CSE216
Foundations of Computer Science

Instructor: Zhoulai Fu

State University of New York, Korea

C crash course

- C Language Overview.
- C Environment Setup
- C Program Structure
- C Basic Syntax
- C Data Types
- C Variables
- C Constants and Literals
- C Storage Classes
- C Operators
- Decision Making in C
- C Loops
- C Functions
- C Scope Rules
- C Arrays
- C Pointers
- C Strings
- C unions

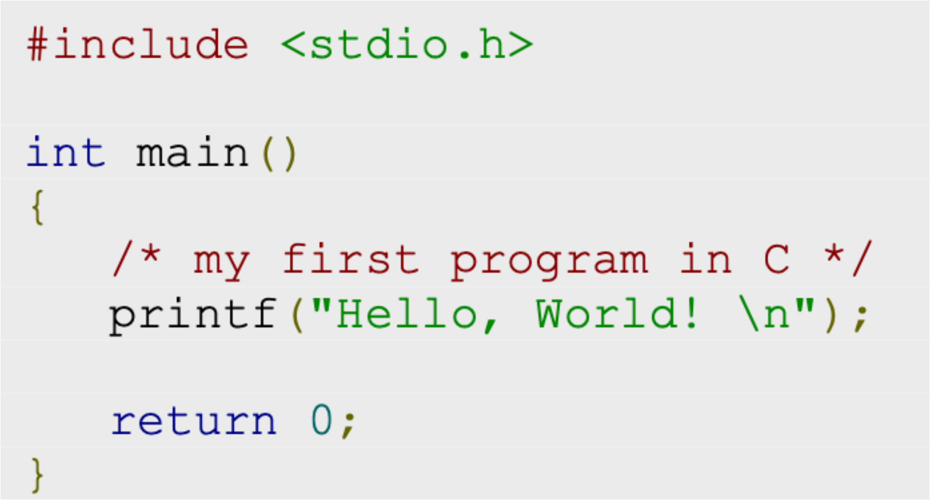
Language Overview

- Dinosaur.
- Legacy.
- Imperative.
- Close to machine

Environment Setup

- Text editor
- C compiler
- C99 (not ANSI C = C89/C90, not C11, not C17)
- gcc -std=c99 main.c
- https://www.jdoodle.com/compile-c99-online/

Program Structure



```
#include <stdio.h>

int main()
{
   /* my first program in C */
   printf("Hello, World! \n");

   return 0;
}
```

- Preprocessor Commands
- Functions
- Variables
- Statements & Expressions
- Comments

Basic Syntax

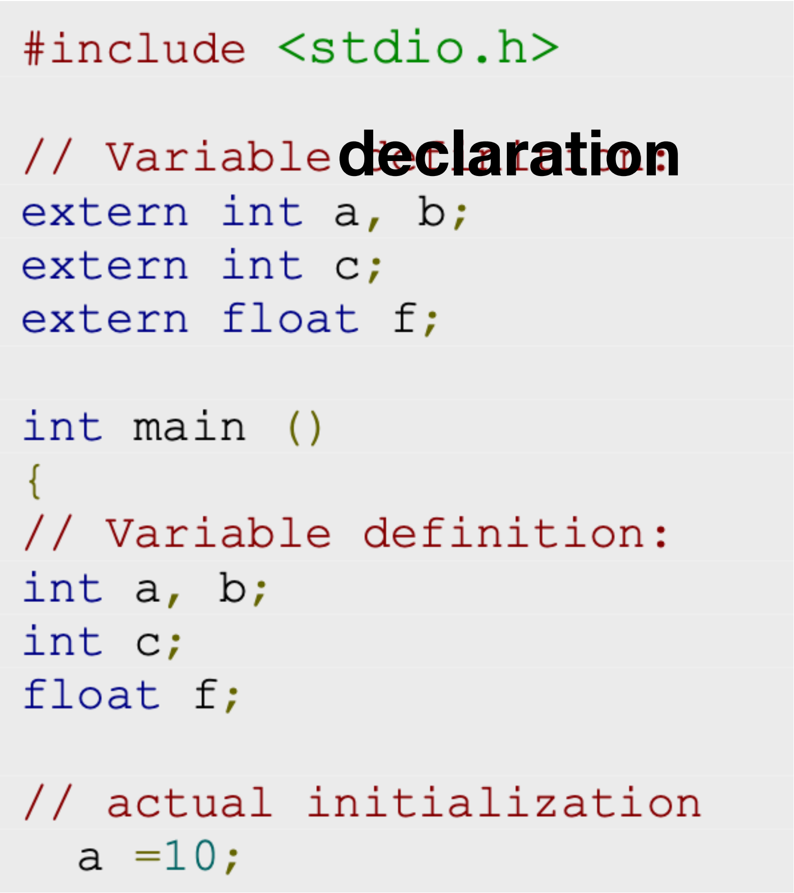
- Program := Statements
- Statement := Tokens
- Token := Keyword | Identifier | Constant | Symbol
- Semicolon ; is statement terminator
- Comments “//…” or “/*…*/” is removed during preprocessing

