汉字花样显示

实验描述

利用十六点阵汉字库hzk16，编写图形模式下汉字显示程序。要求在普通显示之外，增加多种显示方式，如：文件显示、放大、斜体、倒立、彩色、空心、中英文混搭显示、动态显示等。

实现GB码与ZB的相互转换（字库mhzk）。

设计一种汉字输入法，并编程实现。

原理介绍

区位码的计算

GB2312编码方式规定：对任意一个图形字符都采用两个字节表示，每个字节均采用七位编码表示，其中高字节(第一个字节)称为区码，低字节称为位码，其编码的范围均为0xA1~0xFE。GB2312将编码分成94个区，每个区有94位，01～09区为符号、数字区，16～87区为汉字区(0xB0～0xF7)，10～15区、88～94区是有待进一步标准化的空白区。GB2312将收录的汉字分成两级：第一级是常用汉字计3755个，置于16～55区，按汉语拼音字母 / 笔形顺序排列：第二级汉字是次常用汉字计3008 个，置于56～87 区，按部首 / 笔画顺序排列。

因此区位码的计算方式为：

```
q = (word[0] - 0xA0) & 255;   /*计算区码*/
w = (word[1] - 0xA0) & 255;   /*计算位码*/
```

字库偏移量的计算

当使用HZK16汉字16×16点阵字库存储字模时，每一个汉字需要16×16 = 256 bit = 32 byte 进行存储，因此在国标码GB2312-80中偏移量的计算公式为：

```
ofs = ((q - 1) * 94 + w - 1) * 32 = (q * 94 + w - 95) * 32
```

而在哲标码ZB2014中，前256个字符为ASCII字符，因此其偏移量的计算公式为：

```
ofs = ((q - 1) * 94 + w - 1 + 256) * 32 = (q * 94 + w + 161) * 32
```

点阵字体绘制

普通点阵字体的绘制分16行进行，每行绘制2个字节的数据，对于每个字节，如果该位为1，则绘制星号，否则绘制空格。

```
void normal_print(char *buffer) {
    int i , j;
    for (j = 0; j < 16; j++) {
```

```
        for (i = 0; i < 8; i++) {
            if (buffer[2 * j] & (128 >> i)) cout << "*";
            else cout << " ";
        }
        for (i = 0; i < 8; i++) {
            if (buffer[2 * j + 1] & (128 >> i)) cout << "*";
            else cout << " ";
        }
        cout << endl;
    }
    cout << endl;
}
```

将点阵字体文件输出：

```
void file_print(char *buffer) {
    int i , j;
    ofstream out("out.txt" , ios::out);
    if (out.fail()) {
        cout << "文件输出失败" << endl;
        return;
    }
    for (j = 0; j < 16; j++) {
        for (i = 0; i < 8; i++) {
            if (buffer[2 * j] & (128 >> i)) out << "*";
            else out << " ";
        }
        for (i = 0; i < 8; i++) {
            if (buffer[2 * j + 1] & (128 >> i)) out << "*";
            else out << " ";
        }
        out << endl;
    }
}
```

放大打印点阵字体，即将原来的一行用现在的两行进行输出：

```
void larger_print(char *buffer) {
    int i , j;
    string up , down;  /*up为上面一行，down为下面一行*/
    for (j = 0; j < 16; j++) {
        up.clear();
        down.clear();
        for (i = 0; i < 8; i++) {
            if (buffer[2 * j] & (128 >> i)) {
                up += "**";
                down += "**";
            } else {
                up += "  ";
                down += "  ";
            }
        }
```

```
        for (i = 0; i < 8; i++) {
            if (buffer[2 * j + 1] & (128 >> i)) {
                up += "**";
                down += "**";
            } else {
                up += "  ";
                down += "  ";
            }
        }
        cout << up << endl << down << endl;
    }
    cout << endl;
}
```

