

CHƯƠNG 2 : BÀN PHÍM VÀ CURSOR

§1. CÁC MÃ PHÍM MỞ RỘNG

Chúng ta đã thấy bàn phím tạo các mã thông thường cho các chữ cái, các số và dấu chấm câu. Các phím này đều tạo mã ASCII dài 1 byte. Tuy nhiên có nhiều phím và tổ hợp phím không được biểu diễn bằng bộ kí tự dài một byte này ví dụ như các phím chức năng từ F1 đến F10 hay các phím điều khiển cursor . Các phím này được mô tả bằng một mã dài 2 byte. Byte đầu tiên có trị số là 0 và byte thứ hai là trị số mã của phím này .

1. Nhận biết các mã mở rộng : Một mã mở rộng phải có 2 byte và byte đầu tiên là 0 nên chương trình cần phải đọc 2 byte này . Sau đây là đoạn chương trình nhận biết các mã mở rộng *Chương trình 2-1*:

```
#include <string.h>
main()
{
    char key,key1;
    clrscr();
    while ((key=getche())!='x')
        if (key==0)
        {
            key1=getch();
            printf("%3d%3d",key,key1);
        }
        else
            printf("%3d",key);
}
```

Chương trình này sẽ hiện thị các mã của các phím được gõ cho dù chúng là mã một byte hay 2 byte . Ta dùng hàm getch() để không hiển thị kí tự vừa gõ lên màn hình . Trong biểu thức kiểm tra của while chương trình đọc mã đầu tiên . Nếu mã này là 0 , chương trình biết đó là mã mở rộng và đọc tiếp phần thứ hai của mã bằng hàm getch() . Sau đó nó hiển thị cả hai phần . Nếu phần đầu khác không chương trình sẽ cho rằng đây không phải là mã mở rộng và hiện thị mã này .

2. Đoán nhận mã mở rộng : Một cách đoán nhận mã mở rộng là dùng phát biểu switch như trong chương trình sau :

Chương trình 2-2 :

```
main()
{
    int key,key1;
    clrscr();
    while ((key=getche())!='X')
        if (key==0)
        {
            key1=getch();
            switch (key1)
            {
                case 59 : printf("Phim F1 duoc nhan\n");
                           break;
                case 60 : printf("Phim F2 duoc nhan\n");
            }
        }
}
```

```

        break;
    case 75 : printf("Phim left arrow duoc nhan\n");
        break;
    default : printf("Phim mo rong khac duoc nhan\n");
        break;
    }
}
else
    printf("%3d",key);
getch();
}

```

§2. ĐIỀU KHIỂN CURSOR VÀ ANSI.SYS

1. Khái niệm chung : Tập tin ansi.sys cung cấp tập đã chuẩn hoá các mã điều khiển cursor . ANSI - America National Standards Institut. Để bảo đảm sự cài đặt của tập tin ansi.sys trong tập tin config.sys ta đặt dòng lệnh :

device = ansi.sys

2. Điều khiển cursor bằng ansi.sys : ansi.sys dùng dãy escape để điều khiển con nháy . Chuỗi escape gồm nhiều kí tự đặc biệt . Ansi.sys tìm chuỗi escape này qua thành phần của chuỗi trong hàm printf() và giải mã các lệnh theo sau nó . Chuỗi escape luôn luôn giống nhau , gồm kí tự không in được “\x1B”(là mã của kí tự escape) sau đó là dấu [. Sau chuỗi escape có thể có một hay nhiều kí tự . Nhờ chuỗi này con nháy có thể đi lên , xuống , sang trái , phải hay định vị tại một vị trí nào đó . Ví dụ để di chuyển con nháy xuống dưới ta dùng chuỗi “\x1B[B”

Chương trình 2-3 : Viết chương trình in một chuỗi theo đường chéo :

```

main()
{
    clrscr();
    printf("Cho mot chuoi tan cung bang dau .:");
    while (getche()!='.')
        printf("\x1B[B");
    getch();
}

```

3. Dùng #define và chuỗi escape : Chuỗi “\x1B[B” được mã hoá và rất khó đọc . Khi dùng các chương trình phức tạp nên ghi chú rõ ràng bằng cách dùng dẫn hướng #define .