Data Types

- Basic types: *int, char, float, double, long*… No bool. No string
- Void type: *void exit (int), int rand(void), void *malloc(1024)*
- enum type: *enum mbti {ESTP, INFJ…}*
- *Derived types: Pointer types, Array types, Structure types, Union types and Function types.*

Variables

- Var. declaration
- Var. definition
- Var. initialization



```
#include <stdio.h>

// Variable declaration:
extern int a, b;
extern int c;
extern float f;

int main ()
{
// Variable definition:
int a, b;
int c;
float f;

// actual initialization
  a =10;
```

Question

```
#include <stdio.h>
```

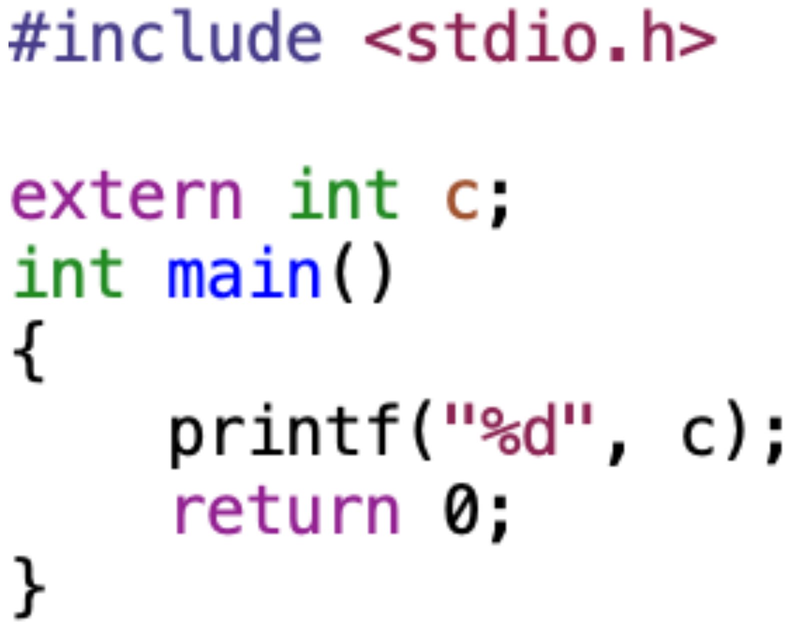
vs.

```
#include “stdio.h”
```

“” looks through current directory, while <> looks through system library folders

Question

- What will happen with “gcc main.c”



