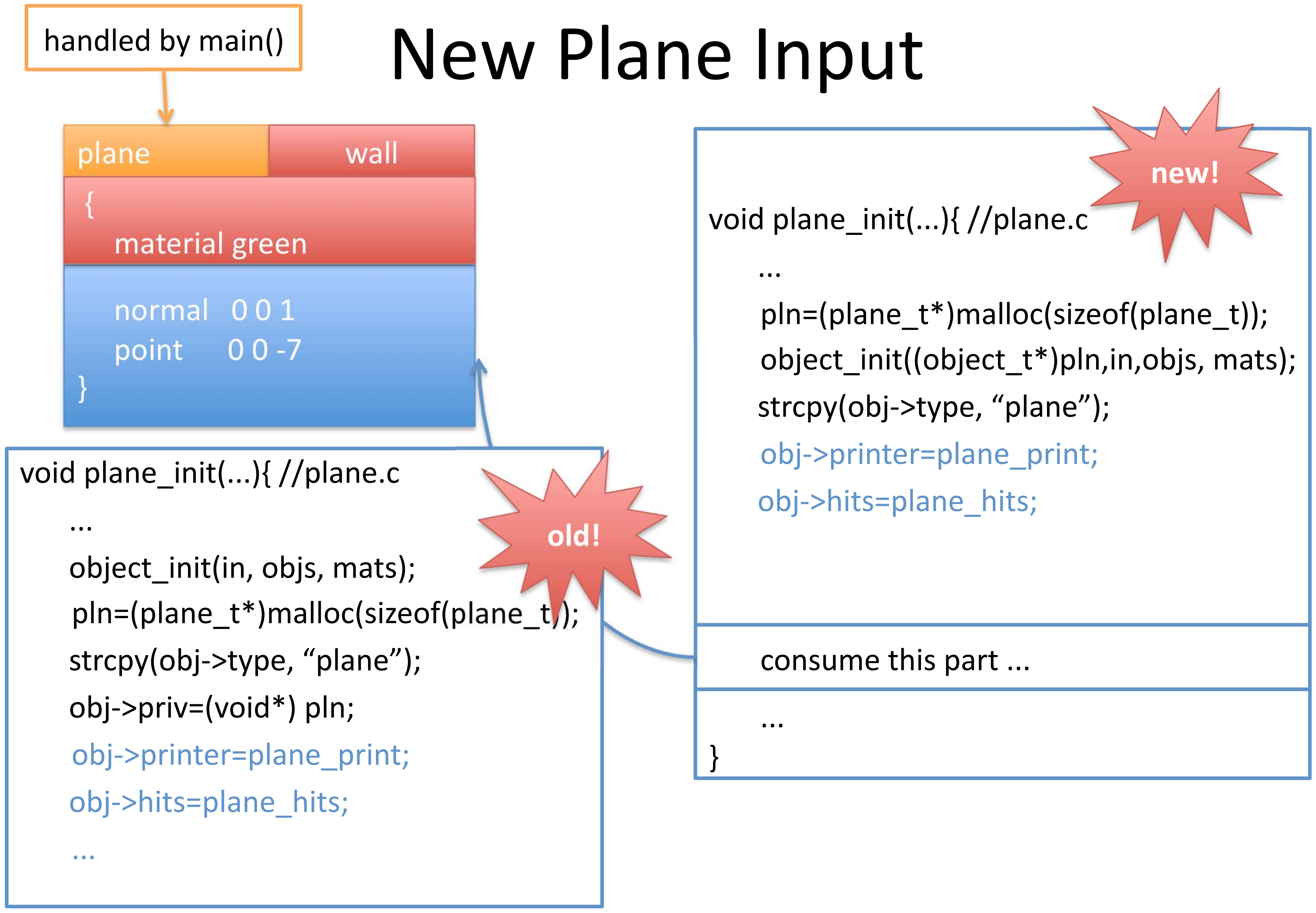
new!

```
void object_init(...){ //object.c
	...
	consume this part ...
	...
	list_add(objs, (void*)obj);
}
```

old!

```
void object_init(...){ //object.c
	...
	obj=(object_t*)malloc(sizeof(object_t));
	consume this part ...
	...
	list_add(objs, (void*)obj);
}
```

New Plane Input

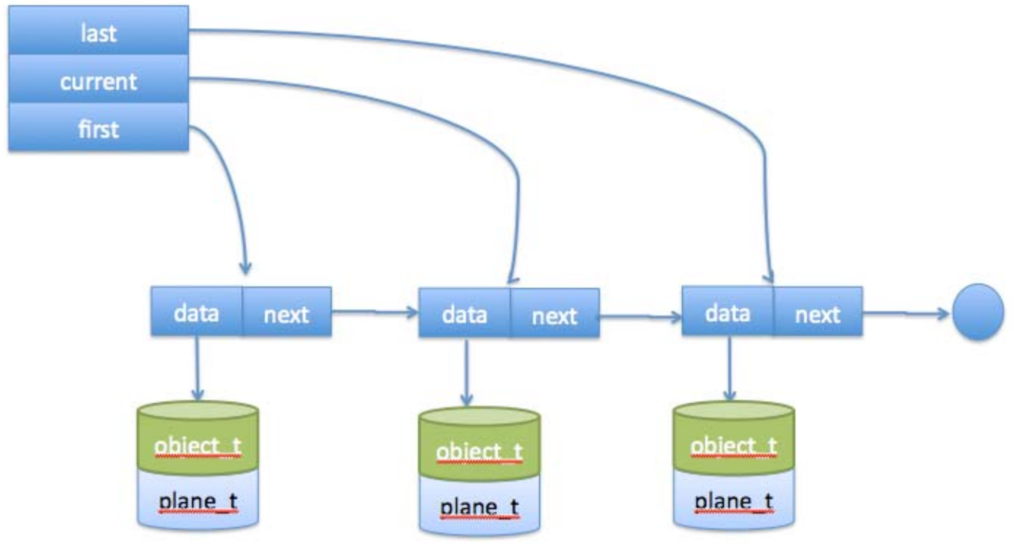


```
plane    wall
{
    material green
    normal  0 0 1
    point   0 0 -7
}
```

```
void plane_init(...){ //plane.c
    ...
    object_init(in, objs, mats);
    pln=(plane_t*)malloc(sizeof(plane_t));
    strcpy(obj->type, "plane");
    obj->priv=(void*) pln;
    obj->printer=plane_print;
    obj->hits=plane_hits;
    ...
```

```
void plane_init(...){ //plane.c
    ...
    pln=(plane_t*)malloc(sizeof(plane_t));
    object_init((object_t*)pln,in,objs, mats);
    strcpy(obj->type, "plane");
    obj->printer=plane_print;
    obj->hits=plane_hits;

    consume this part ...

    ...
}
```

