

## Metody i języki programowania IV semestr, projekt wykłady I

opracował Sławomir Wróblewski na podstawie materiałów ze strony:

<http://www.geniusweb.com/LDP/HOWTO/NCURSES-Programming-HOWTO/>

### Spis treści

Kompilowanie z biblioteką ncurses.....	2
The Hello World !!! Program.....	3
Używanie funkcji inicjalizujących - przykład.....	4
Funkcje wyjścia.....	5
Funkcje wejścia.....	6
Atrybuty.....	7

## **Kompilowanie z biblioteką ncurses**

```
#include <ncurses.h>
```

```
.
```

```
.
```

```
.
```

compile and link: gcc <program file> -lncurses



## Używanie funkcji inicjalizujących - przykład

```
#include <ncurses.h>
int main()
{
    int ch;

    initscr();          /* Start curses mode */

    raw();              /* Line buffering disabled*/

    keypad(stdscr, TRUE); /* We get F1, F2 etc..*/

    noecho();          /* Don't echo() while we do getch */

   printw("Type any character to see it in bold\n");

    ch = getch();      /* If raw() hadn't been called
                       * we have to press enter before it
                       * gets to the program */

    if(ch == KEY_F(1)) /* Without keypad enabled this will */

        printw("F1 Key pressed"); /* not get to us either*/
                                   /* Without noecho() some ugly escape
                                   * charachters might have been printed
                                   * on screen*/

    else
    {
        printw("The pressed key is ");
        attron(A_BOLD);
        printw("%c", ch);
        attroff(A_BOLD);
    }

    refresh();         /* Print it on to the real screen */

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

    return 0;
}
```

## Funkcje wyjścia

`addch()` drukuje znak z atrybutami (podkrślenie, bold itp)

`printw()` analogiczna jak `printf`

`addstr()` drukuje napisy

---

**`mvaddch()`** jest używana do ustawienia kursora w zadanym punkcie i wydrukowania w nim znaku:  
poniższe wywołania dwóch funkcji

```
move(row,col);           /* moves the cursor to rowth row and colth column */
```

```
addch(ch);
```

mogą być zastąpione przez jedno:

```
mvaddch(row,col,ch);
```

Funkcje **`waddch()`** i **`mvwaddch()`** działają analogicznie jak **`addch()`** i **`mvaddch()`**, operują na oknie zadanym jako argument.

---

`printw()` i `mvprintw()`- formatowne wyjście na `stdscr`

`wprintw()` i `mvwprintw()`- formatowne wyjście na okno zadane jako argument

`vwprintw()` - działa analogicznie jak `vprintf()`

Przykład:

```
#include <ncurses.h> /* ncurses.h includes stdio.h */
```

```
#include <string>
```

```
int main()
```

```
{
```

```
    char mesg[]="Just a string";           /* message to be appeared on the screen */
```

```
    int row,col;                          /* to store the number of rows and */
```

```
                                           /* the number of columns of the screen */
```

```
    initscr();                             /* start the curses mode */
```

```
    getmaxyx(stdscr,row,col);             /* get the number of rows and columns */
```

```
    mvprintw(row/2,(col-strlen(mesg))/2,"%s",mesg);
```

```
                                           /* print the message at the center of the screen */
```

```
    mvprintw(row-2,0,"This screen has %d rows and %d columns\n",row,col);
```

```
    printf("Try resizing your window(if possible) and then run this program again");
```

```
    refresh();
```

```
    getch();
```

```
    endwin();
```

```
    return 0;
```

```
}
```

## Funkcje wejścia

getch() czyta znak

scanw() czyta sformatowane wejście

getstr() czyta napisy

scanw() and mvscanw() - formatowne wejście z stdscr

wscanw() and mvwscanw() - formatowne wejście z okna określonego jako argument

vwscanw() - działa analogicznie jak vscanf()

