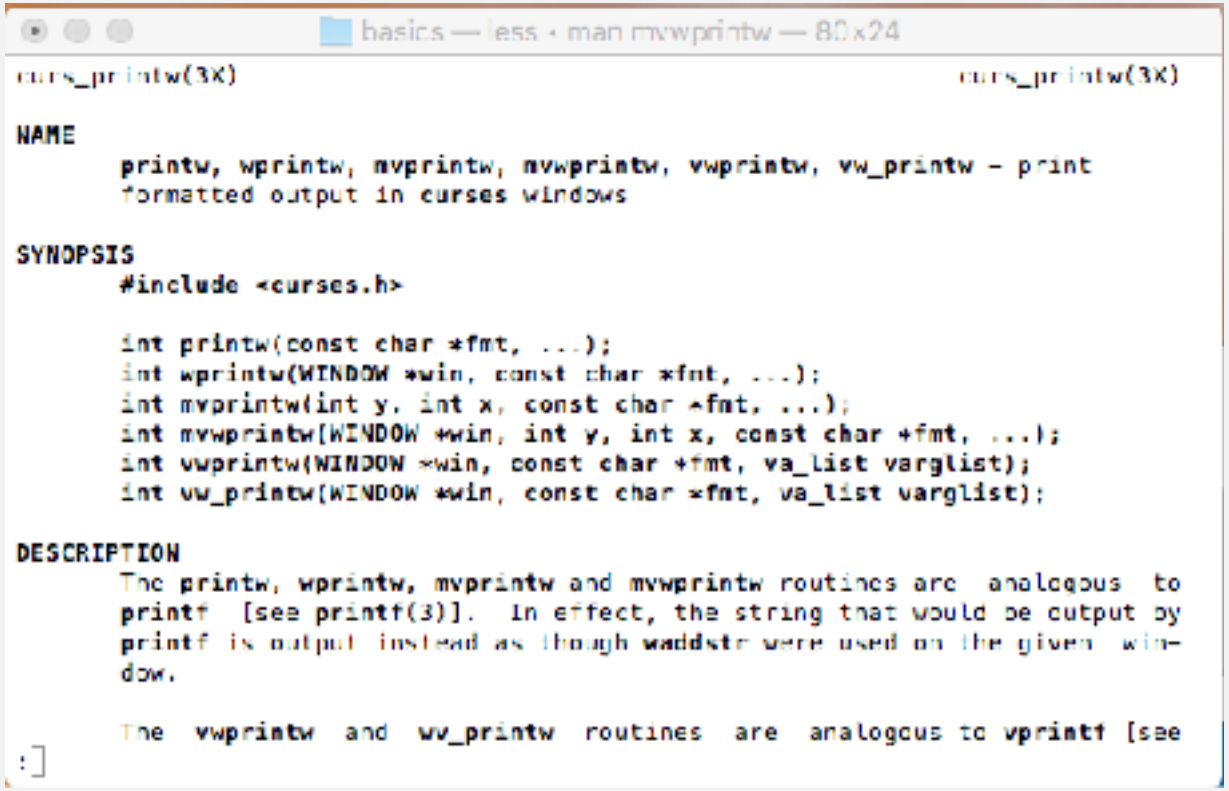
Tetris : Object-Oriented Programming

ncurses

- text-based user-interface library
  - GPU ncurses http://www.gnu.org/software/ncurses/
  - NCURSES Programming HOWTO http://tldp.org/HOWTO/NCURSES-Programming-HOWTO/ : with helpful examples
  - 한국어 번역 https://wiki.kldp.org/wiki.php/NCURSES-Programming-HOWTO
  - $man ncurses on linux command prompt shows manual page
  - $man mvvline etc.



```
basics — less ◂ man mvwprintw — 80×24
curs_printw(3X)                                        curs_printw(3X)

NAME
       printw, wprintw, mvprintw, mvwprintw, vwprintw, vw_printw - print
       formatted output in curses windows

SYNOPSIS
       #include <curses.h>

       int printw(const char *fmt, ...);
       int wprintw(WINDOW *win, const char *fmt, ...);
       int mvprintw(int y, int x, const char *fmt, ...);
       int mvwprintw(WINDOW *win, int y, int x, const char *fmt, ...);
       int vwprintw(WINDOW *win, const char *fmt, va_list varglist);
       int vw_printw(WINDOW *win, const char *fmt, va_list varglist);

DESCRIPTION
       The printw, wprintw, mvprintw and mvwprintw routines are  analogous  to
       printf  [see printf(3)].  In effect, the string that would be output by
