

รายการอ้างอิง

ภาษาไทย

ทักษิณา สวานานนท์. พจนานุกรมศัพท์คอมพิวเตอร์. กรุงเทพฯ:บริษัทดวงกมลสมัย จำกัด, 2536.

สมนึก เจียมเจริญเดช. การพัฒนาโปรแกรมเลียนแบบเทอร์มินัลภาษาไทย. กรุงเทพฯ:คณะวิศวกรรม-
ศาสตร์, 2533.

ภาษาอังกฤษ

AT&T, UNIX System V Release 4.0 Programmer's Reference Manual. Santa Clara, California USA:-
Intel Corporation, 1990.

Berny Goodhard. Unix Curses Explained. Australia:Prentice Hall, 1991.

John Strang, Linda Mui, and Tim O'Reilly. Termcap & Terminfo. United States of America:-
O'Reilly & Associates, Inc., 1991.

Keith Haviland, Ben Salama. UNIX System Programming. Workingham, England:Addison-
Wesley Publishing Company, 1987.

Ramkrishna S. Tare, UNIX Utilities. Singapore:McGraw-Hill Book Company, 1988.

W. Richard Stevens, Advanced Programming. Workingham, England:Addison Wesley Publishing-
Company, 1992.

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ภาคผนวก ก

ฟังก์ชันของเคิร์ส 2 มิติ

<u>ฟังก์ชัน</u>	void beep()
<u>คำอธิบาย</u>	ส่งเสียง beep
<u>ผลลัพธ์</u>	ไม่มี
<u>ฟังก์ชัน</u>	void box(WINDOW *win, char vert, char hor),
<u>คำอธิบาย</u>	ตีกรอบสำหรับวินโดว์ โดยใช้ตัวอักษร vert เป็นเส้นตามแนวตั้ง ใช้ตัวอักษร hor เป็นเส้นตามแนวนอน
<u>ผลลัพธ์</u>	ไม่มี
<u>ฟังก์ชัน</u>	int can_change_color()
<u>คำอธิบาย</u>	ตรวจสอบว่าสามารถเปลี่ยนสีได้หรือไม่
<u>ผลลัพธ์</u>	1 = เปลี่ยนสีได้ 0 = ไม่สามารถเปลี่ยนสีได้
<u>ฟังก์ชัน</u>	void cbreak()
<u>คำอธิบาย</u>	กำหนดให้เทอร์มินัลอยู่ใน cbreak mode
<u>ผลลัพธ์</u>	ไม่มี
<u>ฟังก์ชัน</u>	int char_type(char ch)
<u>คำอธิบาย</u>	ให้ค่าประเภทของ ch ว่าเป็น สระบน-สระล่าง ตัวอักษร-สระกลาง หรือ วรรณยุกต์
<u>ผลลัพธ์</u>	0 = เมื่อ ch เป็น สระบน หรือ สระล่าง 1 = เมื่อ ch เป็นวรรณยุกต์ 2 = เมื่อ ch เป็นตัวอักษร 3 = เมื่อมีข้อผิดพลาด
<u>ฟังก์ชัน</u>	int check_color(short color)
<u>คำอธิบาย</u>	เช็คว่าสามารถใช้สี color ได้หรือไม่
<u>ผลลัพธ์</u>	0 = ใช้ได้ 1 = ไม่มีสีที่ต้องการ

ประวัติผู้เขียน

นางสาวนิลวรรณ ชีวะประเสริฐ เกิดวันที่ 13 กันยายน พ.ศ. 2509 ที่เขตพญาไท กรุงเทพมหานคร สำเร็จการศึกษาปริญญาตรีวิทยาศาสตร์บัณฑิต สาขาฟิสิกส์ ภาควิชาฟิสิกส์ คณะวิทยาศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ วิทยาเขตบางแสน ในปีการศึกษา 2530 ปริญญาตรีวิทยาศาสตร์บัณฑิต สาขาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยมหิดล ในปีการศึกษา 2534 และเข้าศึกษาต่อ ในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิทยาศาสตร์คอมพิวเตอร์ ที่จุฬาลงกรณ์มหาวิทยาลัย เมื่อ พ.ศ. 2535 ปัจจุบันทำงานด้านคอมพิวเตอร์ ที่บริษัทอโตเมชั่นเวอร์กัฟ จำกัด



ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

<u>ฟังก์ชัน</u>	int check_colorpair(int colorpair)
<u>คำอธิบาย</u>	เช็คที่สามารถใช้คู่ของสี colorpair ได้หรือไม่
<u>ผลลัพธ์</u>	0 = ใช้ได้ 1 = ไม่มีคู่ของสีที่ต้องการ
<u>ฟังก์ชัน</u>	int copywin(WINDOW *srcwin, WINDOW *dstwin, int minrowsrc, int mincolsrc, int minrowdst, int mincoldst, int maxrowdst, int maxcoldst, int overlay)
<u>คำอธิบาย</u>	ทำสำเนาวินโดว์จาก srcwin ไป dstwin จากแถวที่ minrowsrc คอลัมน์ที่ mincolsrc ของ srcwin ไปยัง แถวที่ minrowdst ถึง maxrowdst คอลัมน์ที่ mincoldst ถึง maxcoldst
<u>ผลลัพธ์</u>	0 = สำเร็จ 1 = มีข้อผิดพลาด
<u>ฟังก์ชัน</u>	int cup(int row, int col)
<u>คำอธิบาย</u>	ส่งคำสั่งไปยังเทอร์มินัลให้เลื่อนเคอร์เซอร์ไปที่แถวที่ row คอลัมน์ที่ col
<u>ผลลัพธ์</u>	0 เสมอ
<u>ฟังก์ชัน</u>	int dch1()
<u>คำอธิบาย</u>	ส่งคำสั่งไปยังเทอร์มินัลให้ ลบหนึ่งตัวอักขระ ที่ตำแหน่งปัจจุบัน
<u>ผลลัพธ์</u>	0 เสมอ
<u>ฟังก์ชัน</u>	int delwin(win)
<u>คำอธิบาย</u>	ลบวินโดว์ win และคืนพื้นที่ในหน่วยความจำที่วินโดว์นั้นใช้
<u>ผลลัพธ์</u>	0 = สำเร็จ 1 = มีข้อผิดพลาด
<u>ฟังก์ชัน</u>	int doupdate()
<u>คำอธิบาย</u>	ฟังก์ชันนี้จะปรับปรุงส่วนที่มีการเปลี่ยนแปลงของวินโดว์ curscr ให้มีผลกับ physical screen ของเทอร์มินัล
<u>ผลลัพธ์</u>	จำนวนตัวอักขระที่มีการส่งให้ terminal
<u>ฟังก์ชัน</u>	void echo()
<u>คำอธิบาย</u>	กำหนดให้มีการแสดงผลตัวอักขระที่รับเข้ามาทางแป้นพิมพ์
<u>ผลลัพธ์</u>	ไม่มี
<u>ฟังก์ชัน</u>	void ed()
<u>คำอธิบาย</u>	ส่งคำสั่งไปยังเทอร์มินัลให้ ลบตั้งแต่ตำแหน่งปัจจุบันไปจนท้ายจอ
<u>ผลลัพธ์</u>	0 เสมอ

ฟังก์ชัน	<code>void endwin()</code>
คำอธิบาย	จบการใช้เคิร์สฟังก์ชัน
ผลลัพธ์	ไม่มี
ฟังก์ชัน	<code>void flash()</code>
คำอธิบาย	flash จอภาพ
ผลลัพธ์	ไม่มี
ฟังก์ชัน	<code>void flushinp(void)</code>
คำอธิบาย	ล้าง input buffer
ผลลัพธ์	ไม่มี
ฟังก์ชัน	<code>int get_pairno(short attr)</code>
คำอธิบาย	ให้ค่า pairno
ผลลัพธ์	ค่า pairno
ฟังก์ชัน	<code>int getmode()</code>
คำอธิบาย	กำหนดค่าเริ่มต้นสำหรับเทอร์มินัล และตั้งค่าสำหรับ mode ต่างๆ
ผลลัพธ์	0 เสมอ
ฟังก์ชัน	<code>void halfdelay(int tenths)</code>
คำอธิบาย	กำหนดให้มีการรอ input เป็นจำนวน tenths เท่าของ 1/10 วินาที
ผลลัพธ์	ไม่มี
ฟังก์ชัน	<code>int has_colors()</code>
คำอธิบาย	ตรวจสอบว่าสามารถใช้สีได้หรือไม่
ผลลัพธ์	1 = ใช้สีได้ 0 = ไม่สามารถใช้สีได้
ฟังก์ชัน	<code>int ich1()</code>
คำอธิบาย	ส่งคำสั่งไปยังเทอร์มินัลให้แทรกช่องว่าง 1 ช่อง ที่ตำแหน่งปัจจุบัน

