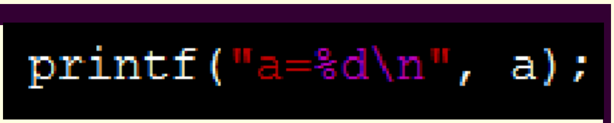
CS 241
Data Organization using C
printf and Primitive Types



```
printf("a=%d\n", a);
```

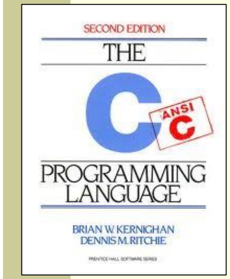
Instructor: **Joel Castellanos**

e-mail: joel@unm.edu

Web: http://cs.unm.edu/~joel/

Office: Farris Engineering Center Room 2110

Read: Kernighan & Ritchie Read



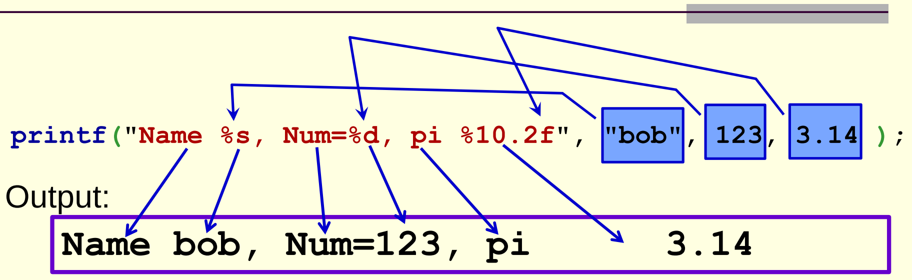
- Due Thursday, Aug 22
  - 1.1: Getting Started
  - 1.2: Variables and Arithmetic Expressions
  - 1.3: The For Statement
  - 1.4: Symbolic Constants
- Due Tuesday, Aug 27 (first iclicker quiz)
  - 1.5: Character Input and Output
- Due Thursday, Aug 29
  - 1.6: Arrays

Quiz: Basic Syntax

One of these things is not like the others. One of these things does not belong. Can you tell which thing is not like the others before the end of this quiz?

```
a) int a = 40 * 2*(1 + 3);
b) int b = (10 * 10 * 10) + 2
c) int c = (2 + 3) * (2 + 3);
d) int d = 1/2 + 1/3 + 1/4 + 1/5 + 1/6;
e) int e = 1/2 - 1/4 + 1/8 - 1/16;
```

printf function



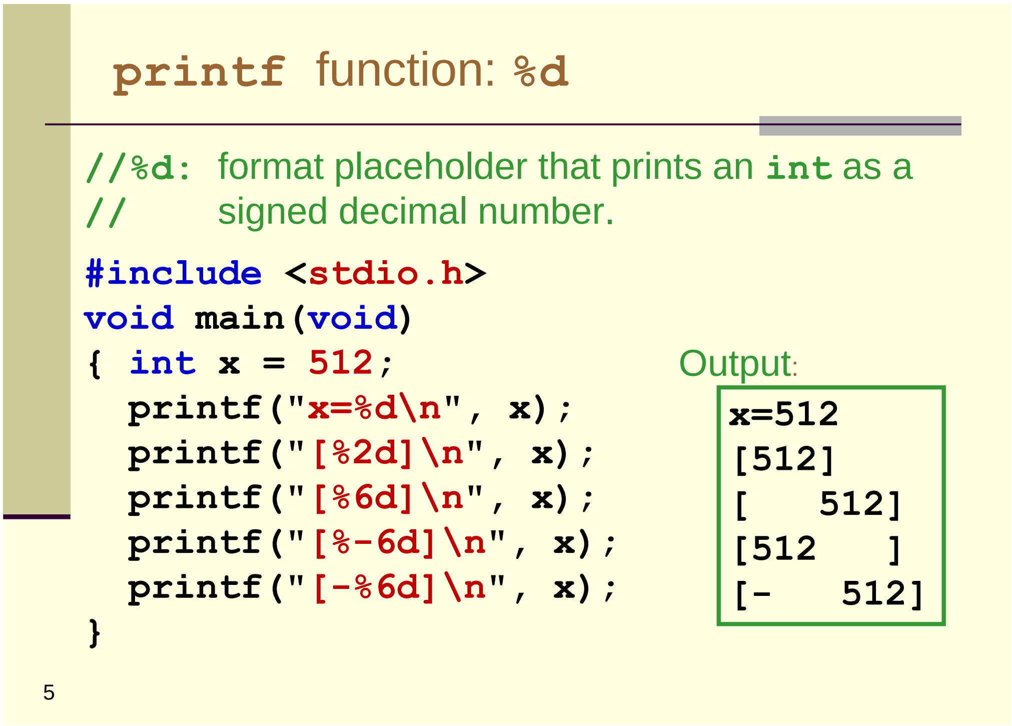
```
printf("Name %s, Num=%d, pi %10.2f", "bob", 123, 3.14 );
```