       printf is output  instead as  though waddstr were used on the given  win-
       dow.

       The  vwprintw  and  vw_printw  routines  are  analogous to vprintf [see
:]
```

Hello_world.cpp for ncurses

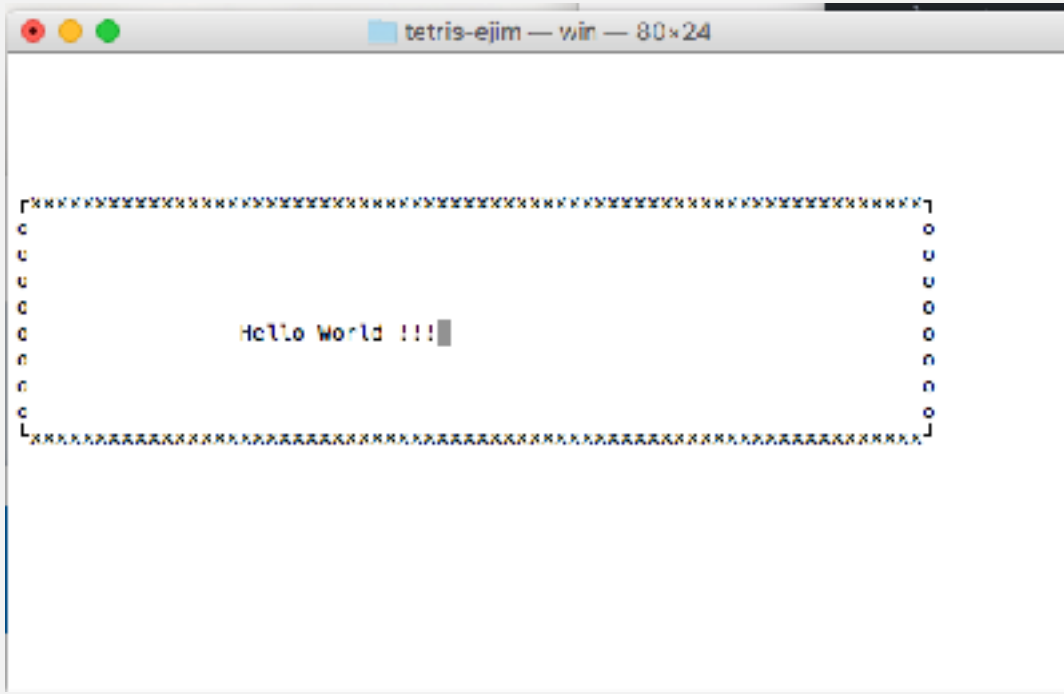
- compile command
  - g++ -o hello hello.cpp -lncurses

```
#include <ncurses.h>
int main()
{
	initscr();		/* Start curses mode		*/
	printw("Hello World !!!");	/* Print Hello World		*/
	refresh();		/* Print it on to the real screen */
	getch();		/* Wait for user input */
	endwin();		/* End curses mode		*/

	return 0;
}
```

creating windows using ncurses library

- window is a non-overlapping part of text screen.
- when curses is initialized, a default window named stdscr which represents the whole size of window in which your terminal (xterm) is running.

```
#include <ncurses.h>
int main()
{
        initscr();
        refresh(); // 꼭 필요함!!
        int height = 10, width = 70, x = 0, y = 5;
        WINDOW *w = newwin(height, width, y, x);
        box(w, 'o', 'x');
        mvwprintw(w, height/2, width/4, "Hello World !!!");
        wrefresh(w);

        getch();                    /* Wait for user input */
        delwin(w);
        endwin();                           /* End curses mode

        return 0;
}
```