ผลลัพธ์ 0 เสมอ

ฟังก์ชัน int init_pair(int colorpair, short foreground, short background)
คำอธิบาย กำหนดคู่ของสี โดยบอกลำดับที่ของคู่สี colorpair สีสำหรับ foreground และสีสำหรับ background

ผลลัพธ์ 0 = สำเร็จ
 1 = มีข้อผิดพลาด

ฟังก์ชัน WINDOW *initscr()
คำอธิบาย สำหรับ กำหนดค่าเริ่มต้นของเคิร์ลโปรแกรม ต้องเรียกเพียงครั้งเดียวในโปรแกรม
ผลลัพธ์ stdscr ถ้าสำเร็จ
 ผลลัพธ์ NULL ถ้าผิดพลาด

ฟังก์ชัน int isendwin(void)
คำอธิบาย เช็คว่ามีการเรียกใช้ฟังก์ชัน endwin() หรือยัง
ผลลัพธ์ 1 = มีการ endwin แล้ว
 0 = ยังไม่มีการ endwin

ฟังก์ชัน void keypad(WINDOW *win, int boolf)
คำอธิบาย กำหนดว่าต้องการใช้ keypad หรือไม่
ผลลัพธ์ ไม่มี

ฟังก์ชัน int mvprintw(int y, int x, char *fmt, ...)
คำอธิบาย เลื่อนตำแหน่งปัจจุบันไปที่แถวที่ y คอลัมน์ที่ x และพิมพ์ output ตามรูปแบบของ fmt ไปที่วินโดว์ stdscr
ผลลัพธ์ ให้ค่า 0 เสมอ

ฟังก์ชัน int mvscanw(register int y, register int x, char *fmt,...)
คำอธิบาย เลื่อนตำแหน่งปัจจุบันไปที่ แถวที่ y คอลัมน์ที่ x พร้อมทั้งอ่านตัวอักษรตามรูปแบบ fmt จากวินโดว์ stdscr
ผลลัพธ์ จำนวนตัวอักษรที่มีการอ่านได้

<u>ฟังก์ชัน</u>	<code>int mvwin(WINDOW *win, int y, int x)</code>
<u>คำอธิบาย</u>	เลื่อนตำแหน่งวินโดว์ win ไปที่ตำแหน่ง x, y
<u>ผลลัพธ์</u>	ผลลัพธ์ค่า 0 เสมอ
<u>ฟังก์ชัน</u>	<code>int mvwprintw(WINDOW *win, int y, int x, char *fmt, ...)</code>
<u>คำอธิบาย</u>	เลื่อนตำแหน่งปัจจุบันไปที่แถวที่ y คอลัมน์ที่ x และพิมพ์ output ตามรูปแบบของ fmt ไปที่วินโดว์ win
<u>ผลลัพธ์</u>	ให้ค่า 0 เสมอ
<u>ฟังก์ชัน</u>	<code>int mvwscanw(WINDOW *win, int y, int x, char *fmt, ...)</code>
<u>คำอธิบาย</u>	เลื่อนตำแหน่งปัจจุบันไปที่ แถวที่ y คอลัมน์ที่ x พร้อมทั้งอ่านตัวอักขระตามรูปแบบ fmt จากวินโดว์ win
<u>ผลลัพธ์</u>	จำนวนตัวอักขระที่มีการอ่านได้
<u>ฟังก์ชัน</u>	<code>WINDOW *newwin(int nl, int nc, int by, int bx)</code>
<u>คำอธิบาย</u>	สร้างวินโดว์ใหม่
<u>ผลลัพธ์</u>	pointer to WINDOW ถ้าสำเร็จ NULL ถ้ามีข้อผิดพลาด
<u>ฟังก์ชัน</u>	<code>void nl(void)</code>
<u>คำอธิบาย</u>	กำหนดให้มี newline ใน input และ output stream
<u>ผลลัพธ์</u>	ไม่มี
<u>ฟังก์ชัน</u>	<code>void nocbreak()</code>
<u>คำอธิบาย</u>	กำหนดให้เทอร์มินัลออกจาก cbreak mode
<u>ผลลัพธ์</u>	ไม่มี
<u>ฟังก์ชัน</u>	<code>void noecho()</code>
<u>คำอธิบาย</u>	กำหนดให้ไม่ต้องแสดงผลตัวอักขระที่รับเข้ามาทางแป้นพิมพ์
<u>ผลลัพธ์</u>	ไม่มี