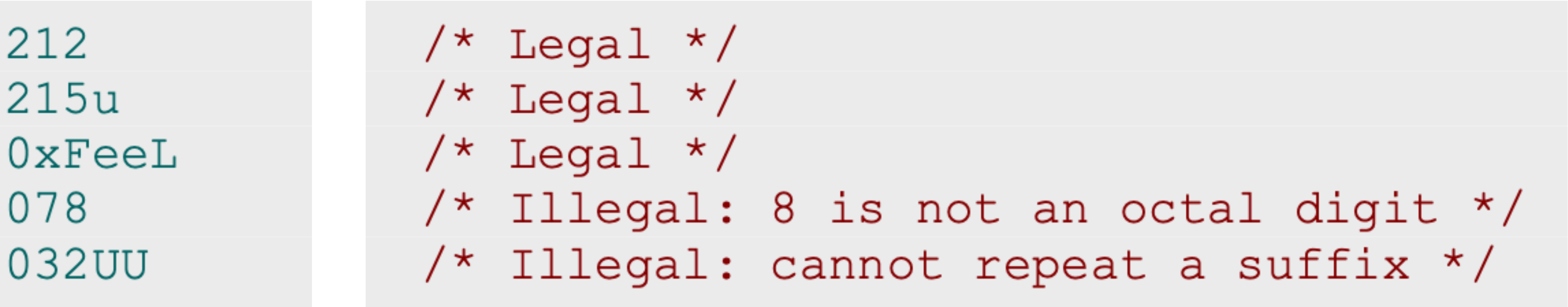
```c
#include <stdio.h>

extern int c;
int main()
{
    printf("%d", c);
    return 0;
}
```

Constants: Integer Literals

- decimal
- octal: 0213
- hexadecimal: 0x4b, 0xA0F
- Unsigned: 30u, 30U
- LongL 42L, 42l
- Suffix is case-insensitive and can be in any order: 30ul — unsigned long

Question

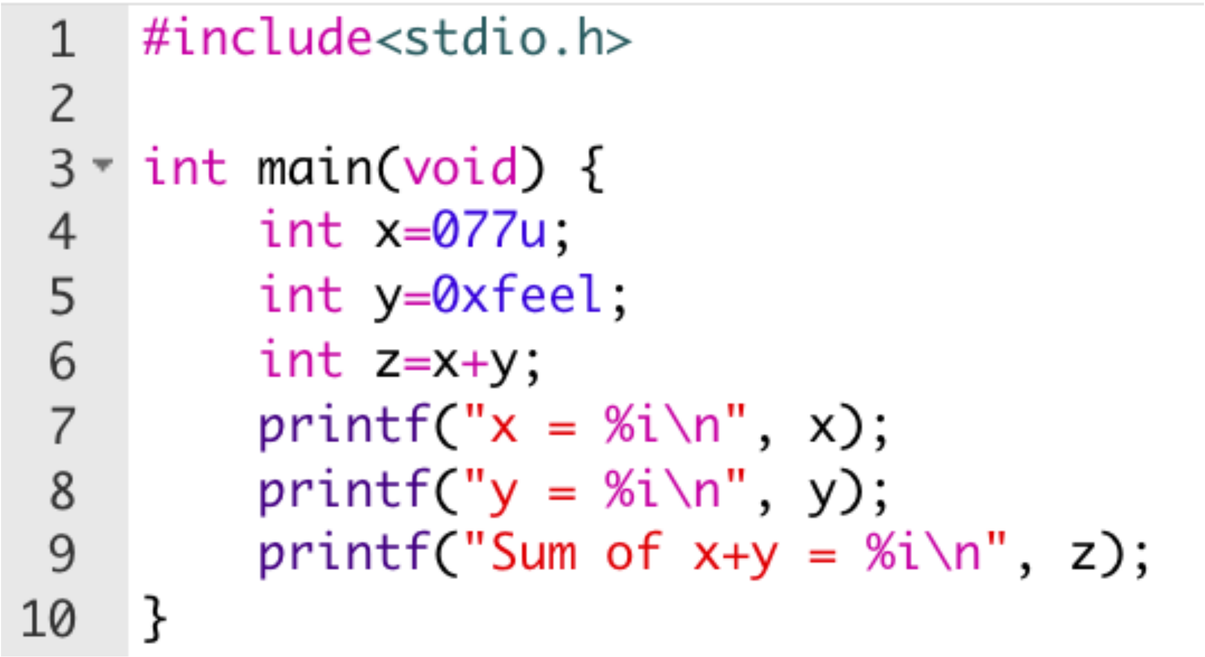
- Which one is illegal?



```
212         /* Legal */
215u        /* Legal */
0xFeeL      /* Legal */
078         /* Illegal: 8 is not an octal digit */
032UU       /* Illegal: cannot repeat a suffix */
```

Try this

- What does this program produce?

