CURSES QUICK REFERENCE GUIDE

Install the Curses Library

Curses is not a standard library in C. To use Curses in programming, install the library with this command:

```
sudo apt install libncurses-dev
```

Compiling the program

To compile the source code (in a file named pex1.c), use the following:

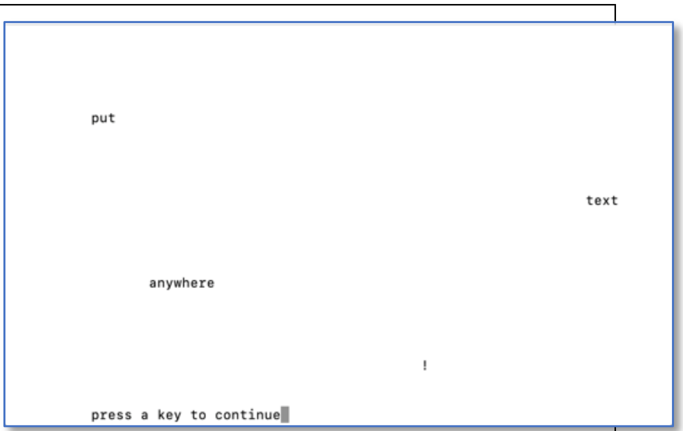
```
gcc pex1.c -Wall -Werror -o pex1 -lcurses
```

Notice the use of the -**`lcurses`** option. It is needed to link the Curses Library with the application code. The first part “**`-l`**” is means link a library, the rest says what library.

Example 1

The following program provides a more substantial program using the Curses library. You can probably deduce from this example the default screen has 23 rows and 79 columns. (If the window has been resized you may see something different.)

```
#include <ncurses.h>

int main(void ) {

   initscr(); // Start curses mode

   move(5, 10);
   printw("put");

   move(10, 70);
   printw("text");

   move(15, 17);
   printw("anywhere");

   move(20, 50);
   printw("!");

   // prevent application from ending until keystroke
   move(23,10);                      // move cursor near bottom of the window
   printw("press a key to continue"); // display a message
   getch();                          // wait for keystroke

   endwin();  // End curses mode

   return 0;
}
```

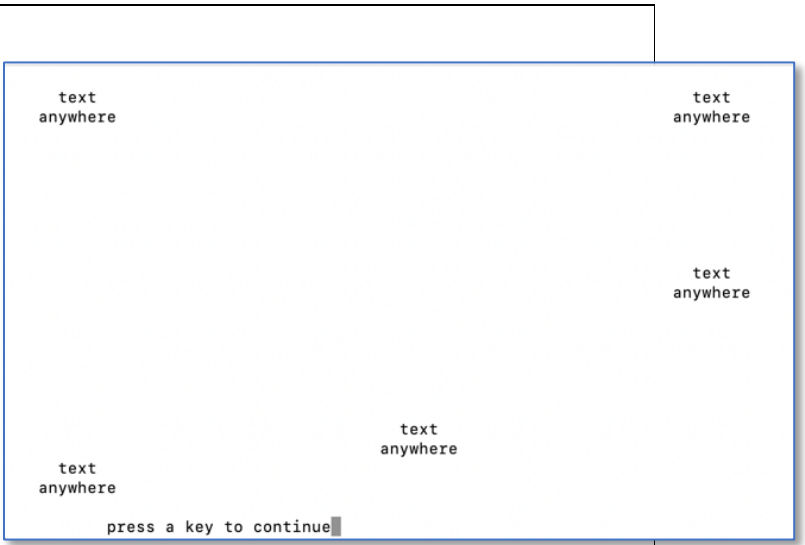


```
      put


                                                                      text


                 anywhere


                                                  !

      press a key to continue
```

Example 2: Using User-Defined Functions

The program below uses a user-defined function to display a simple message. The function is called five times to display the message in five different places.

```
#include <ncurses.h>

void myMessage(int row, int col) {

   move(row, col);
   printw("text");

   move(row+1, col-2);
   printw("anywhere");
}


int main(void ) {

   initscr(); // Start curses mode

   myMessage(10, 70);
   myMessage( 1, 70);
   myMessage( 1,  5);
   myMessage(18, 40);
   myMessage(20,  5);


   // prevent application from ending until keystroke
   move(23,10);                      // move cursor near bottom of the window
   printw("press a key to continue"); // display a message
   getch();                          // wait for keystroke

   endwin();  // End curses mode

   return 0;
}
```



```
  text                                                      text
anywhere                                                  anywhere






                                                            text
                                                          anywhere





                              text
                            anywhere
  text
anywhere

       press a key to continue
```

Color in Curses

Using color in Curses is a four-step process: 1) turn on color mode, 2) define a color pair, 3) turn the color-pair "on" 4) print.

Color in Curses Step 1: turn color mode on

… couldn't be easier: call **`start_color();`**

Color in Curses Step 2: define color-pairs

The Curses Library defines a standard set of colors using named constants: COLOR_BLACK, COLOR_RED, COLOR_GREEN, COLOR_YELLOW, COLOR_BLUE, COLOR_MAGENTA, COLOR_CYAN, and COLOR_WHITE. From these colors, pairs of colors can be defined. The pair is a foreground (text) color and a background color.

