2510 : Lecture 1

January 9, 201

Intro to C

↳ created by : Dennis Ritchie, 1970s
↳ implement Unix OS

Version : ANSI C (C89/90)

Procedural Language - no classes / objects
- by function calls

Hello World

```
/* hello.c */
#include <stdio.h>
int main (void) {
    printf ("hello, world! \n");
    return 0;
}
```

comment : Note ANSI C /* */ comments
// this is not a comment

name of function

parameter list

return type

print formatted

↳ execution starts from a function named main
↳ main has 2 valid forms :
* int main (void)
  int main (...) some args covered later

↳ the value returned by main is passed to the shell that starts the program
↳ convention : if no errors, the program should return 0.
: otherwise, return POSITIVE INT.
// check GCC return number via batch file ?

↳ Check return value
DOS : echo %errorlevel%
bash : echo $?

] print the return value of program

2510 : Lecture 1

January 9, 2017

Compiler

* Cygwin, GCC * Case sensitive

↳ in the Cygwin Shell :

```
$ gcc -ansi -W -Wall -pedantic hello.c
                    // if no errors, generates hello.exe
$ ./hello // executes the program
```

(turn on warnings; prompt; components)

Another Program :

```
#include <stdio.h>
int square (int);          /* function prototype : in this case
int main (void) {             used to declare square which
    int n = 12;               return as an int */
    printf ("%d\n", square (n));
    return 0;
}
int square (int x) {       /* definition of square */
    return x * x;
}
```

(name of header file; /* declaration of square */; int)

↳ compiling this prog. will give you an error b/c square is called before creation. // reads from top to bottom

Function Prototype - checks to see if it's called properly

↳ declaration vs. definition of a function (definition gives the actual code)

Note : A definition is also a declaration

A declaration is the existence of function

2510 : Lecture 1

↳ a function prototype allows the compiler to check that the function is called properly

↳ stdio.h contains prototypes of the function related to I/O
- in above, we need stdio.h for the compiler to check if we are calling printf correctly.

↳ printf needs to take in a STRING.
- cannot do ~~printf (n);~~

C is a dangerous language

- Even if a program runs correctly, there's no guarantee it'll run correctly the next time.
- C has a lot of undefined behavior. 2 examples

1. Uninitialized local variables
   ↳ any uninitialized local variables will be assigned random, so no two uninitialized variables will be equal

2. Buffer overflow
   ↳ there are no warning if an overflow occurs, it will simply overwrite data beside it (in memory) ex. Storing too many values in array

* testing can show that there are bugs, but it cannot show that there aren't any bugs