

,即使是非基本字根的单体结构(注:这些单体结构不一定是汉字),也全部是由基本字根与基本字根或基本字根与单笔划按照一定的关系组成的。这种组成可以按其位置关系将汉字分为四种类型:单、散、连、交结构。掌握汉字的基本结构,对于输入汉字,进行字根拆分就比较容易了。拆字是输入汉字的关键,它有以下三点规律。

1.连笔结构的字:这种结构的汉字在组合时,是由基本字根+单笔划(或弧立点)组成,拆分时一定要拆成单笔和一个基本字根,它不能运用“取大优先”的原则。如:天:应拆成“一、大”,而不能拆成“二、人”;生:应拆成“丨、主”不能拆成“尸、土”。

2.交叉结构或交连混合结构的字:拆字时按书写顺序,运用“取大优先”的序列。如:失:拆成“乚、人”,未:拆成“二、小”。

3.散字根结构的字:拆字时运用取大优先的原则,同时兼顾其直观性。

(1)直观性强的散根结构的字:拆字时主要兼顾直观性。例如:矢应拆成“丨、大”,不能拆成“乚、人”;

(2)直观性不强的字:拆字时运用“取大优先”的原则。如:兰,拆成“艹、二”;气,拆成“乚、乙”。

总之在拆字时要灵活运用“能散不连,兼顾直观,能连不交,取大优先”四句拆字口诀,在拆字过程中做到“散”比“连”优先,“连”比“交”优先,并在可能的几种拆法中字数目少的优先。

三、判断字型 and 末笔识别

五笔字型规定:当一个字不足四码时,要加一个识别码,这是减少重码的有力措施。怎样正确地找出识别码,关键是弄清汉字的字型和取末笔的规则。一般来说,汉字的三种字型中,一型字(左右型)和二型字(上下型)比较容易区分,而三型字比较难以判断字型,要判断一个字是否是三型字(复合型),有以下几个要点。

1.基本字根单独成汉字的虽然属于三型字,但却不需判断字型。因为它们有专门的取码规则:“键名代码十首笔代码十次笔代码十末笔代码。”例如:寸、又、雨、马等字。

2.连笔结构(包括带点结构)的汉字属于三型字。如:天、主、太、产、斗、头等。但要注意,如果单笔划与字根之间有明显间距的字不属于连笔结构。例如:个、旧、么、幻、鱼等。

3.交叉结构的汉字属于三型字。如:农、电、丙、来等。

4.包围型汉字(全包围和半包围)属于三型字。例如:国、区、甸、冈、这等。总之,除了连与交的字外,其余不分上下左右的汉字也都是三型字。

取识别码中末笔的方法有几点规定:

1.包围型字取被包围部分的笔划结构:例如:“因”字的末笔取“、”,“远”的末笔取“L”。

2.汉字的右部含“斜钩、点、撇”的字。如:咸、茂、成、伐、藏等字,书写时按“斜钩、点、撇”的顺序书写,末笔取撇“”。

3.字根“刀、九、力、匕”等右下角为折的字,末笔取折。“注:这几个字根处于末结构时)

四、积累记忆简码

为了提高汉字的录入速度,五笔字型中提供了大量的简码输入方式。简码分为一级简码,二级简码和三级简码。一级简码的实用频率很高,只有 25 个,应熟记住。二级简码和三级简码的数量很多,不需死记硬背,只需在汉字输入中积累记忆。其方法为:两个字根的汉字,就先认为是二级简码,把两个字根输入完后,直接按空格键结束。发现该字是二级简码,就记住;如不是二级简码(即该字调不出)再加识别码。同理,三级简码也采取同样方法输入、记忆。当输入的字多了,简码也就记住了。另外,当识别码不熟悉或字根拆不出时,还可借助于“Z”功能键,有意重码一些字,按相应数字键加以选择即可。

掌握以上几点以后,按汉字的编码规则输入汉字,五笔字型输入技术就入门了。

图形方式下汉字的平滑移动方法

陈永红 穆大明

实现汉字移动的程序并不少见,但效果大都不理想,或有闪烁、或有跳跃。本文介绍了一个用 Turbo C 2.0 实现的、在图形方式下汉字平滑移动显示程序。我们知道,在图形方式下显示汉字比较困难,本文中用的汉字是从 24×24 点阵字库中取出的,所以,这个程序不论在西文或是在中文方式下,汉字均可平滑移动显示。同时也给出了一种在西文状态下显示 24×24 点阵汉字的方法。

