目录

一、 软件需求说明书

1.1 市场现有软件参考分析

随着移动网络平台和智能手机等设备的快速发展，相比传统词典和教学用书，具有较完备功能的教学类辅助软件更多地受到学生青睐。

以英语背单词软件为例，如今市面上有很多用户以学生为主的免费英语单词软件，例如百词斩、扇贝、墨墨等。他们具有丰富的英语词库、完备的译文和发音，简洁明了的交互界面，如下图 1 所示。这些软件不仅可以实现单机学习辅助，还可以联网为用户定制学习计划，并提供多样化背单词方法，如下图 2 所示：

图 1. 百词斩界面

(a)看图选词意　(b)听音选词意　(c)拼写单词　(d)看词意选英文

图 2. 百词斩软件多样化学习单词方法一览

本次课程设计以现有较为成熟的英文背单词软件的相应功能和界面设计为参考，通过 VS 软件进行 c 语言编程，利用 Python 得到相应的单词库，实现了单词查询和多样背单词功能程序的编写，完成了简易单机版英文背单词软件的设计。

1.2 软件功能

本次设计所完成的英文简易背单词软件主要包含单词学习，单词查询，单词记忆三项主要功能。单词背诵记忆功能又包含三种不同的记忆方式，因此共有五项功能，软件主菜单上的全部功能如下图 3 所示：

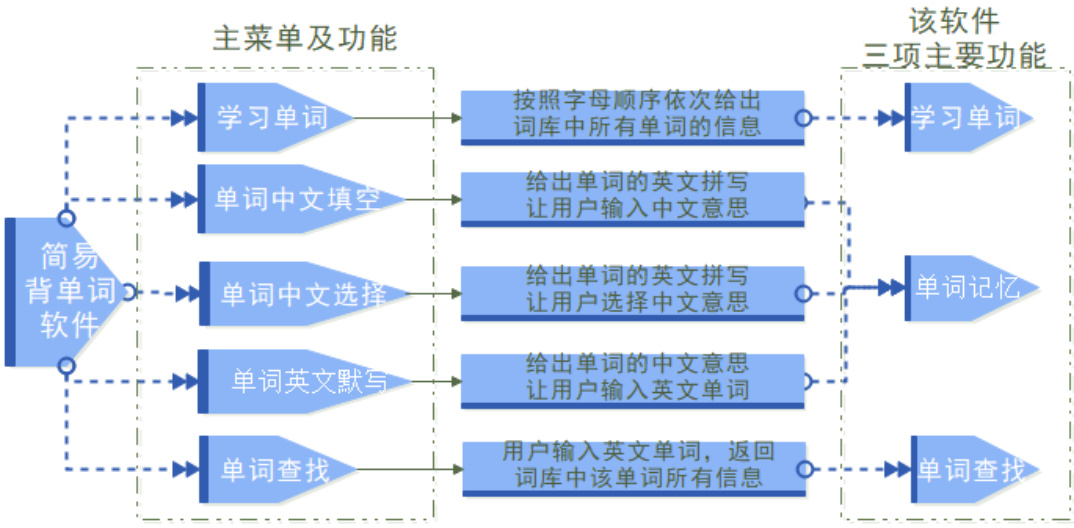


图 3. 软件主菜单功能示意图

- **单词学习：**单词学习功能意在让使用者从字母 a 开始，顺序依次记忆当前单词库中的所有单词。

开始学习后，使用者用键盘按照界面引导输入相应功能数字，即可获取1个单词的英文、汉语翻译、例句和美式发音（随单词同时播放），记忆完毕后再次输入相应功能数字可开始下个单词的记忆。学习完成后可退至主菜单。

