C - Structures, Typecasting, Function Pointers

CS238P: Principles of operating systems - Fall’18

Aftab Hussain
(Adapted from Vikram Narayanan’s CS143A Fall ’17 slides)

University of California, Irvine
void func(char *s, char *t) {
    while (*s++ = *t++);
}
Couple of points on pointers to strings

- Depending on how you declare the strings, you may or may not be able to update them in the same way.
Couple of points on pointers to strings

- Depending on how you declare the strings, you may or may not be able to update them in the same way.
- How they are declared affects how they are stored.
Structures
Due to alignment requirements for different objects, there may be unnamed "holes" in a structure. For instance, if a char is one byte and an int is four bytes, the structure

```c
struct {
    char c;
    int i;
} ;
```

might well require eight bytes, not five. The sizeof operator returns the proper value.
• Change the type of the object for a single operation
  
  \[
  \text{var} = (\text{dest\_type}) \text{ source};
  \]
Typecasting Structure Types

- Change the type of the object for a single operation
  
  ```c
  var = (dest_type) source;
  ```

- Pass generic objects
  
  ```c
  struct cmd { int type; };
  struct execcmd {
    int type;
    char *argv[MAXARGS];
  };
  
  void runcmd(struct cmd *cmd) {
    struct execcmd *ecmd;
    ecmd = (struct execcmd*)cmd;
    //you can access stuff outside cmd struct using argv field.
  }
  ```
A more real structs example
A cool tip for initializing arrays
Designated Initializers

```c
#define CAPSLOCK (1<<3)  
#define NUMLOCK (1<<4)  
#define SCROLLLOCK (1<<5)  
static uchar togglecode[256] = {  
[0x3A] CAPSLOCK,  
[0x45] NUMLOCK,  
[0x46] SCROLLLOCK  
};  
/* equivalent to */  
togglecode[0x3A] = CAPSLOCK;  
togglecode[0x45] = NUMLOCK;  
togglecode[0x46] = SCROLLLOCK;  
```

Initialize the array elements 0x3A, 0x45, 0x46 only

---

2. sheet 77, xv6-rev9.pdf
Function Pointers
Dynamic registration with Function Pointers

- Declare a struct to hold function pointers

```c
#define NDEV 10
#define CONSOLE 1
struct devsw {
    int (*read)(struct inode*, char*, int);
    int (*write)(struct inode*, char*, int);
};
struct devsw devsw[NDEV]; /* global data structure */
```

---

3 sheet 40 xv6-rev9.pdf
4 sheet 82 xv6-rev9.pdf
Dynamic registration with Function Pointers

- Declare a struct to hold function pointers

```c
#define NDEV 10
#define CONSOLE 1
struct devsw {
    int (*read)(struct inode*, char*, int);
    int (*write)(struct inode*, char*, int);
};
struct devsw devsw[NDEV]; /* global data structure */
```

- Register function pointer

```c
int consolewrite(struct inode *ip, char *buf, int n);
int consoleread(struct inode *ip, char *dst, int n);
devsw[CONSOLE].write = consolewrite;
devsw[CONSOLE].read = consoleread;
```
Thank You