Output:

```
Name bob, Num=123, pi       3.14
```

printf format specifiers:

`%s`	string (null terminated `char` array)
`%c`	single `char`
`%d`	signed decimal `int`
`%f`	`float`
`%10.2f`	`float` with at least 10 spaces, 2 decimal places.
`%lf`	`double`



printf function: %d

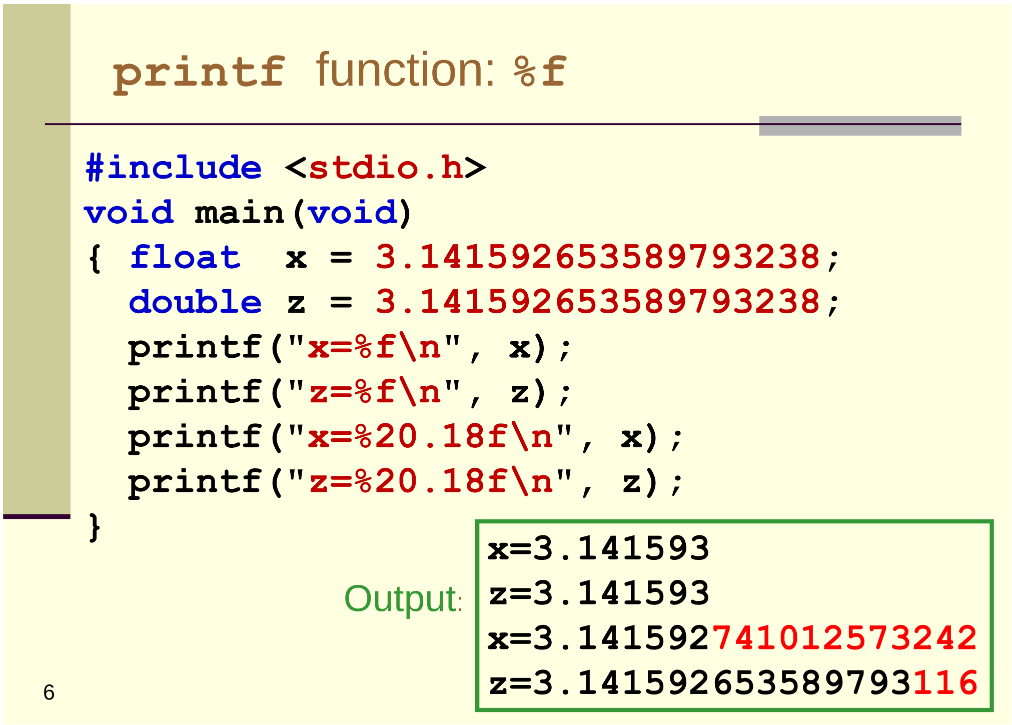
```
//%d: format placeholder that prints an int as a
//     signed decimal number.
#include <stdio.h>
void main(void)
{ int x = 512;
  printf("x=%d\n", x);
  printf("[%2d]\n", x);
  printf("[%6d]\n", x);
  printf("[%-6d]\n", x);
  printf("[-%6d]\n", x);
}
```