- **单词记忆 1**：单词记记忆功能通过借鉴相关软件，利用不完全的单词信息，使用户通过简单交互加深对单词相关内容的记忆。

  单词记忆功能 1 会给出完整英文单词拼写，需用户用键盘输入正确中文释义。用户答案正确则记上 1 分并给出“正确”字样，用户回答错误则返回“错误”字样，同时输出单词的英文、汉语翻译、例句和发音以供用户加深记忆，回答错误不扣分。该模块学习完成后，可得到本次背诵所得分数并退至主菜单。
- **单词记忆 2**：单词记忆功能 2 会给出完整英文单词拼写，需用户在 A,B,C,D 四个选项中选择正确中文释义。每个单词有且仅有一个正确答案，干扰选项是随机的。对用户回答的反馈和积分同功能“单词记忆 1”。
- **单词记忆 3**：单词记忆功能 2 会给出完整中文释义，需用户从键盘正确输入对应的英文单词。对用户回答的反馈和积分规则同“单词记忆 1”。
- **单词查询**：单词查询功能会检索当前词库并输出单词相关信息。进入单词查询功能后，用户输入待查询的英文单词，若当前词库有该单词，则返回其英文、汉语翻译、例句和美式发音；若当前词库中不存在该单词或用户拼写错误，则返回并提示用户“词库中无该单词”。

1.3 软件性能

1） 词库易修改：

部分英语背单词软件的用户希望能定制与自己学习情况匹配的专属词库，应用市场上现有软件基本上均不支持用户自行导入词库。而本次设计只需替换或修改“wordData.txt”文件和“wav”文件夹中的数据，即可实现词库修改。

此外，若存在单词信息不全，例如缺失汉语释义、例句或发音的情况，本次设计中相应的 Python 程序，可以借助网络数据，帮助用户完善词库。

2） 响应时间合理：

本次设计的软件，可在用户做出选择后立刻检索数据反馈相应信息，为用户节省了一定的时间。现有软件出于加深使用者对单词印象的考虑，会在用户做出相应选择或输入后稍作停留，实际上反而降低用户学习效率。

在所有功能中，用户可利用选择是否继续执行该功能的时间，在任意单词处自由决定停留时间，再通过键入相应功能键结束停留，进入下一环节或功能。

快速响应和自由化停留，可提高用户学习效率并保证用户的自由使用。

1.4 非功能性需求

1） 可靠性：

本次设计中单词学习功能是顺序学习，其他记忆单词功能均是乱序，借助生成随机数实现。本次设计在对应部分程序中，加入了检测生成随机数是否重复的代码，可保证学习过程中无重复单词出现以及功能 3 选择中文意思时无重复的干扰选项。保证用户在使用过程中不会遇到小概率事件造成的单词重复、选项重复等情况，软件在使用上具有可靠性。

2） 易用性：

本次设计中程序功能的选择、调用和退出均由用户用键盘输入相应数字实现，并在合适位置加入引导使用的提示语，便于用户操作。此外，任一项功能中用户均可输入-1 退出当前功能，进入主菜单选择其他功能，不必重启程序。

3） 可维护性：

针对用户使用过程中可能出现的问题，我们制作了相应的维护手册，便于用户或测试人员在报错或使用异常时，根据错误信息快速定位错误位置并修正。下方给出了使用中可能出现的部分错误示例，更多情况和详细介绍见文末维护手册。

例如程序无报错，但单词信息出现乱码，可能是由于词库文件 wordData.txt 编码格式非 ANSI 造成。

又例如 Str 堆栈损坏错误可能是由于词库中某一例句过长，而对应结构体成员变量 example 长度过短造成的，需修改该变量的规定长度。

4） 健壮性：

在程序测试中，测试了输入过长字符串、输入空对象、错误指令数字和不符合实际输入类型的输入等极端情况，确定程序存在对空对象的判断，错误功能数字的合理响应，错误类型的合理反馈，保证程序具有健壮性。

5） 可拓展性：

由于本次课程设计时间有限，因此程序所采用的测试词库单词数量较少，仅 200 词，但经过我们的简单测试，该程序可承载较大词库运行，无需做过多调整，运行速度也有所保障。利用本次设计中的 Python 程序，可以联网获得相应的单词释义、发音等数据，便于用户自主设计或拓展词库。