实现汉字平滑移动显示的基本原理是先定出汉字在起始和到位时的行、列的位置,然后通过改变显示汉字的

行或列,同时擦除原位置上的汉字,如此循环,直到汉字到位时停止移动显示。同时,通过控制某一值的大小,使汉字移动的速度变快或者变慢。

程序执行后,可以在屏幕上看到“汉字移动”四个字,首先,这四个字从屏幕的左边平滑的向屏幕的右边移动,当到达右边后,这四个字从右边又平滑的移动到左边,当到达左边界时,这四个字从屏幕的上部开始向下部平滑移动,当到达下部后,这四个字从下部又平滑的移动到上部,当到达上部时,这四个字从屏幕的左上角平滑的向屏幕的右下角移动,当到达右下角后,这四个字从右下角又平滑的移动到左上角,当到达左上角后,程序结束。本程序是在长城 386 / 33 上调试通过的。

程序清单如下:

```
#include "graphics.h"
#include "stdlib.h"
#include "stdio.h"
void phz16()
void phz();
FILE * fp;
main(p);
main()
{
    int driver, mode;
    float m, ml;
    int maxx, mayy, size;
    void * buf;
    driver = TTECT;
    mode = 0;
    initgraph(&driver, &mode, "");
    maxx = getmaxx();
    mayy = getmaxy();
    setfillstyle(SOLID_FILL, LIGHTGRAY);
    bar(1, 1, maxx, mayy);
    phz24(1, 125, 1, 1, 4, p);
    size = image_size(1, 80, 120, 200);
    if (size != -1)
        buf = malloc(size);
    if (buf)
    {
        getimage(1, 80, 120, 200, buf);
        for (m = 1; m <= maxx; m = m + 1)
            putimage(m, 80, buf, COPY_PUT);
        for (m = maxx; m >= 1; m = m - 1);
            putimage(m, 80, buf, COPY_PUT);
        setfillstyle(SOLID_FILL, LIGHTGRAY);
        bar(1, 1, maxx, mayy);
        for (m = 1; m <= mayy; m = m + 1);
            putimage(250, m, buf, COPY_PUT);
        for (m = mayy; m >= 1; m = m - 1)
```

```
            putimage(250, m, buf, COPY_PUT);
        setfillstyle(SOLID_FILL, LIGHTGRAY);
        bar(1, 1, maxx, mayy);
        m = 1;
        for (ml = 1; ml <= mayy; ml = ml + 0.75)
        {
            m = m + 1;
            putimage(m, ml, buf, COPY_PUT);
        }
        m = maxx;
        for (ml = mayy; ml >= 1; ml = ml - 0.75)
        {
            m = m - 1;
            putimage(m, ml, buf, COPY_PUT);
        }
    }
    restorecrtmode();
}
```

```
void phz24(int x, int y, int z, int xzoom,
           int yzoom, int color, char * p)
{
    long ll;
    FILE * fp;
    register int i, j, k, i1, i2, j1, j2, k1, k2, ko;
    char ss[72];
    char * s = p;
    if ((fp = fopen("clib24", "rb")) == NULL);
    {
        printf("Can not open the file / n");
        exit(1);
    }
    while (* s)
    {
        ll = (((* s + 113) - 13) * 94 + ((* s + 1) - 119));
        fseek(fp, ll * 72L, 0);
        fread(ss, sizeof(ss), 1, fp);
        for (i = 1; i < 24; i++)
            for (j = 0; j < 3; j++)
                for (k = 0; k < 8; k++)
                {
                    if ((ss[i * 3 + j] > (7 - k)) & 1)
                    {
                        for (kk = 0; kk <= xzoom / 2; kk++)
                            for (kk1 = 0; kk1 <= yzoom / 2; kk1++)
                                putpixel(xzoom * (x + 1) - kk * 0.5,
                                           yzoom * (y + j * 8 + k) - kk1 * 1, color);
                    }
                }
        x = x + 24 + z;
        s += 2;
    }
}
```