倒立打印点阵字体，只需将循环改变一下方向：

```
void down_print(char *buffer) {
    int i , j;
    for (j = 15; j >= 0; j--) {
        for (i = 0; i < 8; i++) {
            if (buffer[2 * j] & (128 >> i)) cout << "*";
            else cout << " ";
        }
        for (i = 0; i < 8; i++) {
            if (buffer[2 * j + 1] & (128 >> i)) cout << "*";
            else cout << " ";
        }
        cout << endl;
    }
    cout << endl;
}
```

彩色字体，设置控制台输出颜色即可：

```
void color_print(char *buffer , Color color) {
    switch (color) {
        case RED:
            SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE) ,
FOREGROUND_INTENSITY | FOREGROUND_RED);
            break;
        case GREEN:
            SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE) ,
FOREGROUND_INTENSITY | FOREGROUND_GREEN);
            break;
        case BLUE:
            SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE) ,
FOREGROUND_INTENSITY | FOREGROUND_BLUE);
            break;
        case YELLOW:
            SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE) ,
FOREGROUND_INTENSITY | FOREGROUND_RED | FOREGROUND_GREEN);
            break;
```

```
        default: {
            cout << "该颜色不存在" << endl;
            return;
        }
    }
    normal_print(buffer);
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE) , FOREGROUND_INTENSITY |
        FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_BLUE);
}
```

斜体字体，每行加上一定的偏移量，如第一行加15个空格，第二行加14个空格，第k行加16-k个空格：

```
void italic_print(char *buffer) {
    int i , j , k = 15;
    for (j = 0; j < 16; j++) {
        for (int p = 0; p <= k; p++) cout << " ";
        k--;  /*每次改变偏移量*/
        for (i = 0; i < 8; i++) {
            if (buffer[2 * j] & (128 >> i)) cout << "*";
            else cout << " ";
        }
        for (i = 0; i < 8; i++) {
            if (buffer[2 * j + 1] & (128 >> i)) cout << "*";
            else cout << " ";
        }
        cout << endl;
    }
    cout << endl;
}
```

国标码转哲标码

在哲标码中，以16位为1字节取代原先的8位为1字节，所有的字符分为“使用字符”和“显示字符”。其中“使用字符”为单字节16位，以15位统一编码(0~0x7FFF)，第16位为0；而显示符号为32位，一般只由专业阅读软件处理。

哲标码的使用字符中，0至0x1FFF包括了原先的ASCII码和其它的字母符号，而0x2000至0x7FFF则存储汉字。

所以将国标码字符串转换成哲标码字符串的思路是，先判断字符串是否是普通ASCII字符串，如果是则直接扩展成16位，否则分别取出其区位码并转换成相应的文字。

```
void gtoz(char* hzk , char* mhzk) {
    int index = 0 , pos = 0;
    int len = strlen(hzk);
    int res;
    while (pos < len) {
        if (hzk[pos] & 0x80 == 0) { /*普通ASCII码*/
            mhzk[index++] = 0;
            mhzk[index++] = hzk[pos++];
        } else {
            int q = (hzk[pos++] - 0xA0) & 0xFF; /*区码*/
            int w = (hzk[pos++] - 0xA0) & 0xFF; /*位码*/
            res = 0x2000 + (q - 1) * 94 + (w - 1);  /*16位ZB码*/
            mhzk[index++] = (res >> 8) & 0xFF;   /*得到高位*/
```

