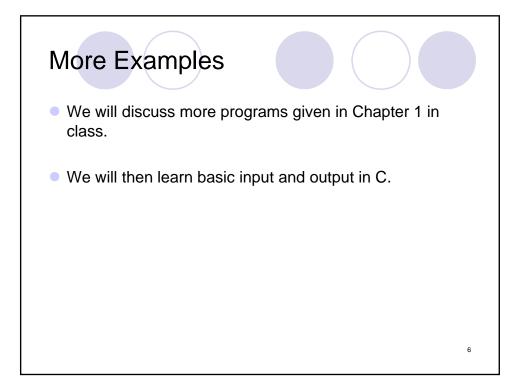
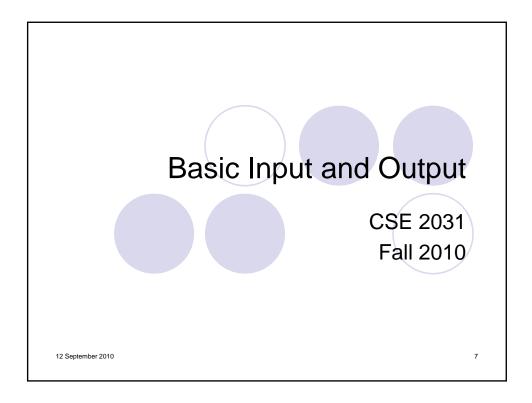
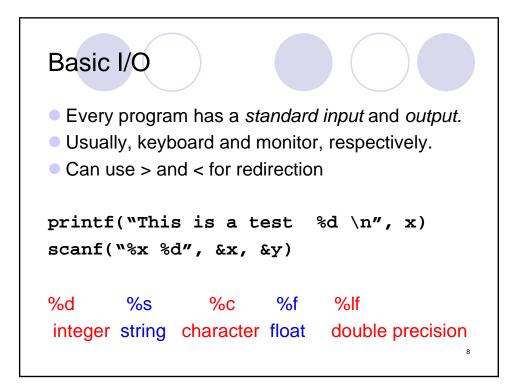
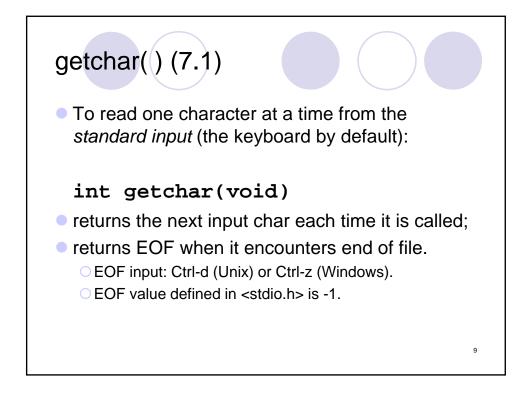


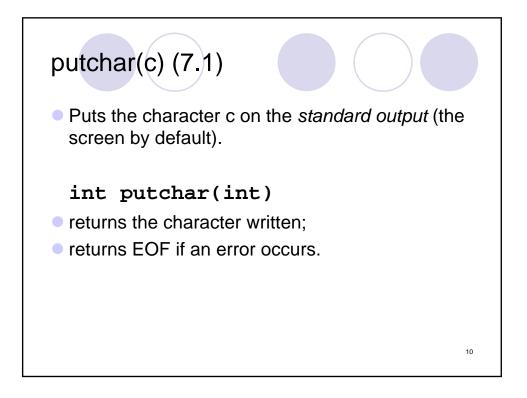
\n	New line
\t	Tab
\"	Double quote
//	The \ character
/0	The null character
\'	Single quote