Output:

```
x=512
[512]
[   512]
[512   ]
[-   512]
```

5

5



printf function: %f

```
#include <stdio.h>
void main(void)
{ float  x = 3.141592653589793238;
  double z = 3.141592653589793238;
  printf("x=%f\n", x);
  printf("z=%f\n", z);
  printf("x=%20.18f\n", x);
  printf("z=%20.18f\n", z);
}
```

Output:

```
x=3.141593
z=3.141593
x=3.141592741012573242
z=3.141592653589793116
```

6

6

Significant Figures

Using **/usr/bin/gcc** on moons.unm.edu, a **float** has 7 ***significant figures***.

Significant figures are ***not*** the same as ***decimal places***.

Ms. Significant Figure

```
float x = 1.0/30000.0;
float z = 10000.0/3.0;
printf("x=%.7f\n", x);
printf("x=%.11f\n", x);
printf("x=%.15f\n", x);
printf("z=%f\n", z);
```

Output:

```
x=0.0000333
x=0.00003333333
x=0.000033333333704
z=3333.333252
```

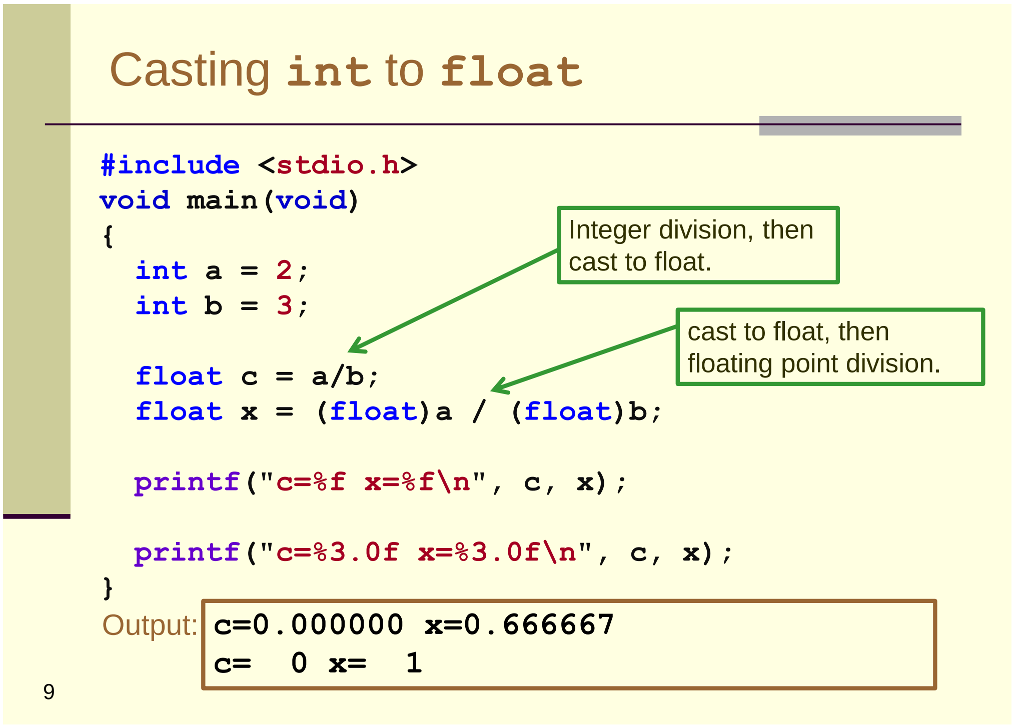
printf function: **%e**

%e: Format placeholder that prints a **float** or **double** in ***Scientific Notation***.

```
#include <stdio.h>
void main(void)
{
  float x = 1.0/30000.0;
  float y = x/10000;
  float z = 10000.0/3.0;
  printf("x=%e\n", x);
  printf("y=%e\n", y);
  printf("z=%e\n", z);
  printf("x=%.2e\n", x);
}
```

