

Cache Lab Helpful Hints

Jeannie Albrecht

- Code samples taken from www.tutorialspoint.com

fgets

- **fopen** opens files for reading/writing
- **fgets** reads file from stream and stores in char array

```
#include <stdio.h>

int main () {
    FILE *fp;
    char str[60];

    /* opening file for reading */
    fp = fopen("file.txt" , "r");
    if(fp == NULL) {
        perror("Error opening file");
        return(-1);
    }

    if( fgets (str, 60, fp)!=NULL ) {
        /* writing content to stdout */
        printf(“%s\n”,str);
    }
    fclose(fp);

    return(0);
}
```

sscanf

- **sscanf** reads formatted input from string

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main () {
    int day, year;
    char weekday[20], month[20], dtm[100];

    strcpy(dtm, "Saturday March 25 1989");
    sscanf(dtm, "%s %s %d %d", weekday,
           month, &day, &year );

    printf("%s %d, %d = %s\n",
           month, day, year, weekday );

    return(0);
}
```

Pointers!



getopt

- Used to parse command line options
- `./hello -a -c 4`

```
int main (int argc, char **argv) {
    int c;
    char *cvalue = NULL;

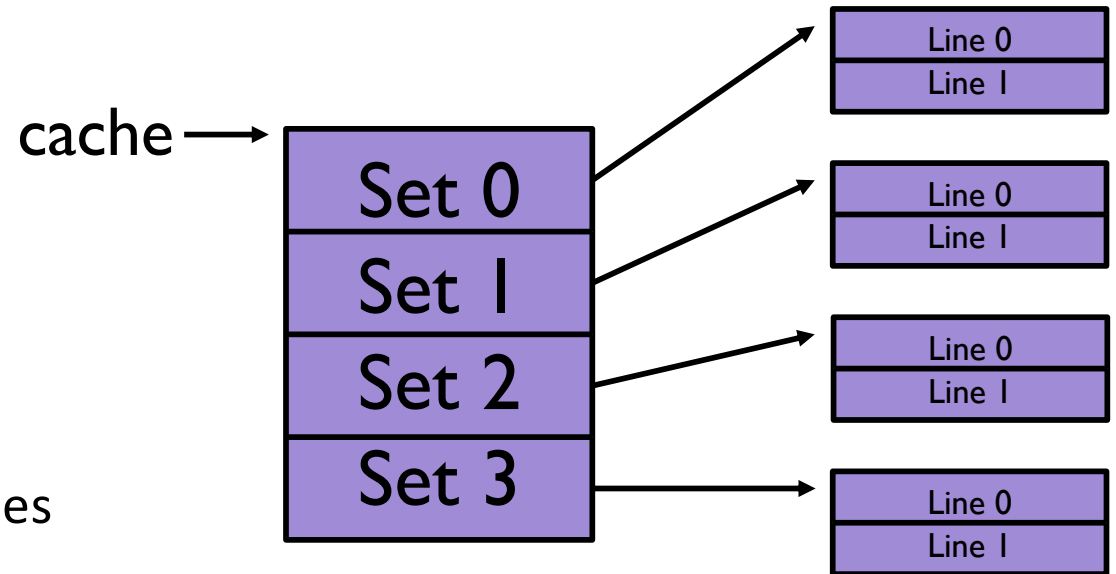
    while ((c = getopt
            (argc, argv, "ac:")) != -1)
        switch (c) {
            case 'a':
                //do something
                break;
            case 'c':
                cvalue = optarg;
                break;
            default:
                //do something
        }
}
```

Cache lab in a nutshell

- Define a struct(s) for representing your cache
- Write/review functions for:
 - main (get command line options, open trace file, read trace file, etc)
 - Initializing cache (i.e., malloc space for cache)
 - Freeing cache (i.e., any allocated memory must be freed)
 - Running simulation (update the flags of our cache accordingly)
 - Other helper functions as needed

Hints

- What is a cache?
 - An array of cache sets
- What is a cache set?
 - Each set is an array of cache lines
- What is a cache line?
 - Valid bit, tag, block
 - Note that we are only simulating a cache in Lab 5, so we don't need to represent the actual data blocks
 - Might need a little extra info to implement LRU
 - Probably want a struct to keep track of this!



Hints

■ What is a cache?

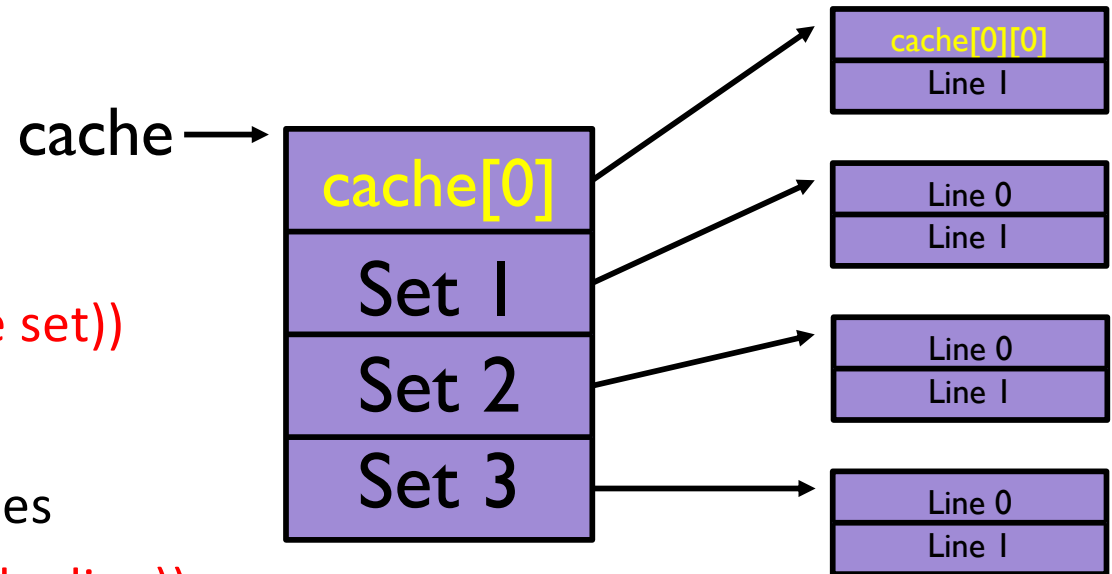
- An array of cache sets
- `cache = malloc(S * sizeof(cache set))`

■ What is a cache set?

- Each set is an array of cache lines
- `cache[i] = malloc(E * sizeof(cache line))`

■ What is a cache line?

- Valid bit, tag, block
- Note that we are only simulating a cache in Lab 5, so we don't need to represent the actual data blocks
- Might need a little extra info to implement LRU
- Probably want a struct to keep track of this!



Getting Started

- Figure out how to read from trace files
- Plan cache line struct
- Think about how you'll implement LRU algorithm