Simple tetris.cpp 1-1

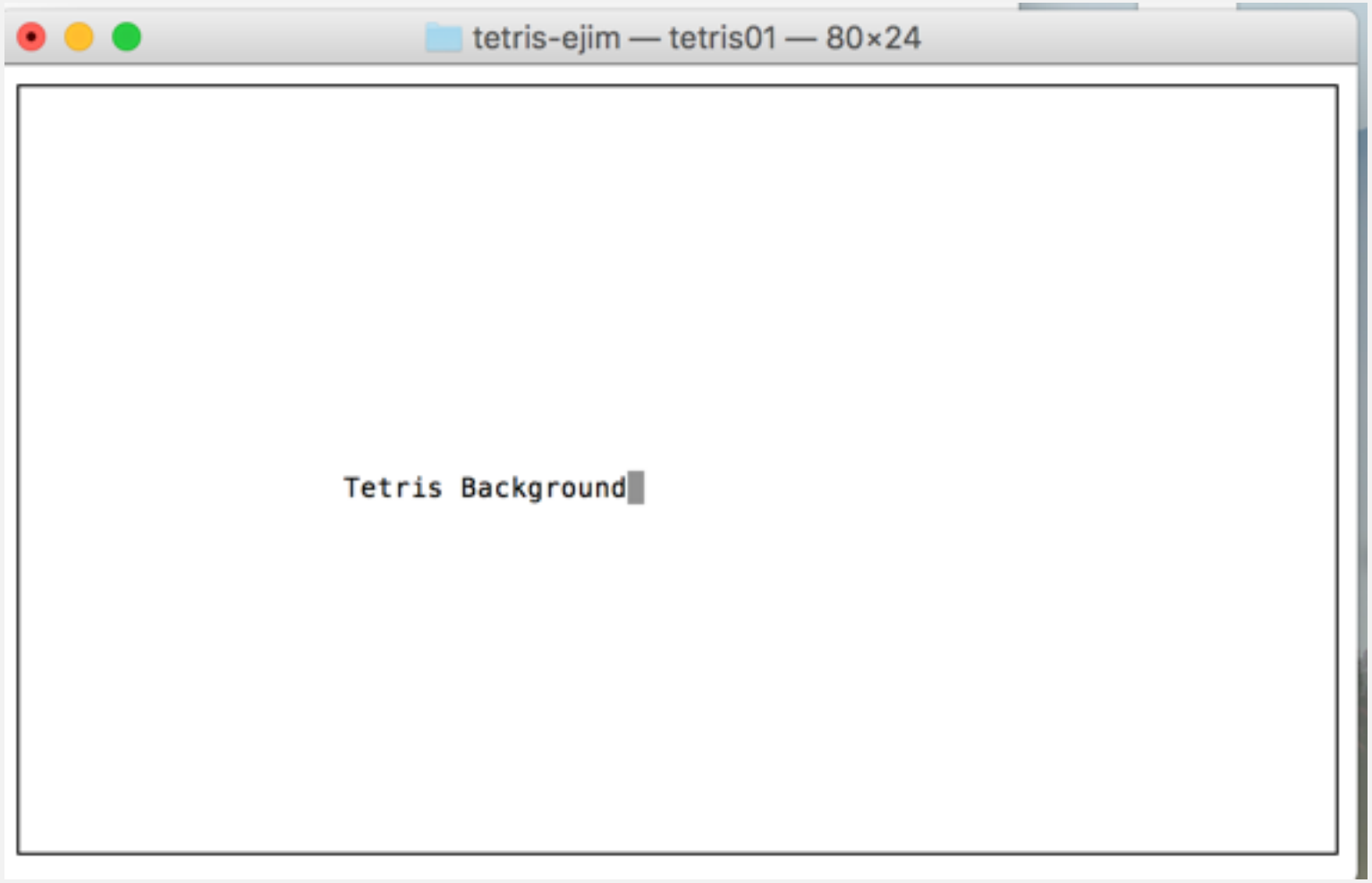
```
#include <ncurses.h>

class Pane {
  int width_, height_;
  int x_, y_;
  WINDOW *win_;
public:
  Pane(int x, int y, int w, int h) :x_(x), y_(y), width_(w), height_(h){
    win_ = newwin(h, w, y, x);
    box(win_, 0, 0);
    mvwprintw(win_, h/2, w/4,"Tetris Background");
    wrefresh(win_);
  }

  ~Pane(){
    delwin(win_);
  }
};
```