Output:

```
x=3.333333e-05
y=3.333333e-09
z=3.333333e+03
x=3.33e-05
```



Casting `int` to `float`

```c
#include <stdio.h>
void main(void)
{
  int a = 2;
  int b = 3;

  float c = a/b;
  float x = (float)a / (float)b;

  printf("c=%f x=%f\n", c, x);

  printf("c=%3.0f x=%3.0f\n", c, x);
}
```

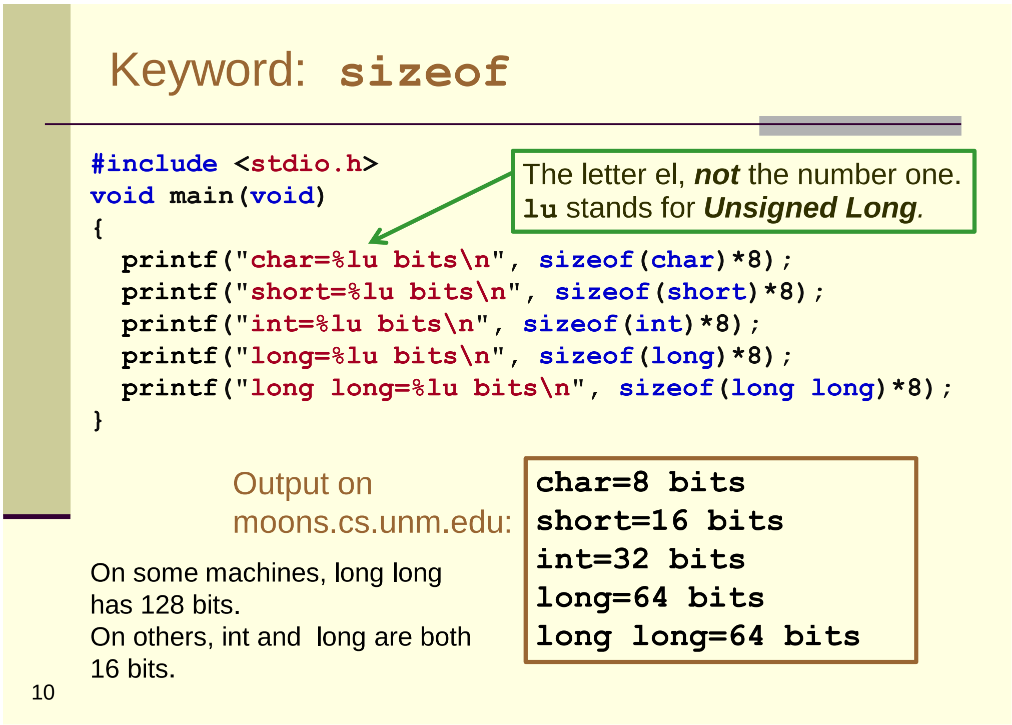
Integer division, then cast to float.

cast to float, then floating point division.

Output:

```
c=0.000000 x=0.666667
c=  0 x=  1
```

9



Keyword: `sizeof`

```c
#include <stdio.h>
void main(void)
{
  printf("char=%lu bits\n", sizeof(char)*8);
  printf("short=%lu bits\n", sizeof(short)*8);
  printf("int=%lu bits\n", sizeof(int)*8);
  printf("long=%lu bits\n", sizeof(long)*8);
  printf("long long=%lu bits\n", sizeof(long long)*8);
}
```

The letter el, ***not*** the number one. `lu` stands for ***Unsigned Long***.

