Introduction to C

EECS 2031

Summer 2014

Przemyslaw Pawluk

May 7, 2014

Introduction

C vs. Java

C syntax is similar to Java (Java has a C-like syntax)

Differences

- No garbage collection
- No classes
- No exceptions (try catch)
- No String type
- Pointers

First program in C

```
#include <stdio.h>
main() {
  printf("hello, world \n");
}
```

Note:

`#include <filename.h>` replaces the line by the actual file before compilation starts.

Communication with Environment – Standard I/O

- Every program has a standard input and a standard output.
- By default, keyboard and monitor, respectively.
- Input functions: `getchar()`, `scanf()`, `fgets()`
- Output functions: `putchar()`, `printf()`, `fputs()`

Output

Most of the time, use printf()

- Very similar to Java
- Returns the number of characters printed
- See section 7.2 in the textbook
- `int printf(char *format, arg1, arg2, ...);`

You can also use `putchar()` for a single character

Input – It's complicated ...

- Several functions for input should never be used because they are unsafe
- They are still in the standard library because a lot of code out there uses them
- Avoid using gets() as well as `scanf()` for strings
- Recommended way to read input: `getchar()` or `fgets()` and `sscanf()`

getchar

Behaviour

- Reads a single character from std. input (keyboard by default)
- Returns the next input char each time it is called
- Returns EOF when encounters it in the input (Ctrl-D or Ctrl-Z)

Documentation

```
man -S 3 getchar
```

scanf

```
int scanf(char *format, arg1, arg2, ...);
```

Behaviour

- Reads characters from the standard input, interprets them according to the specification in `format`, and stores the results through the remaining arguments (pointers)
- Stops when it exhausts its format string, or when some input fails to match the control specification
- Returns the number of successfully matched and assigned input items (e.g., to decide how many items were found)
- Returns 0 if the next input character does not match the first specification in the format string (i.e., an error)
- On the end of file, `EOF` is returned

scanf

Reading integer

```
int num;
scanf("%d", &num);
```

Reading char and float

```
char c; float f;
scanf("%c %f", &c, &f);
```

sscanf

It is similar to `scanf()` except it parses a string provided as an argument rather than standard input

Homework

- Read Chapters 1 and 7 from the textbook
- Do the lab question