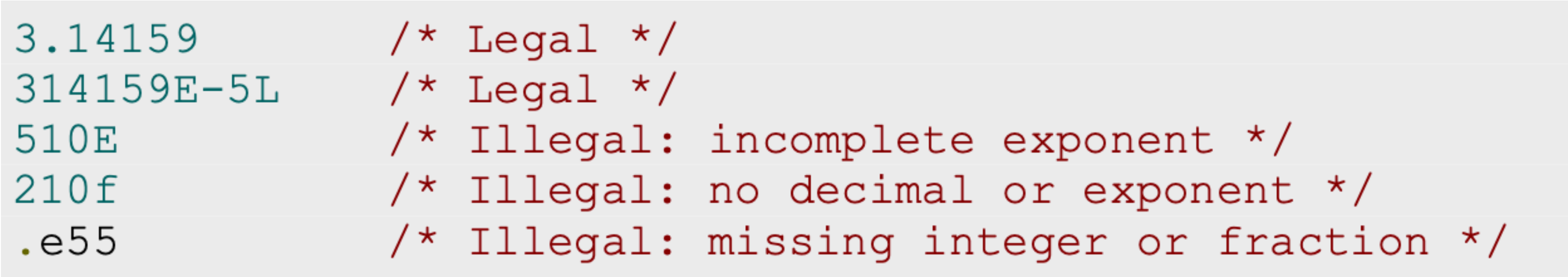


```c
#include<stdio.h>

int main(void) {
    int x=077u;
    int y=0xfeel;
    int z=x+y;
    printf("x = %i\n", x);
    printf("y = %i\n", y);
    printf("Sum of x+y = %i\n", z);
}
```

Try this: jdoodle.com/ia/IB5

Constants: Floating-point literals

- Decimal form
- Scientific notation form

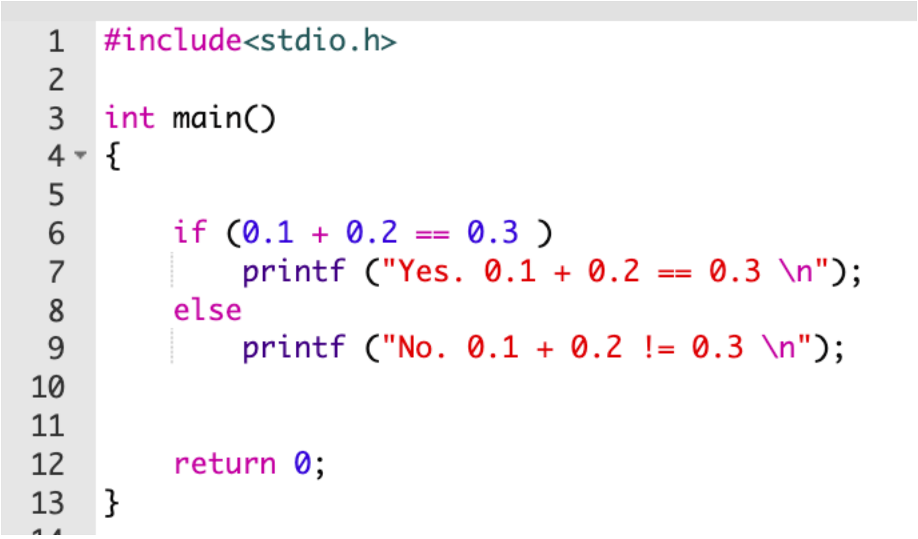


```
3.14159       /* Legal */
314159E-5L    /* Legal */
510E          /* Illegal: incomplete exponent */
210f          /* Illegal: no decimal or exponent */
.e55          /* Illegal: missing integer or fraction */
```

This kind of details is for your literature, not for the exam.