二、 软件概要设计

2.1 程序整体流程图和软件架构

本次设计项目“简易背单词软件”旨在借助当前单词库数据满足用户的基本背单词功能需求，该软件的整体架构如下图所示：

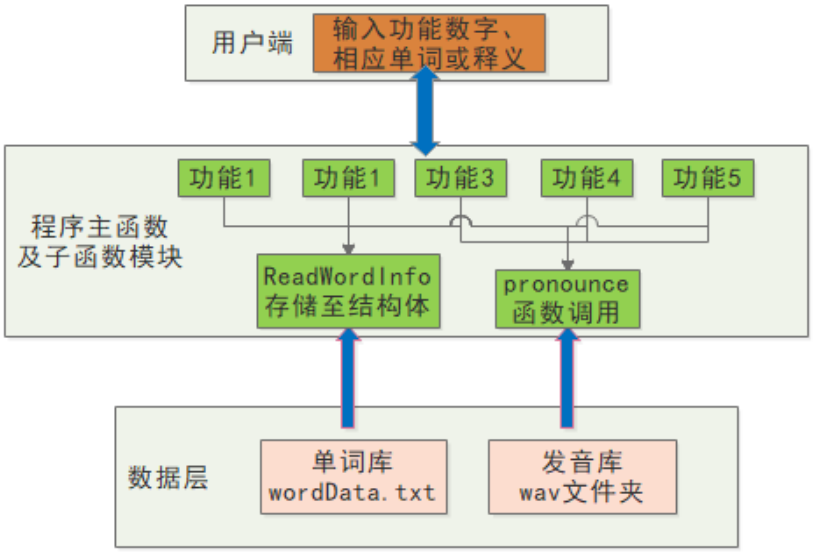


图 4. 简易英文背单词软件整体架构

2.2 生成词库

考虑到实际应用中，用户可能想在软件中导入自己的词库进而背特定范围内的单词，然而大部分用户没有满足要求的现成词库，因此本次设计加入了完善词库的设计。具体的原理和方法见“十、词库生成”。（点击蓝字可跳转至相应位置）

生成词库的程序主要分为三部分，可以分别实现根据用户指定的英文单词得到相应的汉语翻译，得到英式或美式发音，将发音文件统一命名便于程序调用这三项主要功能。很大程度上降低了用户的工作量，使软件更具有普适性。