Chương trình 2-4 :

```

#define c_down "\x1B[B"
main()
{
    while (getche()!='.')
        printf(c_down);
    getch();
}

```

Tóm tắt các lệnh điều khiển con nháy

Mã	Công dụng
----	-----------

"[2J"	Xoá màn hình và đưa con nháy về home
"[K"	Xoá đến cuối dòng
"[A"	Đưa con nháy lên một dòng
"[B"	Đưa con nháy xuống một dòng
"[C"	Đưa con nháy sang phải một cột
"[D"	Đưa con nháy sang trái một cột
"[%d;%df"	Đưa con nháy đến vị trí nào đó
"[s"	Cất giữ vị trí con nháy
"[u"	Khôi phục vị trí con nháy
"[%dA"	Đưa con nháy lên một số dòng
"[%dB"	Đưa con nháy xuống một số dòng
"[%dC"	Đưa con nháy sang phải một số cột
"[%dD"	Đưa con nháy sang trái một dòng và nhiều cột

4. Điều khiển con nháy từ bàn phím : Sau đây là chương trình cho phép bạn vẽ các hình đơn giản trên màn hình

Chương trình 2-5 :

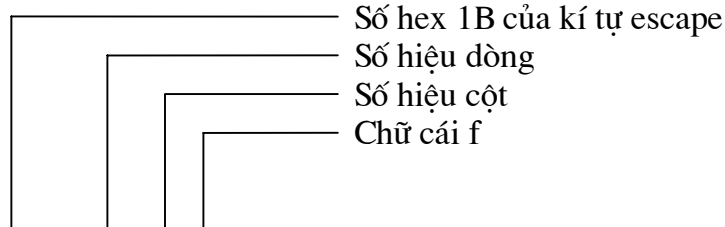
```
#define clear    "\x1B[2J"
#define c_left   "\x1B[D"
#define c_right  "\x1B[C"
#define c_up     "\x1B[A"
#define c_down   "\x1B[B"
#define l_arrow  75
#define r_arrow  77
#define u_arrow  72
#define d_arrow  80
#define across   205
#define updown   186
main()
{
    int key;
    printf(clear);
    while ((key=getch())!=0)
    {
        key=getche();
        switch (key)
        {
            case l_arrow : printf(c_left);
                           putchar(across);
                           break;
            case r_arrow : printf(c_right);
                           putchar(across);
                           break;
            case u_arrow : printf(c_up);
                           putchar(updown);
                           break;
            case d_arrow : printf(c_down);
                           putchar(updown);
                           break;
        }
    }
}
```

```

    printf(c_left);
}
getch();
}

```

5. Đưa con nháy đến vị trí bất kì : Chuỗi escape dạng sau sẽ đưa con nháy đến vị trí bất kì trên màn hình



“\x1B[10;40f”

Sau đây là một chương trình ví dụ về cách dùng chuỗi đó

Chương trình 2-6 :

```

#define true 1
#define clear "\x1B[2J"
#define erase "\x1B[K"
main()
{
    int row=1,col=1;
    printf(clear);
    while(true)
    {
        printf("\x1B[23;1f");
        printf(erase);
        printf("Nhap vao so dong va so cot dang(20,40)");
        scanf("%d%d",&row,&col);
        printf("\x1B[%d;%df",row,col);
        printf("*(%d,%d)",row,col);
    }
}

```

§6. TRÌNH BÀY CHỖ BẤT KÌ TRÊN MÀN HÌNH

Sau đây là chương trình dùng chuỗi định vị cursor .Chương trình cung cấp hai menu định vị dọc theo màn hình .