Question

- What does this program produce?

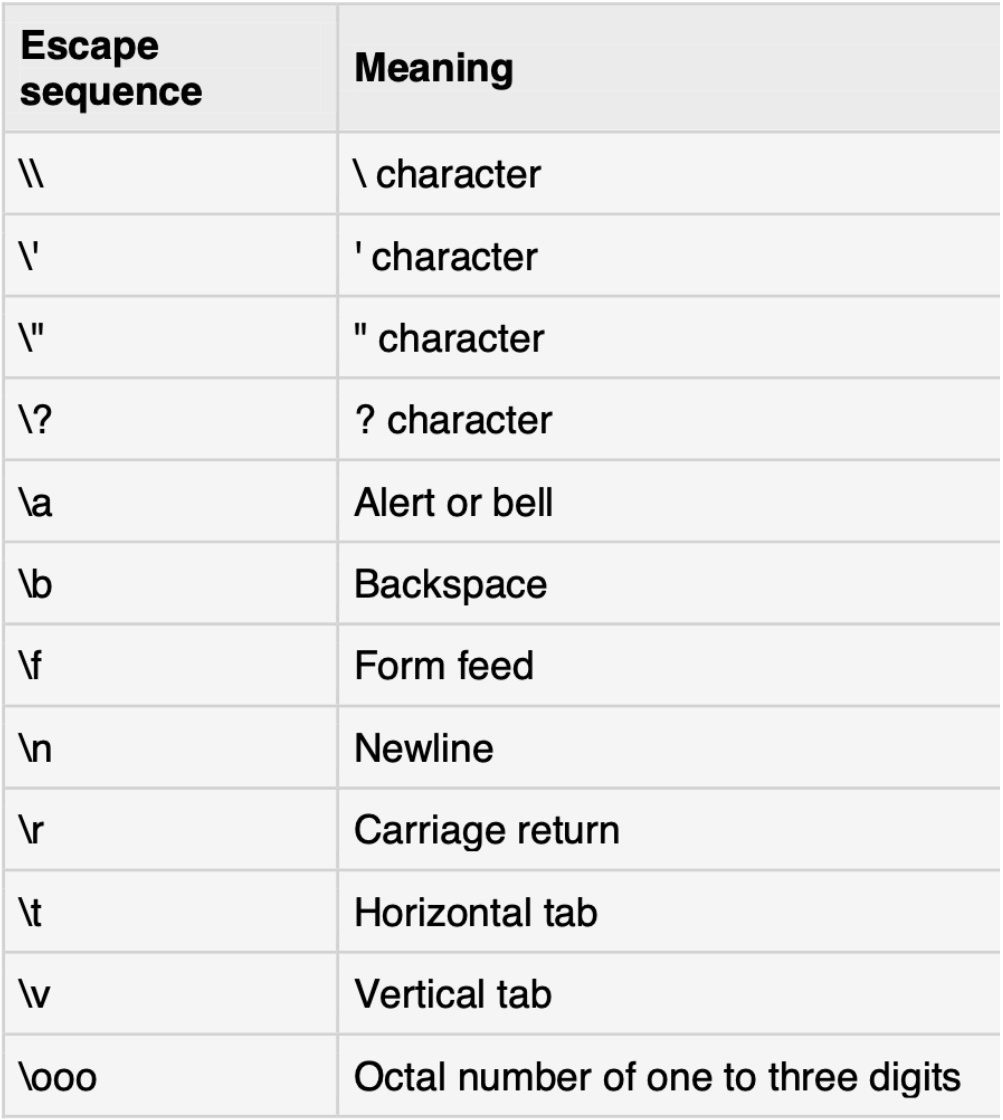


```c
#include<stdio.h>

int main()
{

    if (0.1 + 0.2 == 0.3 )
        printf ("Yes. 0.1 + 0.2 == 0.3 \n");
    else
        printf ("No. 0.1 + 0.2 != 0.3 \n");


    return 0;
}
```

jdoodle.com/ia/IB9

Constants: chars

- plain character (e.g., 'x'),
- escape sequence (e.g., ‘\t')
- universal character (e.g., '\u02C0').