程序整体流程框图如下所示：

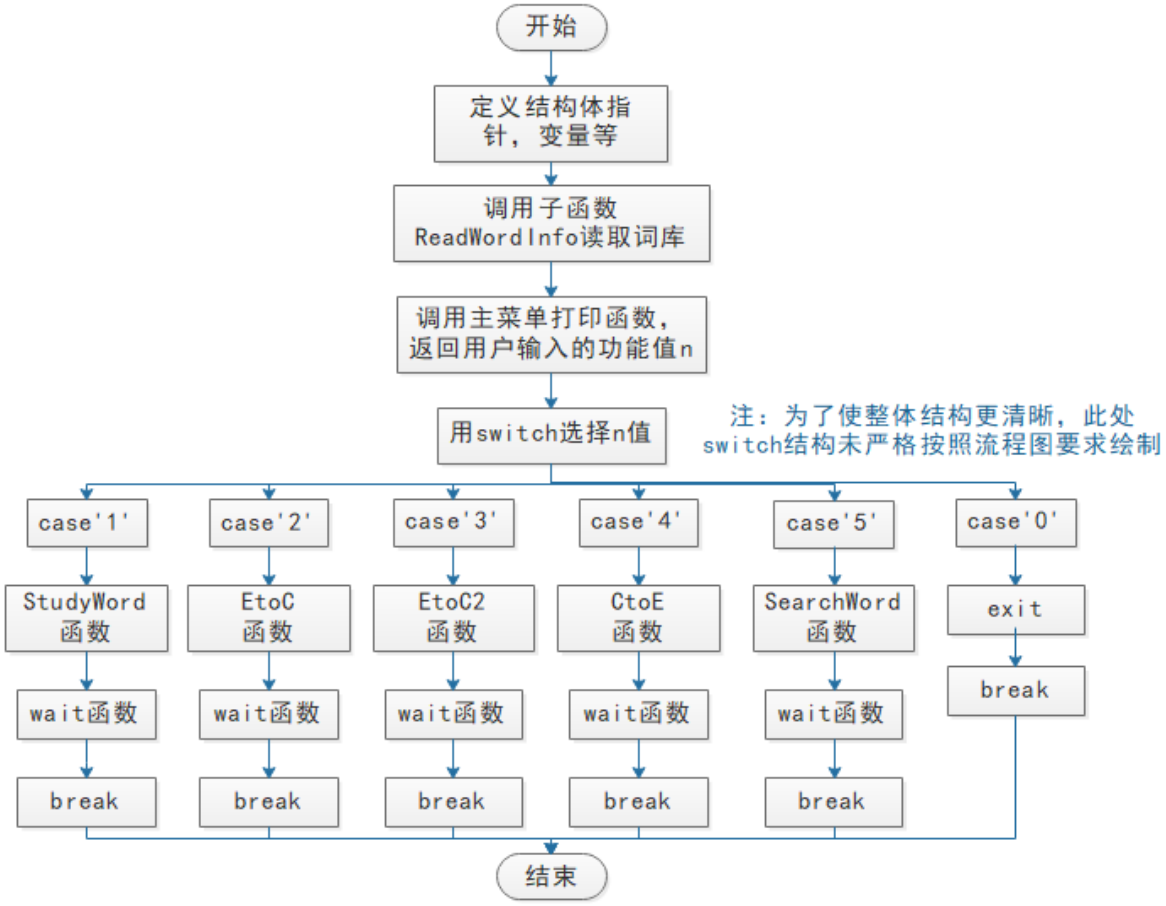


图 5. 简易英文背单词程序整体流程框图

2.2 程序不同模块功能介绍

为实现简易背单词软件主菜单页面、用户交互以及 5 项基本功能，本次设计的主程序包含了较多子函数，每个子函数可视为一个具有特定功能的模块，下表列出了所有子函数的功能、类型和返回值等信息。

表 1. 简易背单词软件各模块子函数简介

子函数名	类型	作用	返回值
menu	int	打印主菜单并获取用户输入的功能数字	输入的功能选择数字
wait	void	使当前界面短暂停留，按任意键继续	
ReadWordInfo	int	读取词库信息并存入结构体	词库中单词总数
StudyWord	void	单词学习，按序输出词库单词供用户学习	
EtoC	void	单词记忆 1，使用户根据英文输入汉语释义	
EtoC2	void	单词记忆 2，使用户根据英文选择汉语释义	
CtoE	void	单词记忆 3，使用户根据汉语写出英文单词	
SearchWord	void	查单词，输入英文单词并输出当前词库释义	
pronounce	void	发音，朗读当前单词在词库中存储的发音	

三、 软件详细设计

上述概要设计中介绍了软件整体设计思路和框架，下面将依次详细介绍各子函数模块的作用、调用值、返回值以及算法描述等。

考虑到词库信息的读取和后续调用，在本次程序中我们借助结构体 word 存储词库中每个单词的所有信息，如下表 2 所示，该结构体成员也会在下述子模块中使用，特此声明。

表 2. 程序结构体及内部成员变量说明表

结构体名称	字段名称	数据类型	数据长度	说明
word	english	char	200	英文单词
	chinese		200	单词中文释义
	example		2000	单词例句