ฟังก์ชัน void nonl(void)
คำอธิบาย กำหนดไม่ให้มี newline ใน input และ output stream
ผลลัพธ์ ไม่มี

ฟังก์ชัน void noraw()
คำอธิบาย กำหนดให้เทอร์มินัลออกจาก raw mode
ผลลัพธ์ ไม่มี

ฟังก์ชัน int overlay(WINDOW *win1, WINDOW *win2)
คำอธิบาย เขียนวินโดว์ win1 ทับบนวินโดว์ win2
ผลลัพธ์ 0=สำเร็จ
 1=มีข้อผิดพลาด

ฟังก์ชัน int overwrite(WINDOW *win1, WINDOW *win2)
คำอธิบาย เขียนวินโดว์ win1 ทับบนวินโดว์ win2
ผลลัพธ์ 0=สำเร็จ
 1=มีข้อผิดพลาด

ฟังก์ชัน int pair_content(int pair, short *foreground, short *background)
คำอธิบาย ขอค่า foreground color และ background color ของคู่สีลำดับที่ pair โดยเก็บค่าของสีไว้ที่ foreground และ background ตามลำดับ
ผลลัพธ์ 0 = สำเร็จ
 1 = มีข้อผิดพลาด

ฟังก์ชัน int printw(char *fmt, ...)
คำอธิบาย พิมพ์ output ตามรูปแบบของ fmt ไปที่วินโดว์ stdscr
ผลลัพธ์ ให้ค่า 0 เสมอ

ฟังก์ชัน void raw()
คำอธิบาย กำหนดให้เทอร์มินัลอยู่ใน raw mode
ผลลัพธ์ ไม่มี

ฟังก์ชัน	<code>int scanw(char *fmt, ...)</code>
คำอธิบาย	อ่านตัวอักขระตามรูปแบบ <code>fmt</code> จากวินโดว์ <code>stdscr</code>
ผลลัพธ์	จำนวนตัวอักขระที่มีการอ่านได้
ฟังก์ชัน	<code>void scroll(WINDOW *win)</code>
คำอธิบาย	เลื่อนวินโดว์ ขึ้น 1 บรรทัด
ผลลัพธ์	ไม่มี
ฟังก์ชัน	<code>int scroll0(WINDOW *win, int boolf)</code>
คำอธิบาย	ตั้งค่าให้วินโดว์ scroll ได้
ผลลัพธ์	0 เสมอ
ฟังก์ชัน	<code>int setterm(char *term)</code>
คำอธิบาย	เรียกฟังก์ชัน กำหนดค่าเริ่มต้นของ <code>terminfo</code>
ผลลัพธ์	0 = สำเร็จ 1 = มีข้อผิดพลาด
ฟังก์ชัน	<code>int start_color()</code>
คำอธิบาย	เช็คที่สามารถใช้สีได้หรือไม่ และเตรียมตัวแปรสำหรับฟังก์ชันสี
ผลลัพธ์	0 = สำเร็จ 1 = มีข้อผิดพลาด หรือ ไม่สามารถใช้สีได้
ฟังก์ชัน	<code>void touchwin(WINDOW *win)</code>
คำอธิบาย	ฟังก์ชันทำเสมือนว่าวินโดว์ <code>win</code> มีการเปลี่ยนแปลงทั้งวินโดว์
ผลลัพธ์	ไม่มี
ฟังก์ชัน	<code>void ungetch(int key)</code>
คำอธิบาย	ส่งค่า <code>key</code> กลับไปยัง <code>input buffer</code> ของเคอร์ส
ผลลัพธ์	ไม่มี
ฟังก์ชัน	<code>int waddalpha(WINDOW *win, int ch)</code>