Simple tetris.cpp 1-2

```
class Tetris {
  static const int ROWS = 24;
  static const int COLS = 80;
  Pane *bgPane_;

public:
  Tetris(){
    initscr();
    refresh();

    bgPane_ = new Pane(0,0,COLS,ROWS);
  }

  ~Tetris(){
    delete bgPane_;
    endwin();
  }

  void play(){
    int input;
    input = getch();
  }
};
```

Simple tetris.cpp main() 1-3

```
int main(){
  Tetris t;
  t.play();
}
```



Simple tetris.cpp 2-1

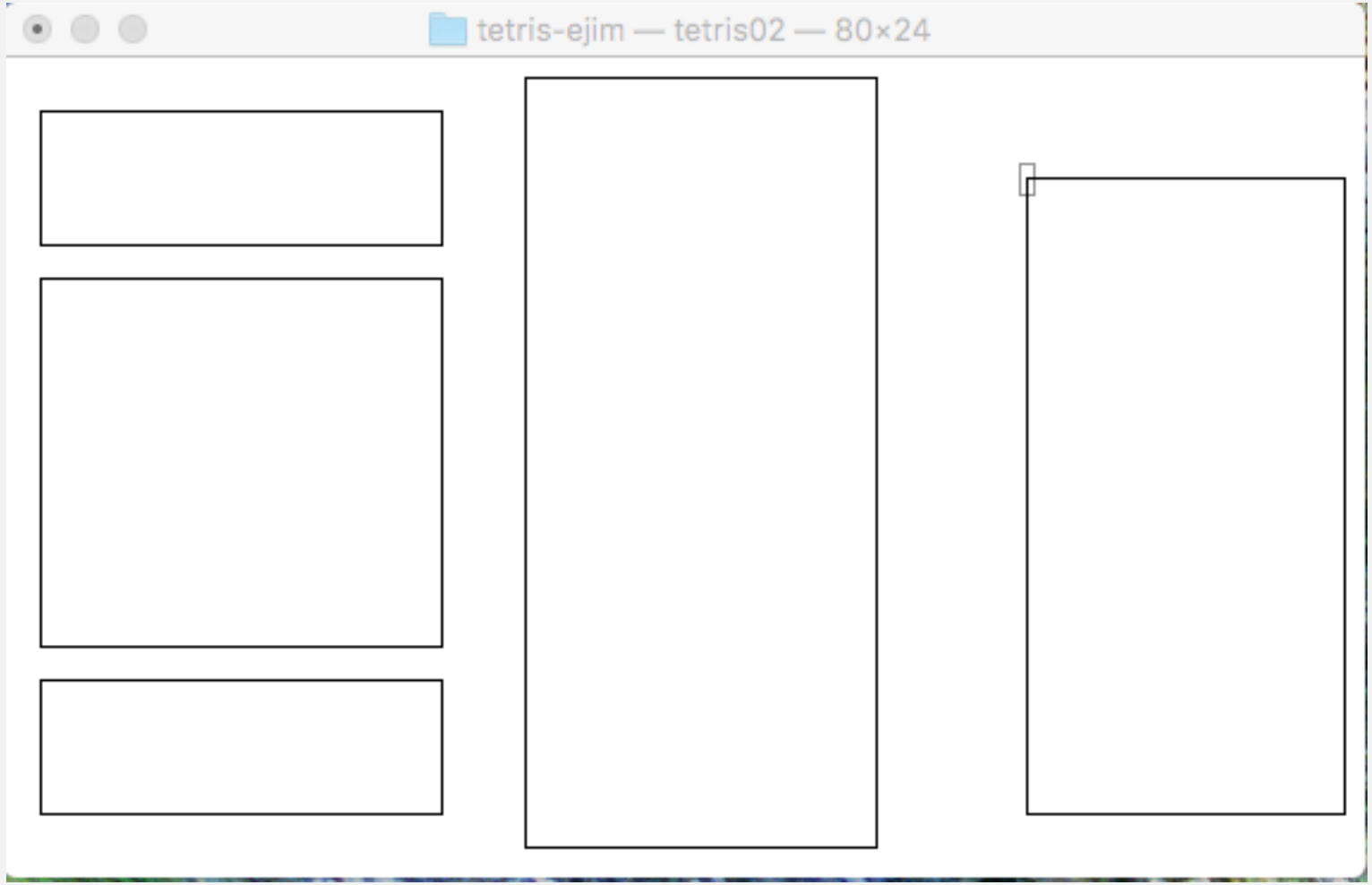
```
#include <ncurses.h>

class Pane {
  int width_, height_;
  int x_, y_;
  WINDOW *win_;
public:
  Pane(int x, int y, int w, int h) :x_(x), y_(y), width_(w), height_(h){
    win_ = newwin(h, w, y, x);
    box(win_, 0, 0);
    wrefresh(win_);
  }
  ~Pane(){
    delwin(win_);
  }
};
```