Escape sequence	Meaning
\\	\ character
\'	' character
\"	" character
\?	? character
\a	Alert or bell
\b	Backspace
\f	Form feed
\n	Newline
\r	Carriage return
\t	Horizontal tab
\v	Vertical tab
\ooo	Octal number of one to three digits

Try this

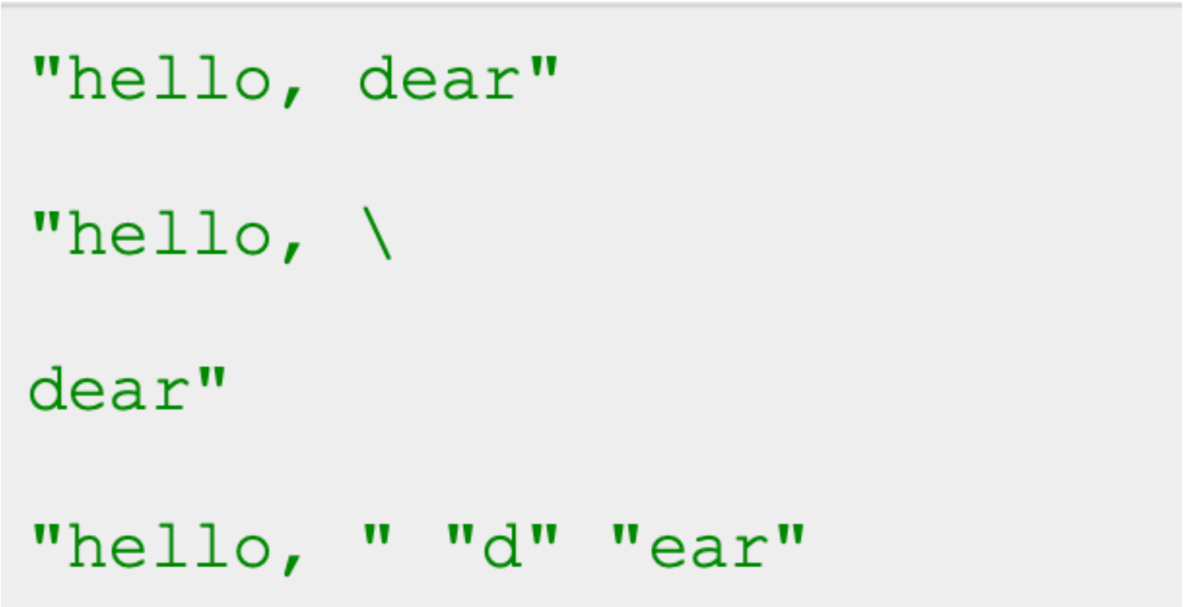
```
#include <stdio.h>

int main() {
    printf("1. Hello, World!\n");           // newline escape character
    printf("2. Hello,\tWorld!\n");          // tab escape character
    printf("3. Hello,\\World!\n");          // backslash escape character
    printf("4. Hello,\'World!\n");          // single quote escape character
    printf("5. Hello,\"World!\n");          // double quote escape character
    printf("6. Hello,\aWorld!\n");          // alert(bell) escape character
    printf("7. Hello,\bWorld!\n");          // backspace escape character
    printf("8. Hello,\fWorld!\n");          // form feed escape character
    printf("9. Hello,\rWorld!\n");          // carriage return escape character
    printf("10. Hello,\vWorld!\n");          // vertical tab escape character
    printf("11. Hello,\x48World!\n");        // hexadecimal number escape character
    printf("12. Hello,\101World!\n");        // octal number escape character
    printf("13. Hello,\u03B1World!\n");      // unicode escape character

    return 0;
}
```

