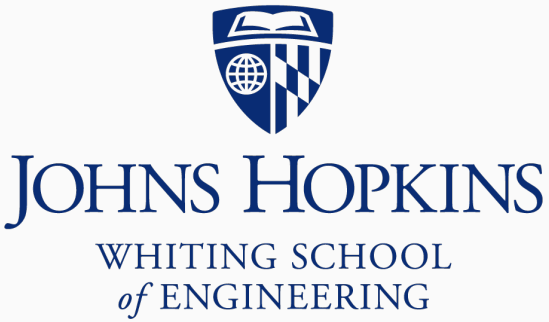
Printing messages

Ben Langmead

ben.langmead@gmail.com

www.langmead-lab.org



Source markdown available at github.com/BenLangmead/c-cpp-notes

Printing messages

`printf` prints a message to screen

```c
#include <stdio.h>

// Print "Hello world!" followed by newline and exit
int main() {
    printf("Hello world!\n");
    return 0;
}
```

```
$ gcc hello_world.c -std=c99 -pedantic -Wall -Wextra
$ ./a.out
Hello world!
```

\n at the end is a "newline" character, so subsequent output appears on the next line

Printing

`printf` can handle numbers and other types

```c
#include <stdio.h>

int main() {
    int x = 71;
    float y = 5.0 / 9.0 * (x - 32);
    printf("%.2f\n", y); // print y up to 2 decimal places
    return 0;
}
```

```
$ gcc mysterious.c -std=c99 -pedantic -Wall -Wextra
$ ./a.out
21.67
```

Printing

`printf` is a function taking one or more *arguments*

Arguments are comma-separated & specified between parentheses

```c
printf("Hello, world!\n"); // one argument

double y = 3.33;
printf("%.2lf\n", y);       // 2 arguments

printf("%d items left; price: $%.2lf\n", 10, 44.44);  // 3 args
```

Printing

First argument is a *format string*

```
"%d items left; price: $%.2f"
```

Format string contain format specifiers that start with %

- Each specifies *type* of an item to print

Printing

Specifier	Type	Example output
%d	int	-43
%u	unsigned	77
%f	float	3.333333
%c	char	P
%s	string	Hello, world!

Putting `l` just after the `%` modifies the type to be longer

- `%ld` for `long` (instead of `int`)
- `%lu` for `unsigned long` (instead of `unsigned`)
- `%lf` for `double` (instead of `float`)

Printing

```
printf("Hello, world!\n");
```

```
    Hello, world!
```

```
printf("%d\n%d\n%d\n%c\n", 3, 2, 1, '!');
```

```
    3
    2
    1
    !
```

```
printf("%f rounded is %.2f\n", 10.0/3, 10.0/3);
```

```
    3.333333 rounded is 3.33
```

Printing

Questions about types?

- What is the difference between `int` and `long`?
- How large are they?
- How are they represented in the computer?
- When and how can we convert between them?

We will answer these when we discuss numeric representations