Przykład:

```
#include <ncurses.h> /* ncurses.h includes stdio.h */
```

```
#include <string>
```

```
int main()
```

```
{
```

```
    char mesg[]="Enter a string: ";    /* message to be appeared on the screen */
```

```
    char str[80];
```

```
    int row,col;    /* to store the number of rows and *
```

```
                    * the number of columns of the screen */
```

```
    initscr();    /* start the curses mode */
```

```
    getmaxyx(stdscr,row,col);    /* get the number of rows and columns */
```

```
    mvprintw(row/2,(col-strlen(mesg))/2,"%s",mesg);
```

```
                    /* print the message at the center of the screen */
```

```
    getstr(str);
```

```
    mvprintw(LINES - 2, 0, "You Entered: %s", str);
```

```
    getch();
```

```
    endwin();
```

```
    return 0;
```

```
}
```

## Atrybuty

Przykład: Program czyta plik źródłowy języka c, wyświetla jego zawartość z komentarzami wyświetlanymi pogrubioną czcionką

```
#include <ncurses.h>
int main(int argc, char *argv[])
{
    int ch, prev;
    FILE *fp;
    int goto_prev = FALSE, y, x;
    if(argc != 2)
    {
        printf("Usage: %s <a c file name>\n", argv[0]);
        exit(1);
    }
    fp = fopen(argv[1], "r");
    if(fp == NULL)
    {
        perror("Cannot open input file");
        exit(1);
    }

    initscr();          /* Start curses mode      */
    prev = EOF;
    while((ch = fgetc(fp)) != EOF)
    {
        if(prev == '/' && ch == '*') /* If it is / and * then only
                                     * switch bold on */
        {
            attron(A_BOLD);
            goto_prev = TRUE; /* Go to previous char / and
                               * print it in BOLD */
        }

        if(goto_prev == TRUE)
        {
            getyx(stdscr, y, x);
            move(y, x - 1);
            printw("%c%c", '/', ch); /* The actual printing is done
                                       * here */
            ch = 'a'; /* 'a' is just a dummy
                       * character to prevent */
                       // "/*" comments.
            goto_prev = FALSE; /* Set it to FALSE or every
                               * thing from here will be */
        }
        else
            printw("%c", ch);

        refresh();

        if(prev == '*' && ch == '/')
            attroff(A_BOLD); /* Switch it off once we got *
                               and then */
    }
}
```

```

        prev = ch;
    }

    getch();
    endwin();          /* End curses mode          */
    return 0;
}

```

**getyx()** i **move()** - odpowiednio pobiera pozycje kursora, ustawia pozycje kursora

atrybuty wideo:

A_NORMAL	Normal display (no highlight)
A_STANDOUT	Best highlighting mode of the terminal.
A_UNDERLINE	Underlining
A_REVERSE	Reverse video
A_BLINK	Blinking
A_DIM	Half bright
A_BOLD	Extra bright or bold
A_PROTECT	Protected mode
A_INVIS	Invisible or blank mode
A_ALTCHARSET	Alternate character set
A_CHARTEXT	Bit-mask to extract a character
COLOR_PAIR(n)	Color-pair number n

**attron()**, **attroff()**, - odpowiednio włącza / wyłącza zadane atrybuty

Przykład:

```
attron(A_REVERSE | A_BLINK);
```

**attrset()** - nadpisuje nowe atrybuty dla całego okna

**attr\_get()** - pobiera aktualne atrybuty i kolory okna

**chgat()** - zmienia atrybuty grupie znaków począwszy od bieżącego położenia kursora ()

przykład:

```
chgat(-1,A_REVERSE, 0 , NULL);
```

1 parametr – ilość znaków do zmiany (gdy -1 oznacza do końca linii)

2 parametr – atrybut

3 parametr – kolor

4 parametr – nieużywany, powinien być NULL

Uwaga!

analogiczne funkcje operujące na oknach posiadają w nazwie przedrostek 'w', np. **wattrset()**

d