Constants: strings

- strings = char sequences ending with \0

- break a long line into multiple lines = separate them using whitespaces

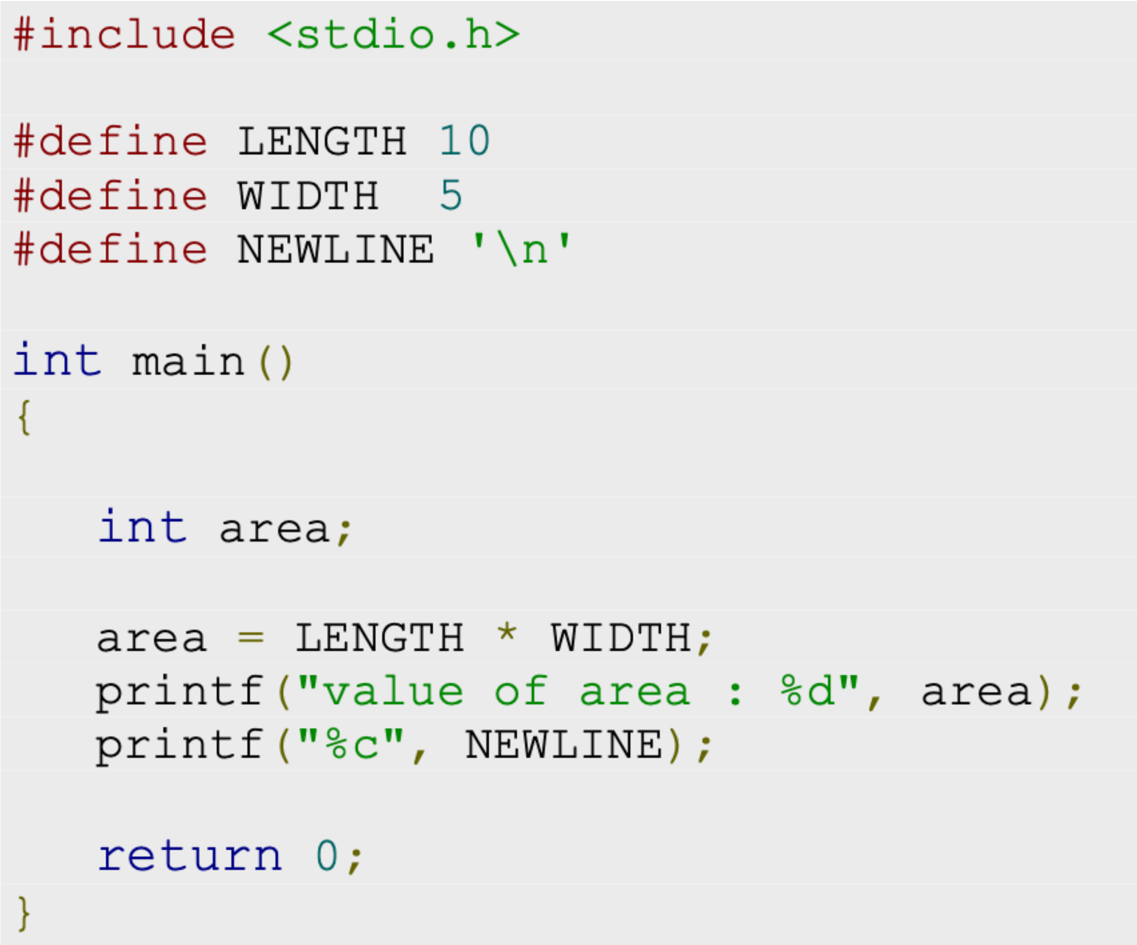
- All the three forms are identical



```
"hello, dear"

"hello, \

dear"

"hello, " "d" "ear"
```