คำอธิบาย เพิ่มตัวอักษรกลาง ch ลงในบัฟเฟอร์ ที่เก็บข้อมูลสำหรับการแสดงผล ของวินโดว์ win และกำหนด attribute ของคอลัมน์ให้เท่ากับ current attribute ของวินโดว์

ผลลัพธ์
0 = เพิ่มสำเร็จ
1 = มีข้อผิดพลาด

ฟังก์ชัน int waddch(WINDOW win, int ch)

คำอธิบาย เพิ่มตัวอักษร ch ลงในบัฟเฟอร์ ที่เก็บข้อมูลสำหรับการแสดงผล ของวินโดว์ win โดยเรียกใช้ waddalpha กรณีที่ ch เป็นตัวอักษร เรียกใช้ wadvowel กรณีที่ ch เป็นสระบนหรือสระล่าง และเรียกใช้ waddwanayut กรณีที่ ch เป็นวรรณยุกต์

ผลลัพธ์
0 = เพิ่มสำเร็จ
1 = มีข้อผิดพลาด

ฟังก์ชัน int waddstr(WINDOW win, char *str)

คำอธิบาย เพิ่ม str ลงในบัฟเฟอร์ ที่เก็บข้อมูลสำหรับการแสดงผล ของวินโดว์ win โดยเรียกใช้ waddch

ผลลัพธ์
0 = เพิ่มสำเร็จ
1 = มีข้อผิดพลาด

ฟังก์ชัน int wadvowel(WINDOW *win, int ch)

คำอธิบาย เพิ่มสระบน หรือสระล่าง ลงในบัฟเฟอร์ ที่เก็บข้อมูลสำหรับการแสดงผล ของวินโดว์ win

ผลลัพธ์
0 = เพิ่มสำเร็จ
1 = มีข้อผิดพลาด ซึ่งอาจเกิดจาก

- มีวรรณยุกต์อยู่ในคอลัมน์นั้นก่อนแล้ว
- ในคอลัมน์นั้นไม่มีตัวอักษรที่อยู่ในบรรทัดมาก่อน
- อ้างคอลัมน์ไม่ถูกต้อง

ฟังก์ชัน int waddwanayut(WINDOW *win, int ch)

คำอธิบาย เพิ่มวรรณยุกต์ ลงในบัฟเฟอร์ ที่เก็บข้อมูลสำหรับการแสดงผล ของวินโดว์ win

ผลลัพธ์
0 = เพิ่มสำเร็จ
1 = มีข้อผิดพลาด ซึ่งอาจเกิดจาก

- ในคอลัมน์นั้นไม่มีตัวอักษรที่อยู่ในบรรทัดมาก่อน
- อ้างคอลัมน์ไม่ถูกต้อง

ฟังก์ชัน	<code>int wattroff(WINDOW *win, short attrs)</code>
คำอธิบาย	ลบ attrs ออกจาก video attribute ของวินโดว์ win
ผลลัพธ์	0 เสมอ
ฟังก์ชัน	<code>int wattron(WINDOW *win, short attrs)</code>
คำอธิบาย	ตั้งให้ video attribute ของวินโดว์ win มี attribute attrs เพิ่มเข้ามา
ผลลัพธ์	0 เสมอ
ฟังก์ชัน	<code>int wattrset(WINDOW *win, short attrs)</code>
คำอธิบาย	กำหนดให้ video attribute ของวินโดว์ win มีค่าเท่ากับ attrs
ผลลัพธ์	0 เสมอ
ฟังก์ชัน	<code>int wclear(WINDOW *win)</code>
คำอธิบาย	ลบ win ด้วยช่องว่าง ถ้าเป็นวินโดว์เต็มจอ จะให้ค่า win->_clear = 1 ทำให้เมื่อมีการสั่ง refresh() refresh จะส่ง รหัสควบคุมไปลบจอภาพที่เทอร์มินัล
ผลลัพธ์	ผลลัพธ์ค่า 0 เสมอ
ฟังก์ชัน	<code>void clear0(WINDOW *win, int boolf)</code>
คำอธิบาย	ให้ค่า win->_clear เท่ากับ boolf ถ้า boolf เป็นจริง เมื่อมีการสั่ง refresh() refresh จะส่ง รหัสควบคุมไปลบจอภาพที่เทอร์มินัล
ผลลัพธ์	ไม่มี
ฟังก์ชัน	<code>int wclrtoeol(WINDOW *win)</code>
คำอธิบาย	ลบจอภาพด้วยช่องว่าง ตั้งแต่ที่เคอร์เซอร์อยู่ไปจนถึงสิ้นสุดจอภาพ
ผลลัพธ์	ไม่มี
ฟังก์ชัน	<code>int wclrtoeol(WINDOW *win)</code>
คำอธิบาย	ลบตัวอักษรกลาง สระบนหรือสระล่าง และวรรณยุกต์ตั้งแต่ตำแหน่งปัจจุบันบนวินโดว์ win ไปจนถึงสิ้นสุดบรรทัด
ผลลัพธ์	0 = สำเร็จ 1 = มีข้อผิดพลาด