New Plane Output



```
int main(...){ //main.c
        //load materials and objects
        ...
        //print out all materials
        object_list_print(objs, stdout);
            ...
```

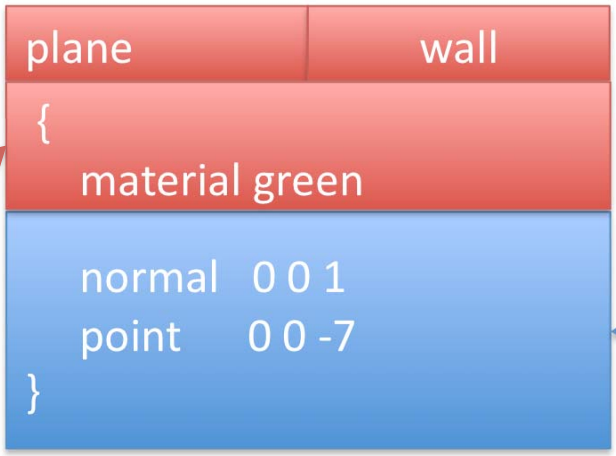
```
void object_list_print(...){ /object.c
        while(list_not_end(objs){
                obj=(object_t*)list_get_data(objs);
                obj->printer(obj, out);
                list_next_link(objs);
        }
}
```

```
plane          wall
{
    material green

    normal   0 0 1
    point    0 0 -7
}
```

What is obj->printer?

obj->printer==plane_print!!!

New Plane output



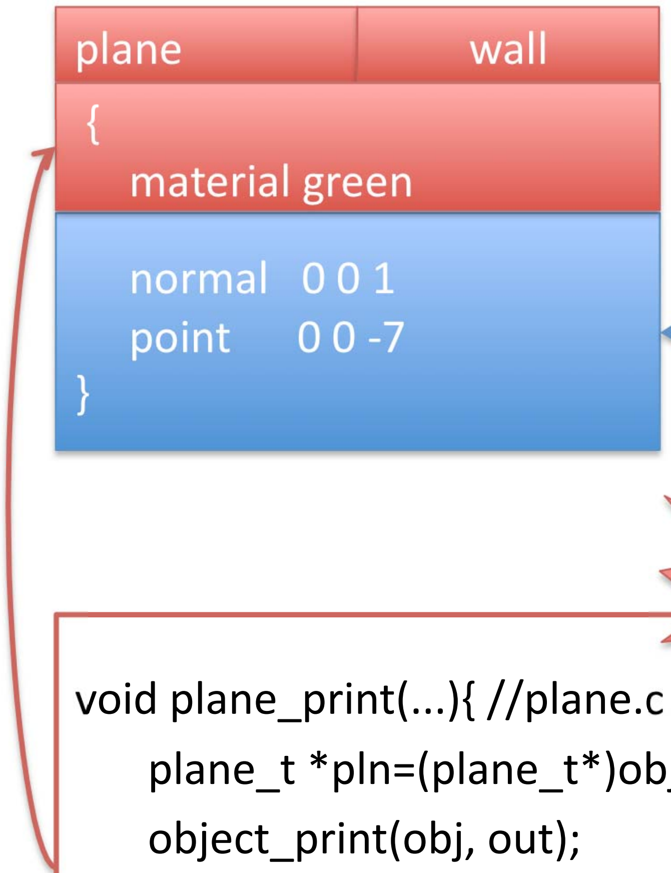
```
plane                wall
{
    material green
    normal   0 0 1
    point    0 0 -7
}
```

```
void plane_print(...){ //plane.c
    plane_t *pln=(plane_t*)obj;//masquerade
    object_print(obj, out);
    pix_print(out, “normal”, pln->normal);
    pix_print(out, “point”, pln->point);
}
```

```
void object_print(...){ //object.c
    fprintf(out, “%s %s\n”, obj->type, obj->name);
    fprintf(out, “{\n”);
    fprintf(out, “ %s %s\n”, “material”, material_getname(obj->mat));
}
```

obj->printer==plane_print!!!

New Plane output



new!

```
void plane_print(...){ //plane.c
      plane_t *pln=(plane_t*)obj;//masquerade
      object_print(obj, out);
      pix_print(out, “normal”, pln->normal);
      pix_print(out, “point”, pln->point);
}
```

old!

```
void plane_print(...){ //plane.c
      plane_t *pln=(plane_t*)obj->priv;
      object_print(obj, out);
      pix_print(out, “normal”, pln->normal);
      pix_print(out, “point”, pln->point);
}
```