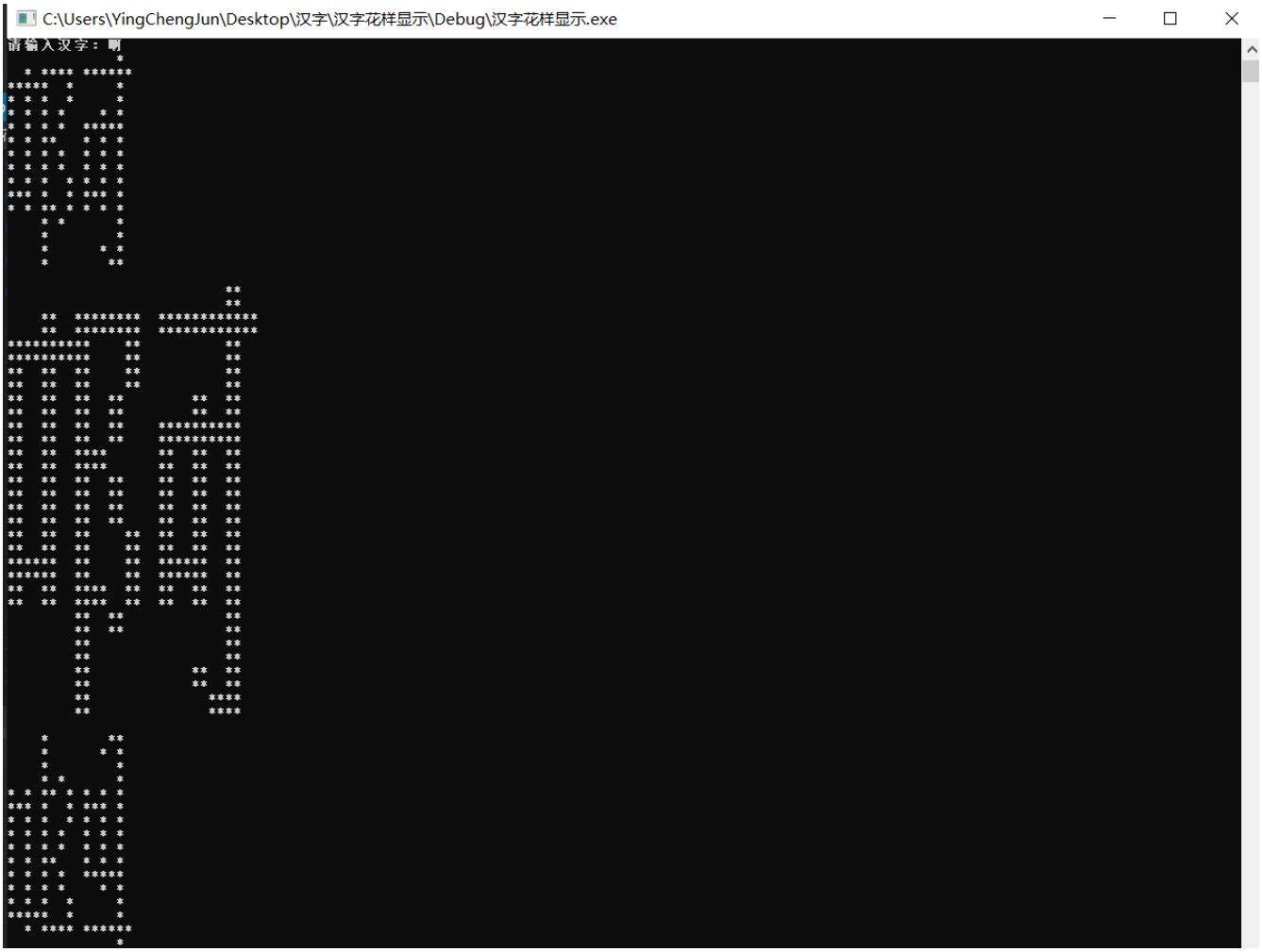
```
            mhzk[index++] = res & 0xFF; /*得到低位*/
        }
    }
    mhzk[index] = '\0';
}
```