ฟังก์ชัน	<code>int wdech(WINDOW win)</code>
คำอธิบาย	ลบตัวอักษรกลาง สระบนหรือสระล่าง และวรรณยุกต์ที่อยู่ในตำแหน่งปัจจุบัน บนวินโดว์ win พร้อมทั้งขยับตัวอักษรทั้งหมดตั้งแต่ตำแหน่งปัจจุบัน ไปจนถึงสิ้นสุดบรรทัด มาแทนที่ตำแหน่งที่ลบไป
ผลลัพธ์	0 = สำเร็จ 1 = มีข้อผิดพลาด
ฟังก์ชัน	<code>void wdeleteln(WINDOW *win)</code>
คำอธิบาย	ลบบรรทัดที่อยู่ ณ ตำแหน่ง y ปัจจุบัน บนวินโดว์ win พร้อมทั้งเลื่อนบรรทัดถัดๆ ไปขึ้นมา
ผลลัพธ์	ไม่มี
ฟังก์ชัน	<code>int wdelvowel(WINDOW win)</code>
คำอธิบาย	ลบสระบนหรือสระล่างที่อยู่ในตำแหน่งปัจจุบัน บนวินโดว์ win โดยแทนที่ด้วยช่องว่าง ณ ตำแหน่งของสระ
ผลลัพธ์	0 = สำเร็จ 1 = มีข้อผิดพลาด
ฟังก์ชัน	<code>int wdelwanayut(WINDOW win)</code>
คำอธิบาย	ลบวรรณยุกต์ที่อยู่ในตำแหน่งปัจจุบัน บนวินโดว์ win โดยแทนที่ด้วยช่องว่าง ณ ตำแหน่งของวรรณยุกต์
ผลลัพธ์	0 = สำเร็จ 1 = มีข้อผิดพลาด
ฟังก์ชัน	<code>int wechochar(WINDOW win, int ch)</code>
คำอธิบาย	เพิ่มตัวอักษร ch ลงในบัฟเฟอร์ ที่เก็บข้อมูลสำหรับการแสดงผล ของวินโดว์ win โดยเรียกใช้ <code>waddch</code> และทำการ <code>refresh</code> จอภาพทันที
ผลลัพธ์	1 เสมอ
ฟังก์ชัน	<code>int werase(WINDOW *win)</code>
คำอธิบาย	ลบ win ด้วยช่องว่าง
ผลลัพธ์	ผลลัพธ์ค่า 0 เสมอ

ฟังก์ชัน int wgetch(WINDOW *win)
คำอธิบาย อ่านค่าตัวอักษรที่ส่งจากเทอร์มินัลที่สัมพันธ์กับวินโดว์ win ถ้าอยู่ใน cbreak() mode โปรแกรมจะหยุดรอจนกว่าจะมีตัวอักษรเข้ามา ถ้าอยู่ใน nocbreak() mode โปรแกรมจะหยุดรอจนกว่าจะพบ newline
 ถ้าไม่กำหนด noecho() ตัวอักษรที่รับมาได้จะแสดงบนจอภาพด้วย
ผลลัพธ์ ผลลัพธ์ตัวอักษรที่อ่านได้
 -1 = มีความผิดพลาด

ฟังก์ชัน int wgetstr(WINDOW *win, char str[])
คำอธิบาย เรียก wgetch() จนได้รับ carriage return
ผลลัพธ์ 0 = สำเร็จ
 1 = มีข้อผิดพลาด