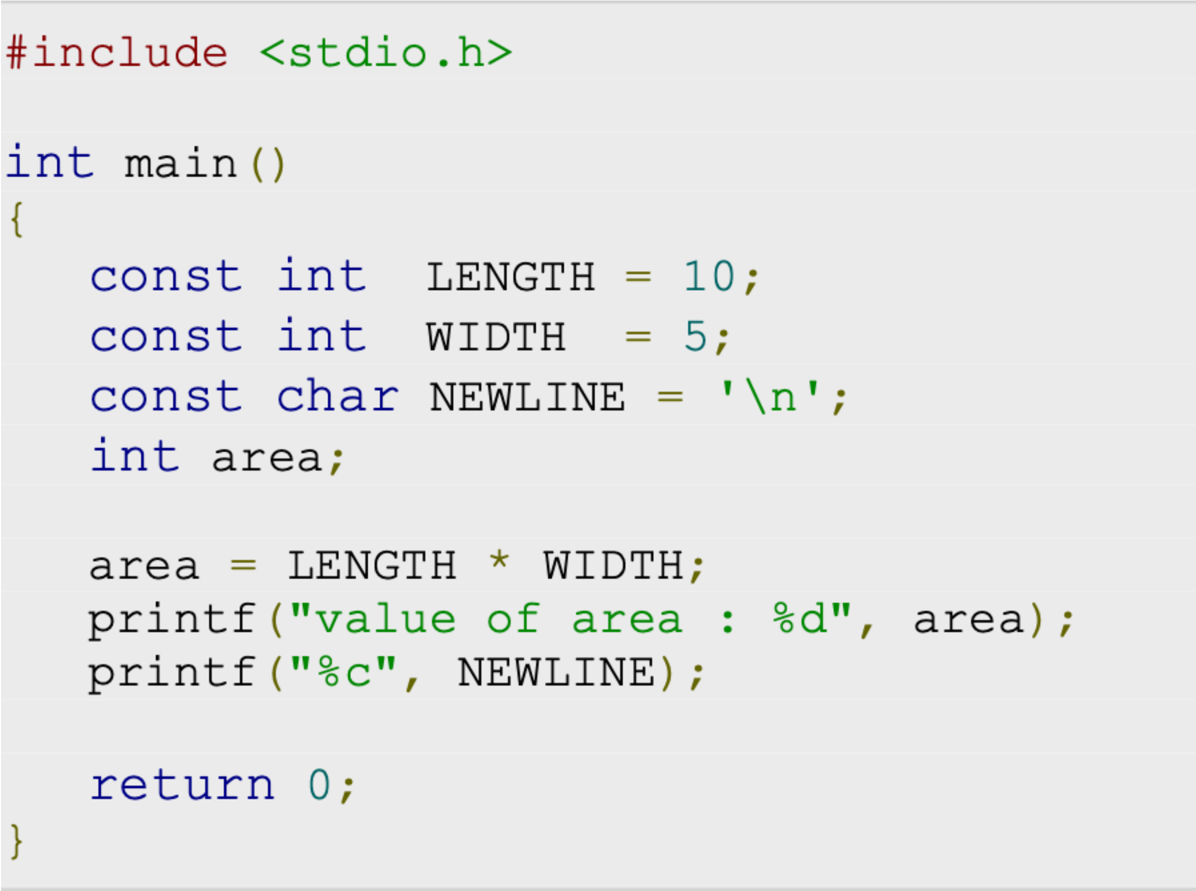
Defining Constants

Using #define preprocessor.



```c
#include <stdio.h>

#define LENGTH 10
#define WIDTH  5
#define NEWLINE '\n'

int main()
{

   int area;

   area = LENGTH * WIDTH;
   printf("value of area : %d", area);
   printf("%c", NEWLINE);

   return 0;
}
```

Using const preprocessor.



```c
#include <stdio.h>

int main()
{
   const int  LENGTH = 10;
   const int  WIDTH  = 5;
   const char NEWLINE = '\n';
   int area;

   area = LENGTH * WIDTH;
   printf("value of area : %d", area);
   printf("%c", NEWLINE);

   return 0;
}
```