Simple tetris.cpp 2-2

```
class Tetris {
  Pane *infoPane_, *helpPane_, *nextPane_, *mainPane_, *statPane_;

public:
  Tetris(){
    initscr();
    cbreak();
    refresh();

    infoPane_ = new Pane(1,1,25,5);
    helpPane_ = new Pane(1,6,25,12);
    nextPane_ = new Pane(1,18,25,5);
    mainPane_ = new Pane(30,0,22,24);
    statPane_ = new Pane(60,3,20,20);
  }
```

Simple tetris.cpp 2-3

```
  ~Tetris(){
    delete infoPane_;
    delete helpPane_;
    delete nextPane_;
    delete mainPane_;
    delete statPane_;
    endwin();
  }
  void play(){
    int input;
    input = getch();
  }
};

int main(){
  Tetris t;
  t.play();
}
```



NCURSES EXTENDED CHARACTERS

Upper left corner	┌	ACS_ULCORNER
Lower left corner	└	ACS_LLCORNER
Upper right corner	┐	ACS_URCORNER
Lower right corner	┘	ACS_LRCORNER
Tee pointing right	├	ACS_LTEE
Tee pointing left	┤	ACS_RTEE
Tee pointing up	┴	ACS_BTEE
Tee pointing down	┬	ACS_TTEE
Horizontal line	─	ACS_HLINE
Vertical line	│	ACS_VLINE
Large Plus or cross over	┼	ACS_PLUS
Scan Line 1		ACS_S1
Scan Line 3	⎺	ACS_S3
Scan Line 7	⎽	ACS_S7
Scan Line 9	_	ACS_S9
Diamond	◆	ACS_DIAMOND
Checker board (stipple)	▒	ACS_CKBOARD
Degree Symbol	°	ACS_DEGREE
Plus/Minus Symbol	±	ACS_PLMINUS
Bullet	·	ACS_BULLET
Arrow Pointing Left	<	ACS_LARROW
Arrow Pointing Right	>	ACS_RARROW
Arrow Pointing Down	v	ACS_DARROW
Arrow Pointing Up	^	ACS_UARROW
Board of squares	#	ACS_BOARD
Lantern Symbol	␋	ACS_LANTERN
Solid Square Block	#	ACS_BLOCK
Less/Equal sign	≤	ACS_LEQUAL
Greater/Equal sign	≥	ACS_GEQUAL
Pi	π	ACS_PI
Not equal	≠	ACS_NEQUAL
UK pound sign	£	ACS_STERLING