ฟังก์ชัน int winattr(WINDOW *win)
คำอธิบาย ให้ค่า attribute ณ ตำแหน่งปัจจุบัน ของวินโดว์ win
ผลลัพธ์ ค่า attribute เป็น int

ฟังก์ชัน int winch(WINDOW *win)
คำอธิบาย ให้ค่า ตัวอักษรกลางหรือสระกลาง ณ ตำแหน่งปัจจุบัน ของวินโดว์ win
ผลลัพธ์ ค่าตัวอักษรเป็น int

ฟังก์ชัน int wrefresh(WINDOW *win, int ch)
คำอธิบาย แทรก ch ลงในบัฟเฟอร์ ที่เก็บข้อมูลสำหรับการแสดงผล ของวินโดว์ win โดยเรียกใช้ waddch และทำการ refresh จอภาพทันที
ผลลัพธ์ 0 = เพิ่มสำเร็จ
 1 = มีข้อผิดพลาด

ฟังก์ชัน int winsertln(WINDOW *win)
คำอธิบาย เพิ่มบรรทัดเหนือบรรทัดปัจจุบัน
ผลลัพธ์ 0=สำเร็จ
 1=มีข้อผิดพลาด

<u>ฟังก์ชัน</u>	<code>int winvowel(WINDOW *win)</code>
<u>คำอธิบาย</u>	ให้ค่า ตัวสระบน หรือ สระล่าง ณ ตำแหน่งปัจจุบัน ของวินโดว์ win
<u>ผลลัพธ์</u>	ค่าตัวอักษรเป็น int
<u>ฟังก์ชัน</u>	<code>int winwanayut(WINDOW *win)</code>
<u>คำอธิบาย</u>	ให้ค่า วรณยุกต์ ณ ตำแหน่งปัจจุบัน ของวินโดว์ win
<u>ผลลัพธ์</u>	ค่าตัวอักษรเป็น int
<u>ฟังก์ชัน</u>	<code>int wmove(WINDOW *win, int y, int x)</code>
<u>คำอธิบาย</u>	เลื่อนตำแหน่ง win->curx, win->cury ให้อยู่ที่ตำแหน่ง x, y
<u>ผลลัพธ์</u>	ผลลัพธ์ค่า 0 เสมอ
<u>ฟังก์ชัน</u>	<code>int wnoutrefresh(WINDOW *win)</code>
<u>คำอธิบาย</u>	ฟังก์ชันนี้จะปรับปรุง curscr ด้วยข้อมูลบนสำหรับแสดงผลของวินโดว์ win
<u>ผลลัพธ์</u>	0=สำเร็จ 1=มีข้อผิดพลาดเนื่องจาก win เป็น NULL pointer
<u>ฟังก์ชัน</u>	<code>int wprintw(WINDOW *win, char *fmt, ...)</code>
<u>คำอธิบาย</u>	พิมพ์ output ตามรูปแบบของ fmt ไปที่วินโดว์ win
<u>ผลลัพธ์</u>	ให้ค่า 0 เสมอ
<u>ฟังก์ชัน</u>	<code>int wrefresh(win)</code>
<u>คำอธิบาย</u>	ฟังก์ชันนี้จะปรับปรุงส่วนที่มีการเปลี่ยนแปลงของวินโดว์ให้มีผลกับ physical screen ของเทอร์มินัล โดยเรียกใช้ฟังก์ชัน wnoutrefresh(win) และ doupdate()
<u>ผลลัพธ์</u>	0=สำเร็จ 1=มีข้อผิดพลาด
<u>ฟังก์ชัน</u>	<code>int wscanw(WINDOW *win, char *fmt, ...)</code>
<u>คำอธิบาย</u>	อ่านตัวอักษรตามรูปแบบ fmt จากวินโดว์ win
<u>ผลลัพธ์</u>	จำนวนตัวอักษรที่มีการอ่านได้
<u>ฟังก์ชัน</u>	<code>int wstandend(WINDOW *win)</code>
<u>คำอธิบาย</u>	ลบ stdout ออกจาก video attribute ของวินโดว์ win
<u>ผลลัพธ์</u>	0 เสมอ
<u>ฟังก์ชัน</u>	<code>int wstandout(WINDOW *win)</code>
<u>คำอธิบาย</u>	ตั้งให้ video attribute ของวินโดว์ win มี attribute stdout
<u>ผลลัพธ์</u>	0 เสมอ