Output on moons.cs.unm.edu:

```
char=8 bits
short=16 bits
int=32 bits
long=64 bits
long long=64 bits
```

On some machines, long long has 128 bits.
On others, int and long are both 16 bits.

10

printf function: %c

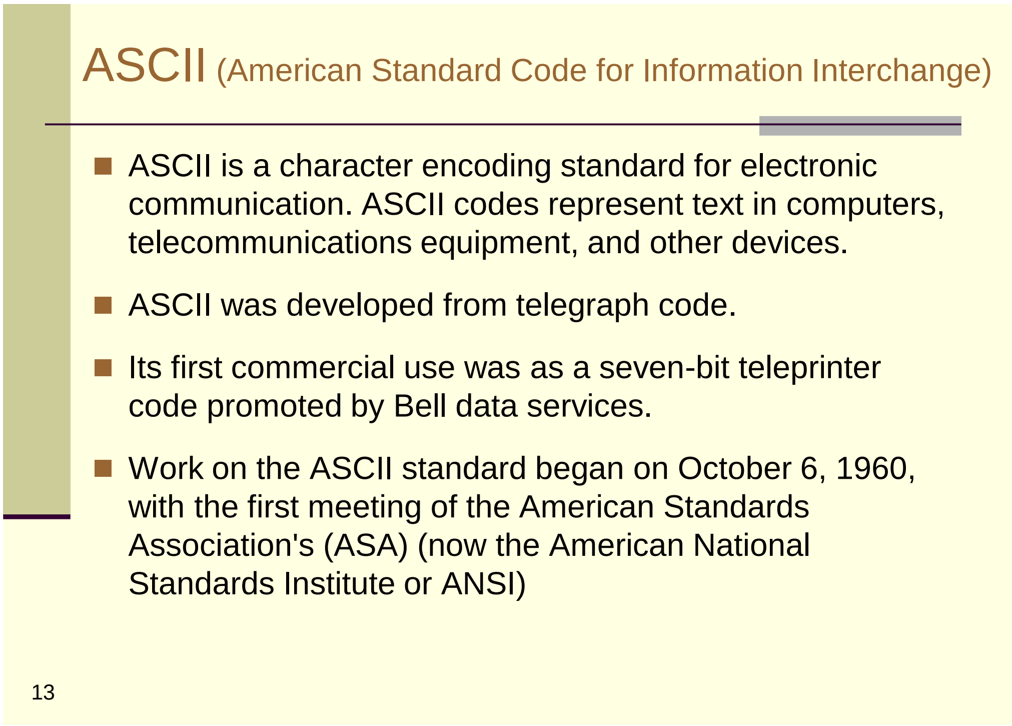
```
#include <stdio.h>
void main(void)
{
  char x = 'A';
  char y = 'B';
  printf("The ASCII code for %c is %d\n", x, x);
  printf("The ASCII code for %c is %d\n", y, y);
  y++;
  printf("The ASCII code for %c is %d\n", y, y);
}
```

Output:

```
The ASCII code for A is 65
The ASCII code for B is 66
The ASCII code for C is 67
```