Simple tetris.cpp 3-1

```
class Pane {
protected :
  int width_, height_;
  int x_, y_;
  WINDOW *win_;
public:
  Pane(int x, int y, int w, int h) :x_(x), y_(y), width_(w), height_(h){
    win_ = newwin(h, w, y, x);
  }

  virtual ~Pane(){
    delwin(win_);
  }
  virtual void draw(){
    box(win_, 0, 0);
    wrefresh(win_);
  }
};
```

Simple tetris.cpp 3-2

```
class InfoPane : public Pane {
public:
  InfoPane(int x, int y, int w, int h) : Pane(x,y,w,h){}
  void draw(){
    init_pair(2, COLOR_GREEN, COLOR_BLACK);
    wattron(win_, COLOR_PAIR(2));
    for (int i=0; i<height_; i++)
      mvwhline(win_, i, 0, ACS_CKBOARD, width_);
    mvwprintw(win_, 0,0, "INFO PANE");
    wattroff(win_, COLOR_PAIR(2));
    wrefresh(win_);
  }
};
```

Simple tetris.cpp 3-3

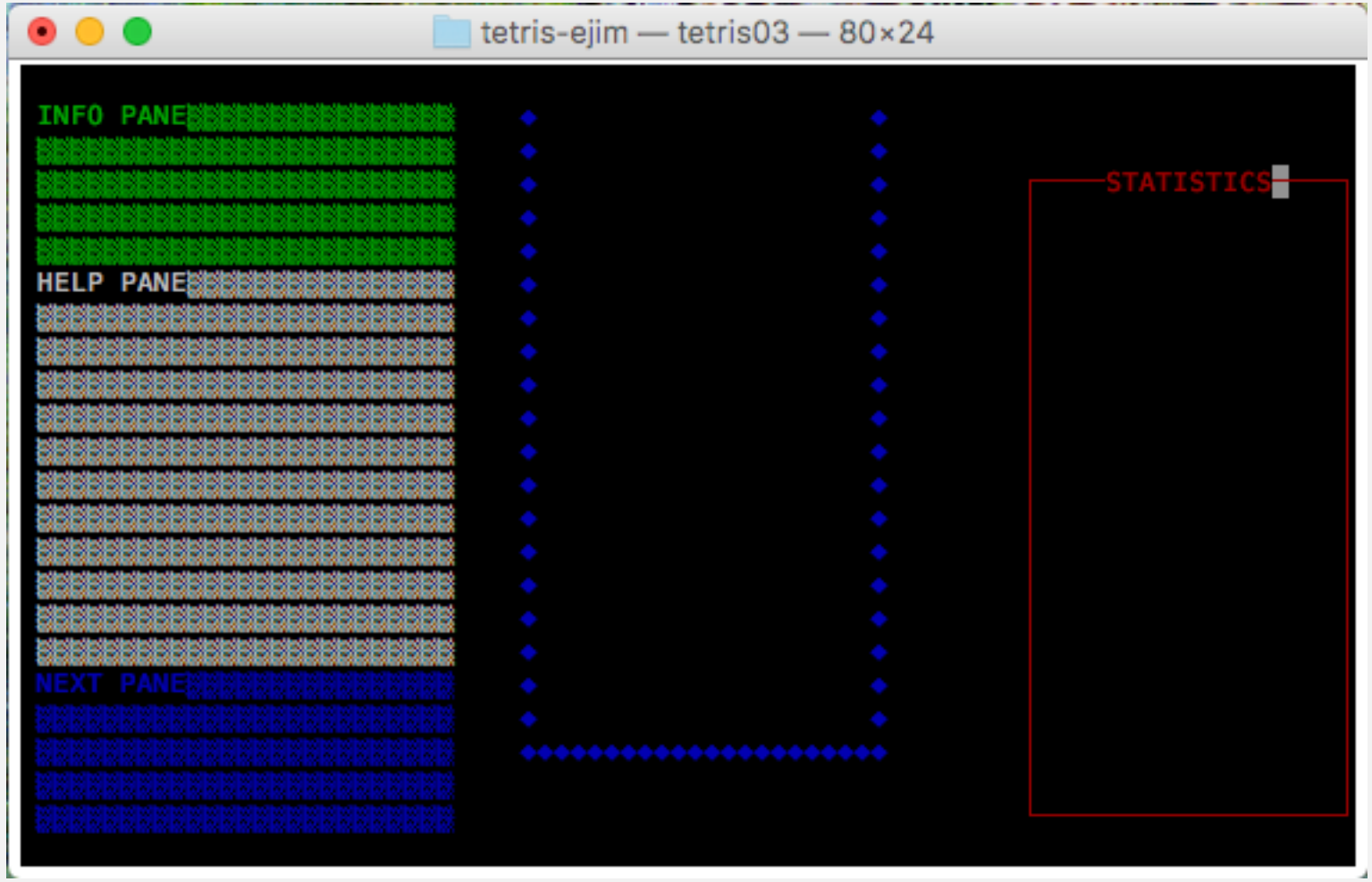
```
class BoardPane : public Pane {
public:
  BoardPane(int x, int y, int w, int h) : Pane(x,y,w,h){}
  void draw(){
    init_pair(5, COLOR_BLUE, COLOR_BLACK);
    wattron(win_, COLOR_PAIR(5));
    mvwvline(win_, 1, 0, ACS_DIAMOND, height_ - 4);
    mvwvline(win_, 1, width_-1, ACS_DIAMOND, height_ - 4);
    mvwhline(win_, height_ - 4, 0, ACS_DIAMOND, width_);
    wattroff(win_, COLOR_PAIR(5));