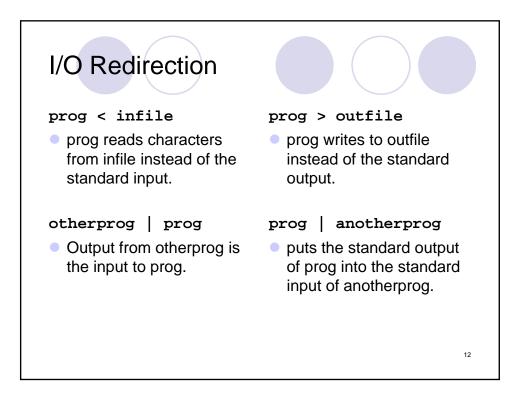


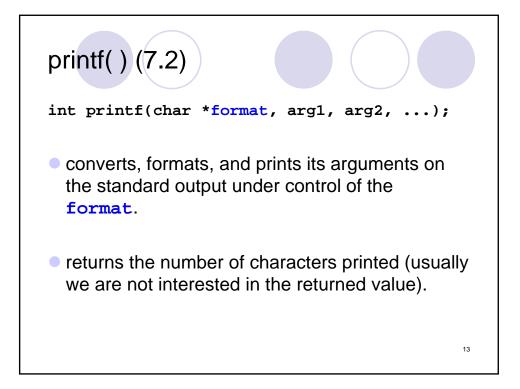






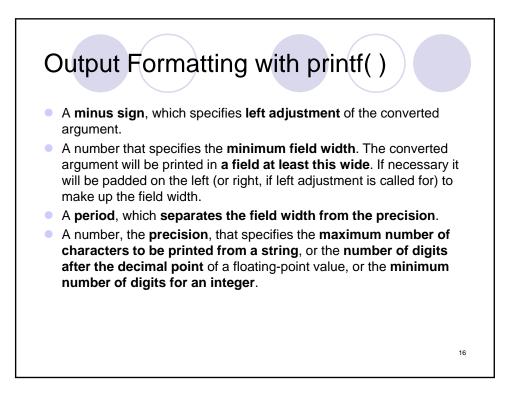
```
Example
#include <stdio.h>
#include <ctype.h>
main() /* convert input to lower case*/
{
    int c
    while ((c = getchar()) != EOF)
    putchar(tolower(c));
    return 0;
}
```

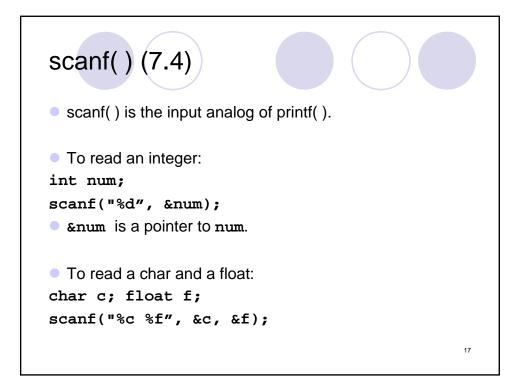




printf() Example	s
<pre>printf(":%s:",</pre>	<pre>whello, world");</pre>
<pre>printf(":%10s:",</pre>	"hello, world");
printf(":%.10s:",	"hello, world");
<pre>printf(":%-10s:",</pre>	<pre>whello, world");</pre>
printf(":%.15s:",	<pre>whello, world");</pre>
printf(":%-15s:",	<pre>whello, world");</pre>
	<pre>, "hello, world");</pre>
printf(":%-15.10s:	", "hello, world");
:%5:	:hello, world:
:%10s:	:hello, world:
:%.10s:	:hello, wor:
:%-10s:	:hello, world:
:%.15s:	:hello, world:
	:hello, world :
	: hello, wor:
:%-15.10s:	:hello, wor :

orintf Conversions			
Character	Argument type; Printed As		
d,i	int; decimal number		
0	int; unsigned octal number (without a leading zero)		
x,X	int; unsigned hexadecimal number (without a leading 0x or 0X), using abcdef or ABCDEF for 10,,15.		
u	int; unsigned decimal number		
с	int; single character		
s	char *; print characters from the string until a '\0' or the number of characters given by the precision.		
f	double; [-] m.dddddd, where the number of d's is given by the precision (default 6).		
e,E	double; $[-]m.dddddde+/-xx$ or $[-]m.ddddddE+/-xx$, where the number of <i>d</i> 's is given by the precision (default 6).		
g,G	double; use %e or %E if the exponent is less than -4 or greater than or equal to the precision; otherwise use %f. Trailing zeros and a trailing decimal point are not printed.		
р	void *; pointer (implementation-dependent representation).		
0/0	no argument is converted; print a %		





scanf Conversions		
Character	Input Data; Argument type	
d	decimal integer; int *	
i	integer; int *. The integer may be in octal (leading 0) or hexadecimal (leading 0x or 0X).	
0	octal integer (with or without leading zero); int *	
u	unsigned decimal integer; unsigned int *	
x	hexadecimal integer (with or without leading 0x or 0X); int *	
с	characters; char *. The next input characters (default 1) are placed at the indicated spot. The normal skip-over white space is suppressed; to read the next non-white space character, use %1s	
	character string (not quoted); char *, pointing to an array of characters long enough for the string and a terminating '\0' that will be added.	
sta I	floating-point number with optional sign, optional decimal point and optional exponent; float $*$	
5	literal %; no assignment is made.	