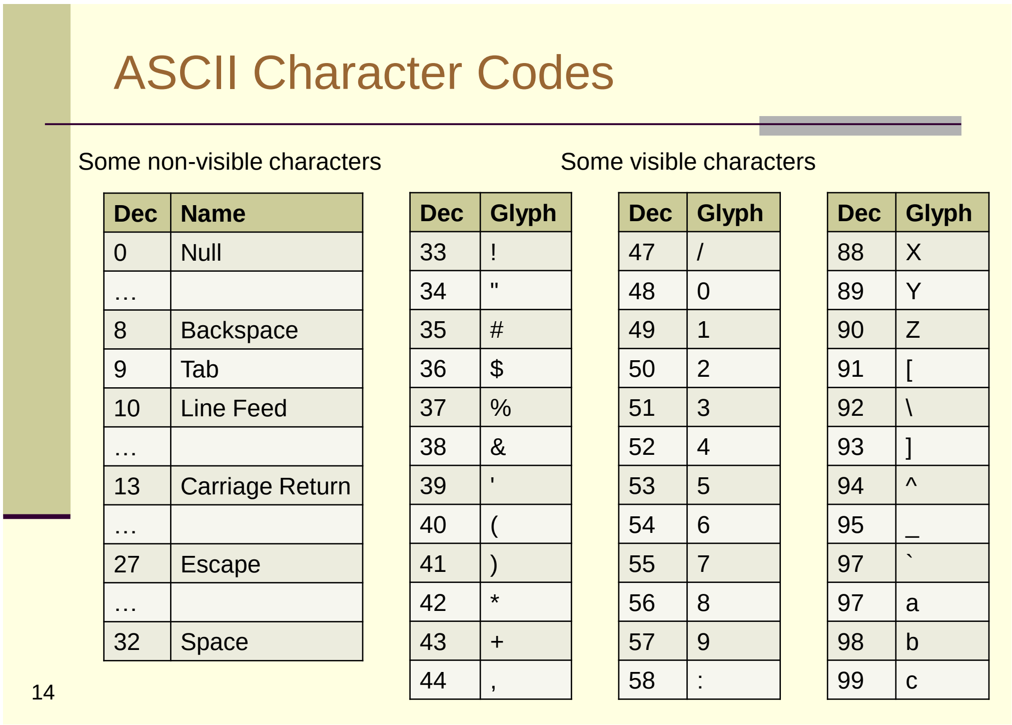
ASCII (American Standard Code for Information Interchange)

- ASCII is a character encoding standard for electronic communication. ASCII codes represent text in computers, telecommunications equipment, and other devices.
- ASCII was developed from telegraph code.
- Its first commercial use was as a seven-bit teleprinter code promoted by Bell data services.
- Work on the ASCII standard began on October 6, 1960, with the first meeting of the American Standards Association's (ASA) (now the American National Standards Institute or ANSI)



ASCII (American Standard Code for Information Interchange)

- ASCII is a character encoding standard for electronic communication. ASCII codes represent text in computers, telecommunications equipment, and other devices.
- ASCII was developed from telegraph code.
- Its first commercial use was as a seven-bit teleprinter code promoted by Bell data services.
- Work on the ASCII standard began on October 6, 1960, with the first meeting of the American Standards Association's (ASA) (now the American National Standards Institute or ANSI)



ASCII Character Codes

Some non-visible characters

Dec	Name
0	Null
…	
8	Backspace
9	Tab
10	Line Feed
…	
13	Carriage Return
…	
27	Escape
…	
32	Space

Some visible characters

Dec	Glyph
33	!
34	"
35	#
36	$
37	%
38	&
39	'
40	(
41	)
42	*
43	+
44	,

Dec	Glyph
47	/
48	0
49	1
50	2
51	3
52	4
53	5
54	6
55	7
56	8
57	9
58	:

