
Even More C

CSE 413, Autumn 2007
10-22-2007

1

Topics

- Command line arguments
- stdout/stdin & fgets/fputs
- #define
- hw 4
- structs

2

Command line arguments

- Supplied to main as a pair:
`int main(int argc, char** argv) or`
`int main(int argc, char* argv[])`
- `argc` = # of parameters
- `argv[]` = an array containing the parameters as strings
- `argv[0]` is the program name

3

Examples

- [See read_example.c file]
- ```
wordfind cow
argv[0] = "wordfind"
argv[1] = "cow"
```

4

---

## fgets & stdin

- Example:
- ```
fgets(dest, 10, stdin)
```
- `dest` is the address of an array of length at least 10 chars
 - `10` is \leq size of the array `dest`. At *most* 9 chars will be read and then a `'\0'` char will be added on.
 - `stdin` is the stream that input should be read from
 - stops reading when the first `'\n'` is encountered
 - `'\n'` is included in the string
 - Returns NULL on end of file or error

General:
`fgets(char* dest, int n, FILE *in)`

5

fputs & stdout

- Examples:
- ```
fputs(src, stdout)
fputs("Hello World", stdout)
```
- `src` is the address of an array of chars
  - `stdout` is the stream that input should be written to
  - The stream `stdout` is printed to the screen by default (although you can re-direct it with `>`)

General:  
`fputs(char* source, FILE *out)`

6

## Printing Examples

```
char * some_string;
...
printf("%s", some_string);
```

same as:

```
fputs(some_string, stdout);
```

7

## fputs vs. printf

```
fputs(char* source, FILE *out)
```

- Just takes a string as input
- Can specify which FILE to print to

```
printf("%s and an int = %d",
 some_string, an_int);
```

- **f** is for "format"
- can contain formatting info, other variables.
- prints to **stdout** by default

8

## #define

```
#define MAX_STUDENTS 100

int asg01_grades[MAX_STUDENTS];
int asg02_grades[MAX_STUDENTS];
int asg03_grades[MAX_STUDENTS];
```

9

## Structs

- A struct is a record. (similar to a Java object with no methods.)
  - » x.f is for field access.
  - » (\*x).f in C is like x.f in Java.
  - » x->f is an abbreviation for (\*x).f.
- There is a huge difference between passing a struct and passing a pointer to a struct.
- (see struct example code)

10