Defining a color pair involves two steps:

1) #define a name for the pair, for example **`#define RED_BLUE 1`**

Each #define for a color pair needs a unique integer value. In the example the number 1 is used. A second #define, for a second pair of colors, could use the number 2.

2) call the **`init_pair()`** function to associate the name with two colors. For example,
`init_pair(RED_BLUE, COLOR_RED, COLOR_BLUE);`

Color in Curses Step 3: turn the color-pair on

Continuing with the example, use the function call **`attron(COLOR_PAIR(RED_BLUE));`** to begin to use the RED_BLUE color-pair ("attribute on").

Color in Curses Step 4: print

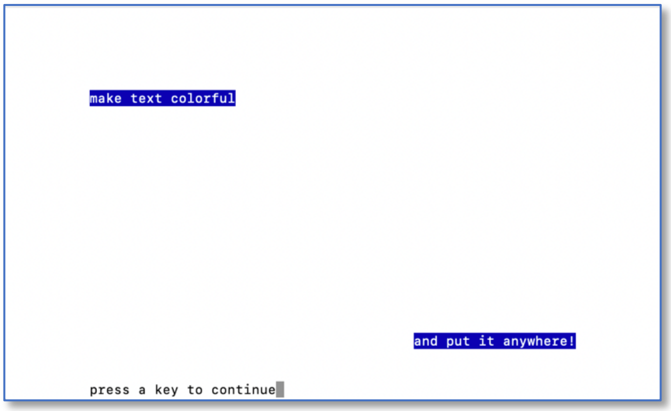
Just like before, but now the text uses the color-pair most recently turned on.

Here is a complete example.

```
10. #include <ncurses.h>
11.
12. #define WHITE_BLUE 1
13. #define BLACK_WHITE 2
14.
15. int main(void ) {
16.
17.    initscr(); // Start curses mode
18.
19.    start_color(); // enable color
20.
21.    init_pair(WHITE_BLUE, COLOR_WHITE, COLOR_BLUE);   // define color pair WHITE_BLUE
22.    init_pair(BLACK_WHITE, COLOR_BLACK, COLOR_WHITE); // define color pair BLACK_WHITE
23.    wbkgd(stdscr, COLOR_PAIR(BLACK_WHITE));           // set standard color for window
24.
25.    attron(COLOR_PAIR(WHITE_BLUE));
26.    move(5, 10);
27.    printw("make text colorful");
28.
29.    move(20, 50);
30.    printw("and put it anywhere!");
31.    attroff(COLOR_PAIR(WHITE_BLUE));
32.
33.
34.    // prevent applicaiton from ending until keystoke
35.    move(23,10);                       // move cursor near bottom of the window
36.    printw("press a key to continue"); // display a message
37.    getch();                           // wait for keystroke
38.
39.    endwin();  // End curses mode
40.
41.    return 0;
42. }
```

Source Code Listing 3: curses3.c – the output is shown below

Notice the program creates a BLACK_WHITE color-pair (line 22) that is set as the default colors for the window (line 23). These colors are used when the WHITE_BLUE color-pair is turned "off" (line 31).



```
make text colorful

and put it anywhere!

press a key to continue
```

Curses Functions

`attroff(<attribute>)` turn an attribute off

An attribute is a color, bold, or other feature of text. (See attron() for more details.)

`attron(<attribute>)` turn an attribute on

turn an attribute on: an attribute is a color, bold, or other feature of text. Some attribute value are A_BOLD, A_DIM, A_BLINK, or COLOR_PAIR(<n>). After an attribute is turned on, it can be turned off. For color, see also: **init_pair()** and **start_color()**

`endwin()` End window / end curses mode

Call this function last in your program.

`getch()` wait for keystroke

This function simply waits for a key to be pressed.

`init_pair(<index>, <color>, <color>)` define a color-pair

The function **start_color()** must be called before init_pair() can be used. A color pair must be initialized before it is an attribute that can be turned on; this is a two step process. (a) Use a **#define** to give a unique name to a unique integer for a color pair. For example:

```
#define RED_BLUE
```

The use init_pair() to associate the name with color values. For example:

```
init_pair(RED_BLUE, COLOR_RED, COLOR_BLUE);
```

The name RED_BLUE can be used to change the color with attron(COLOR_PAIR(RED_BLUE));
These are the colors defined in the Curses Library:

```
COLOR_BLACK        COLOR_GREEN        COLOR_BLUE         COLOR_CYAN
COLOR_RED          COLOR_YELLOW       COLOR_MAGENTA      COLOR_WHITE
```

`initscr()` Initialize the screen / start curses mode

Call this function first in your program.

`move(<row>, <col>)` move the cursor to row, col

move the cursor to row, col

`printw(<string>)` print a string

print a string

`start_color()` enable color

This function must be called before a color pair can be initialized. See: **init_pair()**

`wbkgd(stdscr, COLOR_PAIR(<index>));` set standard color for window

Sets the window background color – or the default color for the screen.

Much more information on Curses is found at:
https://tldp.org/HOWTO/NCURSES-Programming-HOWTO/index.html