Hello World

CS 2022: Introduction to C

Instructor: Hussam Abu-Libdeh

(based on slides by Saikat Guha)

Fall 2011, Lecture 2

Administrivia

Course twitter account:
`http://twitter.com/cs2022`

- check here for announcements and real-time updates

Environment

- OS: GNU/Linux
- Editor: vim
- Compiler: gcc
- Debugger: gdb

Structure of a C Program

Overall Program

```
<some pre-processor directives>

<global declarations>
<global variables>

<functions>
```

Structure of a C Program

Functions

```
<function header>
<local declarations>

<statements>
```

hello.c: Hello World

```
#include <stdio.h>

int main() {
    printf("Hello World\n");
    return 0;
}
```

Compiling and Running

- `$ gcc -o hello hello.c`
- `$ ./hello`
  `Hello World`

vars.c: Variables

```
#include <stdio.h>

int main() {
    int a, b, c;

    a = 10;
    b = 20;
    c = a * b;

    printf("a=%d b=%d c=%d\n", a, b, c);
    return 0;
}
```

```
a=10 b=20 c=200
```

func.c: Functions

```
#include <stdio.h>

int add(int a, int b) {
    printf("a=%d b=%d\n", a, b);
    return a+b;
}

int main() {
    printf("ret=%d\n", add(10, 20));
    return 0;
}
```

```
a=10 b=20
ret=30
```

cond.c: Conditionals

```
#include <stdio.h>

int main() {
    int i = 10;
    if (10 == i) {
        printf("equal to ten\n");
    } else {
        printf("not equal to ten\n");
    }
    return 0;
}
```

```
equal to ten
```

loop.c: Loops

```
#include <stdio.h>

int main() {
    int i;
    for (i = 0; i < 10; i++) {
        printf("%d ", i);
    }
    printf("done.\n");
    return 0;
}
```

```
0 1 2 3 4 5 6 7 8 9 done.
```

rec.c: Recursion

```
#include <stdio.h>

void rec(int a) {
    printf("in %d\n", a);
    if (a > 0) rec(a-1);
    printf("out %d\n", a);
}

int main() {
    rec(2);
    return 0;
}
```

```
in 2
in 1
in 0
out 0
out 1
out 2
```

cmdarg.c: Command Line Args

```
#include <stdio.h>

int main(int argc, char **argv) {
    int n, m;

    n = atoi(argv[1]);
    m = atoi(argv[2]);

    printf("Argument 1: %d\nArgument 2: %d\n", n, m);

    return 0;
}
```

```
Argument 1:  10
Argument 2:  20
```

Bonus Problem

Anagrams

Given a 4-character string as a command line input, print all anagrams that can be generated from it.

```
$ ./anagram abcd
abcd
abdc
acbd
acdb
bacd
...
```

Hint: argv[1][2] is the 2nd character in the input string.