Chương trình 2-7 :

```

#define size1 5
#define size2 4
#define clear "\x1B[2J"
main()
{
    static char *menu1[]=
    {
        "Open",
        "Close"
        "Save"
        "Print"
        "Quit"
    }
}

```

```

    };
    static char *menu2[]=
    {
        "Cut",
        "Copy",
        "Paste",
        "Reformat"
    };
    void display(char *[],int ,int);

    printf(clear);
    display(menu1,size1,20);
    display(menu2,size2,20);
    getch();
}
void display(char *arr[],int size,int hpos)
{
    int j;
    for (j=0;j<size;j++)
    {
        printf("\x1B[%d",j+1,hpos);
        printf("%s\n",*(arr+j));
    }
}

```

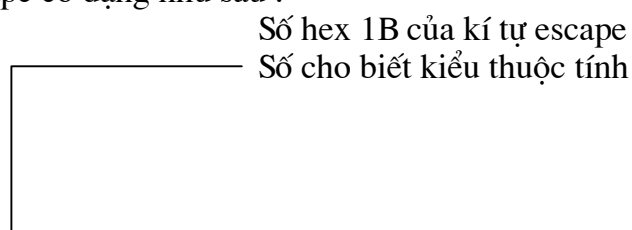
Các mục cho từng menu được cất giữ trong mảng các con trỏ trỏ tới chuỗi . Sau đó chương trình dùng hàm để hiển thị menu . Hàm định vị con nháy nhờ dãy định vị ANSI.SYS , lấy số hiệu dòng từ số hiệu của mục trên menu và số hiệu cột được chương trình chính truyền sang .

§7. CÁC THUỘC TÍNH CỦA KÍ TỰ

Mỗi kí tự hiển thị trên màn hình được cất giữ trong hai byte bộ nhớ . Một byte là mã thông thường của kí tự và byte kia là thuộc tính của nó . Byte thuộc tính ấn định diện mạo của kí tự như chớp nháy , đậm , gạch dưới , đảo màu . Ta có thể dùng chuỗi escape của ANSI để ấn định thuộc tính của kí tự . Theo sau chuỗi kí tự escape và ngoặc vuông là con số và chữ m . Sau đây là danh sách các số tạo hiệu ứng trên màn hình :

- | | |
|-------|---|
| 2,3,6 | màu tối |
| 0 | tắt thuộc tính , thường là màu trắng trên nền đen |
| 1 | đậm |
| 4 | gạch dưới |
| 5 | chớp nháy |
| 7 | đảo màu |
| 8 | không thấy được |

Chuỗi escape có dạng như sau :



“\x 1 B [10 m”

Chuỗi này được gọi trong tiến trình hiển thị . Mỗi khi bật một thuộc tính , tất cả các kí tự sẽ hiển thị theo thuộc tính mới cho đến khi nó tắt đi . Sau đây là chương trình biểu diễn các thuộc tính của kí tự

Chương trình 2-8 :

```
#define NORMAL "\x1B[Om"
#define BOLD "\x1B[1m"
#define UNDER "\x1B[4m"
#define BLINK "\x1B[5m"
#define REVERSE "\x1B[7m"
main()
{
    printf("normal%s blink %s normal \n\n",BLINK,NORMAL);
    printf("normal%s bold %s normal \n\n",BOLD,NORMAL);
    printf("normal%s underline %s normal \n\n",UNDER,NORMAL);
    printf("normal%s reversed %s normal \n\n",REVERSE,NORMAL);
    printf("%s%s reversed and blink %s \n\n",BLINK,REVERSE,NORMAL);
}
```

§8. MENU

Ta xây dựng một chương trình gồm 5 mục menu là Open ,Close,Save,Print,Quit . Các phím mũi tên lên xuống sẽ di chuyển vệt sáng đến các mục cần chọn.Phím INS để chọn và thực hiện công việc tương ứng . Mục Quit sẽ kết thúc chương trình .