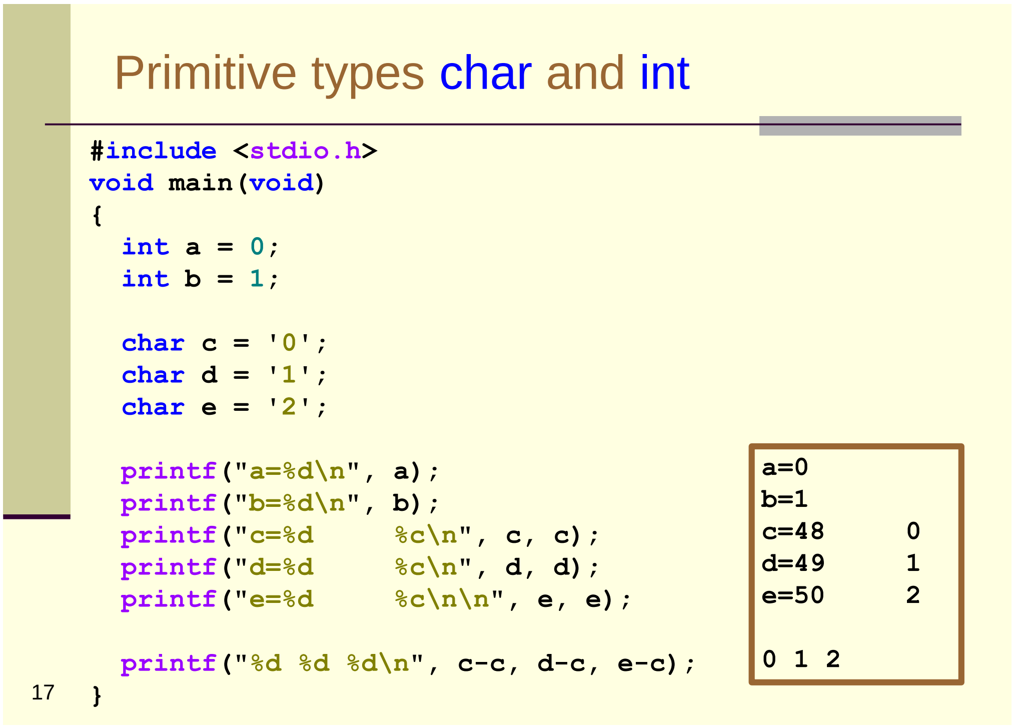
Dec	Glyph
88	X
89	Y
90	Z
91	[
92	\
93	]
94	^
95	_
97	`
97	a
98	b
99	c

ASCII Character Codes (Printable)

32		46	.	60	<	74	J	88	X	102	f	116	t
33	!	47	/	61	=	75	K	89	Y	103	g	117	u
34	"	48	0	62	>	76	L	90	Z	104	h	118	v
35	#	49	1	63	?	77	M	91	[	105	i	119	w
36	$	50	2	64	@	78	N	92	\	106	j	120	x
37	%	51	3	65	A	79	O	93	]	107	k	121	y
38	&	52	4	66	B	80	P	94	^	108	l	122	z
39	'	53	5	67	C	81	Q	95	_	109	m	123	{
40	(	54	6	68	D	82	R	96	`	110	n	124	|
41	)	54	7	69	E	83	S	97	a	111	o	125	}
42	*	56	8	70	F	84	T	98	b	112	p	126	~
43	+	57	9	71	G	85	U	99	c	113	q		
44	,	58	:	72	H	86	V	100	d	114	r		
45	-	59	;	73	I	87	W	101	e	115	s		

ASCII vs Unicode Character Codes

- ASCII codes 0-127 are composed of 7 bits and are standard since 1960s in all countries.
- The 8th bit of ASCII codes (128-255) are extended ASCII. These are less standard but generally include accented vowels (á, é, í, ó, ú, ü, ...), ñ, Ñ, mathematical symbols (π, Σ, ⊥, ∠, ±, …) commercial symbols (£, ©, ®, …) and others.
- Unicode: There are standards for 8 bit (256 symbols), 16 bit (65,536 symbols) and 32 bit (over 4 billion symbols).
- For backward compatibility, the first 7 bits of all the Unicode standards are the same as the 7 bit ASCII codes.



Primitive types char and int

```
#include <stdio.h>
void main(void)
{
  int a = 0;
  int b = 1;

  char c = '0';
  char d = '1';
  char e = '2';

  printf("a=%d\n", a);
  printf("b=%d\n", b);
  printf("c=%d     %c\n", c, c);
  printf("d=%d     %c\n", d, d);
  printf("e=%d     %c\n\n", e, e);

  printf("%d %d %d\n", c-c, d-c, e-c);
}
```

```
a=0
b=1
c=48     0
d=49     1
e=50     2

0 1 2
```