    wrefresh(win_);
  }
};
```

Simple tetris.cpp 3-4

```
class StatPane : public Pane {
public:
  StatPane(int x, int y, int w, int h) : Pane(x,y,w,h){}
  void draw(){
    init_pair(6, COLOR_RED, COLOR_BLACK);
    wattron(win_, COLOR_PAIR(6));
    box(win_, 0, 0);
    mvwprintw(win_, 0, 5, "STATISTICS");
    wattroff(win_, COLOR_PAIR(6));
    wrefresh(win_);
  }
};
```

Simple tetris.cpp 3-5

```
class Tetris {
  Pane *infoPane_, *helpPane_, *nextPane_, *boardPane_, *statPane_;
public:
  Tetris(){
    initscr();
    start_color();
    cbreak();
    refresh();

    infoPane_ = new InfoPane(1,1,25,5);
    helpPane_ = new HelpPane(1,6,25,12);
    nextPane_ = new NextPane(1,18,25,5);
    boardPane_ = new BoardPane(30,0,22,24);
    statPane_ = new StatPane(60,3,20,20);
  }
  ~Tetris(){
    delete infoPane_;
    delete helpPane_;
    delete nextPane_;
    delete boardPane_;
    delete statPane_;
    endwin();
  }
```



Simple tetris.cpp 3-6

```
  void play(){
    int input;
    updateScreen();
    input = getch();
  }

  void updateScreen(){
    infoPane_->draw();
    helpPane_->draw();
    nextPane_->draw();
    boardPane_->draw();
    statPane_->draw();
  }
};

int main(){
  Tetris t;
  t.play();
}
```