Chương trình 2-9 :

```
#define true 1
#define num 5
#define clear "\x1B[2J"
#define erase "\x1B[K"
#define normal "\x1B[Om"
#define reverse "\x1B[7m"
#define home "\x1B[1;1f"
#define bottom "\x1B[20;1f"
#define u_arro 72
#define color "\x1B[4m"
/*#define l_arro 75
#define r_arro 77*/
#define d_arro 80
#define insert 83
main()
{
    static char *item[num]=
    {
        "Open",
        "Close",
        "Save",
        "Print",
        "Quit"
    };
};
```

```

int curpos;
int code;
void display(char *[],int,int);
int getcode(void);
void action(int);
printf(clear);
curpos=0;
while(true)
{
    display(item,num,curpos);
    code=getcode();
    switch (code)
    {
        case u_arro:if (curpos>0)
            --curpos;
            break;
        case d_arro:if (curpos<num-1)
            ++curpos;
            break;
        case insert:action(curpos);
            break;
    }
}

void display(char *arr[],int size,int pos)
{
    int j;
    printf(home);
    for (j=0;j<size;j++)
    {
        if (j==pos)
            printf(reverse);
        printf("%s\n",*(arr+1));
        printf("%s%5s",color,*(arr+j));
        printf(normal);
        printf("%s"," ");
        printf(home);
    }
}

int getcode()
{
    int key;
    while(getch()!=0)
        ;
    return (getch());
}

void action(int pos)

```

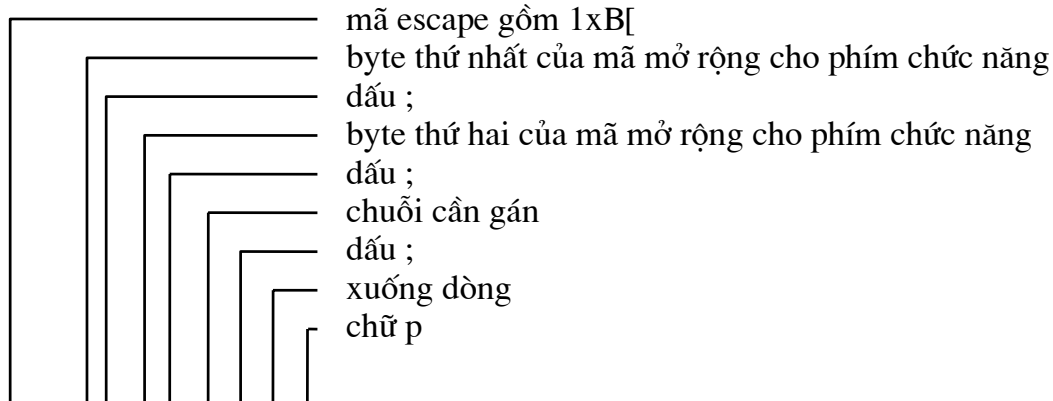
```

{
switch(pos)
{
case 0: printf("Open");
break;
case 1: printf("Close");
break;
case 2: printf("Save");
break;
case 3: printf("Print");
break;
case 4: exit();
}
}

```

§9. GÁN PHÍM CHỨC NĂNG BẰNG ANSI.SYS

Nhờ gán chuỗi vào phím chức năng ta có thể cấu hình lại bàn phím đang dùng .
Dạng thức của chuỗi gán phím chức năng như sau :



\x 1 B [0 ; 68 ; "s" ; 13 p

Chương trình 2-10:

```

main()
{
char str[81];
int key;
clrscr();
printf("Nhap vao mot so cua phim chuc nang :");
gets(str);
key=atoi(str);
printf("Nhap vao mot chuoi de gan phim nay : ");
gets(str);
printf("\x1B[0;%d;\\"%s\\";13p",key+58,str);
}

```