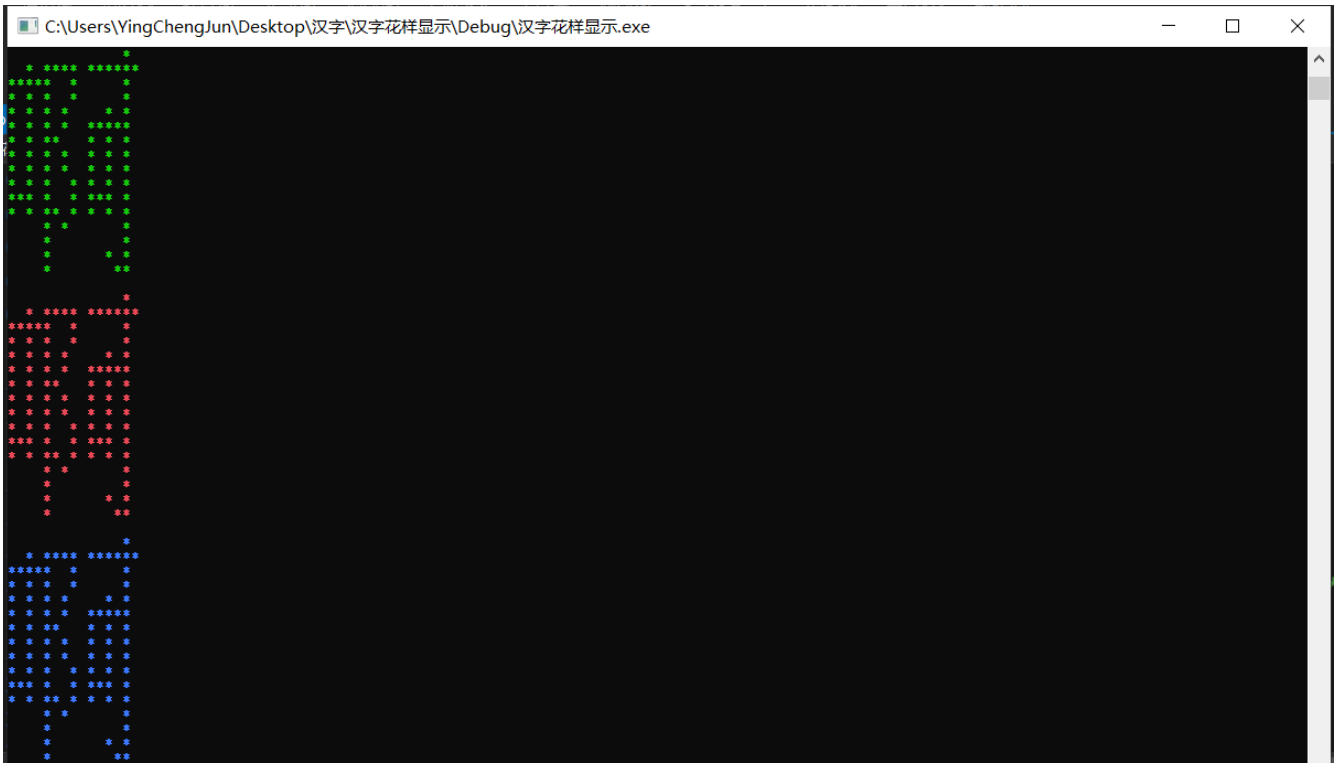
哲标码转国标码

```
void ztog(char* mhzk , char *hzk) {
    int index = 0 , pos = 0;
    int len = strlen(mhzk);
    int res;
    while (pos < len) {
        if (mhzk[pos] == 0) {  /*普通ASCII*/
            hzk[index++] = mhzk[pos + 1];
        } else {
            res = ((mhzk[pos] << 8) + mhzk[pos + 1] - 0x2000) & 0xFFFF; /*得到偏移值*/
            hzk[index++] = (res / 94 + 0xA1) & 0xFF;    /*得到高位字节*/
            hzk[index++] = (res % 94 + 0xA1) & 0xFF;    /*得到低位字节*/
        }
        pos += 2;
    }
    hzk[index] = '\0';
}
```

实验结果

```
/*测试驱动程序*/
int main(void) {
    char word[MAXLENGTH] = { 0 };
    char buffer[MODELSIZE] = { 0 };
    if (init(word , buffer)) {
        normal_print(buffer);   /*正常*/
        file_print(buffer);   /*文件*/
        larger_print(buffer);   /*放大*/
        down_print(buffer);   /*倒立*/
        color_print(buffer , GREEN);   /*彩色*/
        color_print(buffer , RED);
        color_print(buffer , BLUE);
        color_print(buffer , YELLOW);
        italic_print(buffer);   /*斜体*/
    }
}
```


C:\Users\YingChengJun\Desktop\汉字\汉字花样显示\Debug\汉字花样显示.exe
请输入汉字：啊


C:\Users\YingChengJun\Desktop\汉字\汉字花样显示\Debug\汉字花样显示.exe