Single Character Input: getchar()
#include <stdio.h>
main()
{
    int ch;
    ch = getchar();
}

Single Character Output: putchar()
#include <stdio.h>
main()
{
    int ch;
    putchar('?'); /* display a prompt */
    ch = getchar(); /* read a keyboard char */
    putchar(ch); /* echo to screen */
}

Printing One Line: printf()
printf("Type any keyboard character.\n") ;
printf("Your character is %c \n",ch) ;

Special C Character Constants

<table>
<thead>
<tr>
<th>constant</th>
<th>ASCII name</th>
<th>hex value</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>LF</td>
<td>0x0a</td>
<td>newline</td>
</tr>
<tr>
<td>l</td>
<td>LT</td>
<td>0x09</td>
<td>tab</td>
</tr>
<tr>
<td>w</td>
<td>VT</td>
<td>0x0b</td>
<td>vertical tab</td>
</tr>
<tr>
<td>b</td>
<td>BS</td>
<td>0x08</td>
<td>backspace</td>
</tr>
<tr>
<td>f</td>
<td>CR</td>
<td>0x0d</td>
<td>return</td>
</tr>
<tr>
<td>t</td>
<td>FF</td>
<td>0x0c</td>
<td>form feed</td>
</tr>
<tr>
<td>\</td>
<td>0x5c</td>
<td>\</td>
<td>slash</td>
</tr>
<tr>
<td>'</td>
<td>0x27</td>
<td>'</td>
<td>apostrophe</td>
</tr>
<tr>
<td>0</td>
<td>NUL</td>
<td>0x00</td>
<td>null</td>
</tr>
</tbody>
</table>

Conversion Strings and Specifiers

printf("conversion string", variable list)

<table>
<thead>
<tr>
<th>Specifier</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>%c</td>
<td>print a character</td>
</tr>
<tr>
<td>%d</td>
<td>print an integer</td>
</tr>
<tr>
<td>%i</td>
<td>print an integer (same as %d)</td>
</tr>
<tr>
<td>%e</td>
<td>print float value in exponential form</td>
</tr>
<tr>
<td>%f</td>
<td>print float value</td>
</tr>
<tr>
<td>%g</td>
<td>print using %e or %f, whichever is smaller</td>
</tr>
<tr>
<td>%o</td>
<td>print octal value (unsigned)</td>
</tr>
<tr>
<td>%s</td>
<td>print a string</td>
</tr>
<tr>
<td>%x</td>
<td>print hexadecimal integer (unsigned) using lowercase a-f</td>
</tr>
<tr>
<td>%X</td>
<td>print hexadecimal integer (unsigned) using uppercase A-F</td>
</tr>
<tr>
<td>%p</td>
<td>print a pointer value</td>
</tr>
</tbody>
</table>

Formatted Input: scanf()
scanf("conversion string", variable list)

<table>
<thead>
<tr>
<th>Specifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>%c</td>
<td>reads a character</td>
</tr>
<tr>
<td>%d</td>
<td>reads an integer</td>
</tr>
<tr>
<td>%e</td>
<td>reads a float value</td>
</tr>
<tr>
<td>%f</td>
<td>reads a float value</td>
</tr>
<tr>
<td>%h</td>
<td>reads a short integer</td>
</tr>
<tr>
<td>%o</td>
<td>reads an octal value (unsigned)</td>
</tr>
<tr>
<td>%s</td>
<td>reads a string</td>
</tr>
<tr>
<td>%x</td>
<td>reads a hexadecimal integer (unsigned)</td>
</tr>
<tr>
<td>%u</td>
<td>reads an unsigned integer</td>
</tr>
<tr>
<td>%i</td>
<td>reads a decimal, octal or hexadecimal integer</td>
</tr>
<tr>
<td>%p</td>
<td>reads the hexadecimal representation of a pointer</td>
</tr>
</tbody>
</table>

Example:

n = 256
x = 129.546
ch = %

- %3d  prints as 256
- %3d  prints as 256
- %10d prints as 256
- %f   prints as 256.000
- %f   prints as 256 with 5 trailing blanks
- %a   prints as 400
- %x   prints as 100
- %x   prints as 256
- %c   prints as R
- %d   prints as 82
- %e   prints as 1.29546E2
- %7.2f prints as 129.55
- %3f  prints as 129.546

char name[20] ;
scanf("%20s",name) ;
scanf("%d",&n) ;
scanf("%7.3f,%8.4f,%s",&average,&count,name) ;
Reading a Line of Data: gets()

This is a sample string input:

```
# include <stdio.h>    /* L3.C example 1 */)    
main( ) { /* pause to input a symbol from keyboard */ 
    int r ;         /* result */ 
    printf( "To proceed - press a key + <CR>\n" ) ;  
    r = getchar ( ) ;    /* entering a symbol */ 
    printf( "The program continues!\n" ) ;  
    printf( "The received value from getchar is: \%2d \n", r ) ;  
} 
```

```
#include <stdio.h>    /* example 2 */)    
main( ) { 
    int ch ; 
    puts( "This is a sample string." ) ;  
    puts( name ) ;  
    puts( &name ) ;  
} 
```

Printing a String: puts()

```
char name[] ; 
gets(name) ; 
puts( "This is a sample string." ) ;  
puts( name ) ;  
puts( &name ) ;  
```

The puts() function always executes a newline character when it is used. A special version, cputs() does not write a newline when executed.

result = getch( ) ;  
ungetch(result) ;

```
#include <stdio.h>    /*  example 6 */    
main( ) { 
    int n ; 
    char first_nm[20], last_nm[20] ;  
    float cost ; 
    /*1*/printf("Please type in your first and last names.");  
    /*2*/scanf("%s %s", first_nm, last_nm) ;  
    /*3*/putchar(\n) ;  
    /*4*/printf("How many items are there?");  
    /*5*/scanf("%d", &n) ;  
    /*6*/putchar(\n) ;  
    /*7*/printf("What is the unit cost?");  
    /*8*/scanf("%f", &cost) ;  
    /*9*/printf("\n\nOk %s %s, 
There are %d items at \$%4.2f\n",  
        first_nm, last_nm, n, cost) ;  
} 
```
#include <stdio.h>     /* example 7 */
#include <string.h>
main()
{
    char msg[30],inbuff[81] ;      /* assume 80 col screen */
    strcpy(msg,"Hello Turbo C !") ;
    puts(msg) ;
    printf("What is the input ? \n") ;
    /* read a line: This is a sample string input <CR> "*/
    gets(inbuff) ;
    printf("You typed %s \n",inbuff) ;   /* echo */
}