3.1 menu 函数

menu 函数主要作用是输出背单词软件主菜单，并获取用户输入的功能选择数字，对该数字的值加以判断后，将符合要求的功能值作为 menu 函数的返回值返回至主函数。

主菜单包括选择功能的引导词和 5 项基本功能的简单介绍，以及结束程序应输入的数字，主菜单界面如下图所示：

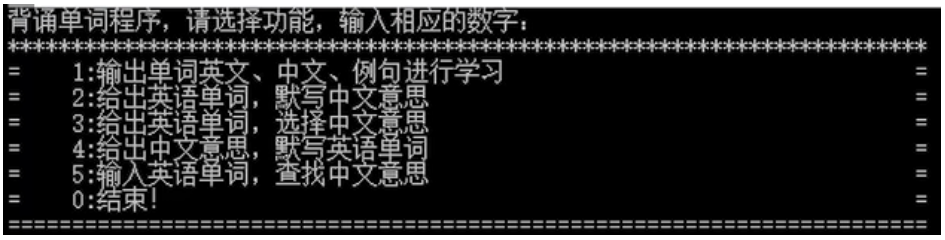


图 6. 主菜单界面

3.2 wait 函数

wait 函数的主要作用是使当前界面短暂停留，按任意键继续，主要用于各功能退出前的短暂停留。该函数运行后会出现下图所示界面，此时用户可以按任意键返回主菜单。

图 7. wait 函数的应用和运行结果

3.3 ReadWordInfo 函数

ReadWordInfo 函数主要作用是读取当前词库中的英文单词、汉语翻译和例句信息并存入 word 结构体变量，便于其他函数调用。

该函数首先需要打开“wordData.txt”文件，若文件加载失败输出提示并返回错误。成功打开文件后将文件加载至缓存区，利用 fgets 函数逐行扫描，若某行开头为大写英文字母(A~Z)或小写英文字母(a~z)则可认为该行为单词信息，再通过判断换行符的个数，确定单词行数进而得到当前词库文件中的单词总数（词库中一个单词信息占据一行）。

得到单词数后，指针定位至单词数据开头，逐行将所有单词英文拼写、中文释义和例句存入相应的结构体成员中。最后关闭文件，将单词数作为该函数返回值。该子模块的流程图如下所示：

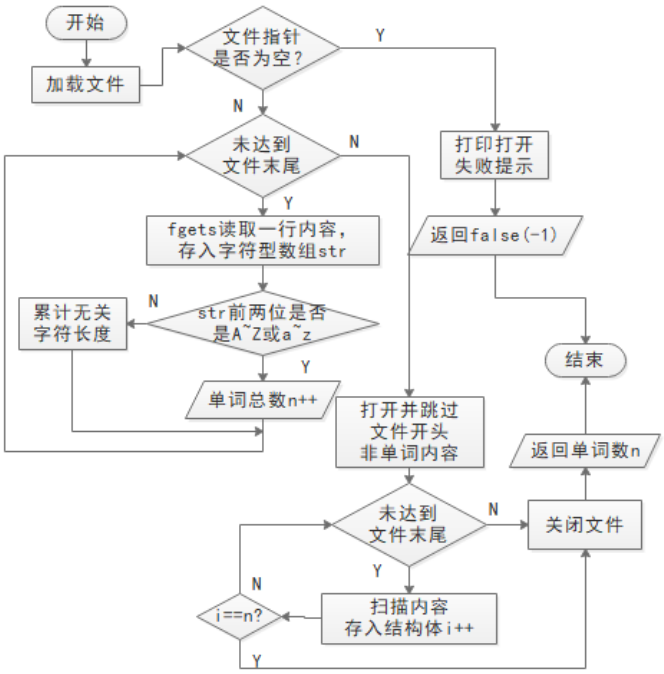


图 8.　ReadWordInfo 模块流程图

3.4　pronounce 函数

wav 文件夹中存储了利用 python 程序得到当前词库所有单词的美式发音，以某单词在词库中的顺序标号为文件名存储，格式为 wav.

pronounce 函数中将每个发音的文件名作为字符串数组 str 的一个元素存入该数组，借助 vector 存放字符串型对象 str，利用 PlaySound 函数播放当前单词发音。

3.5　StudyWord 函数

该函数对应主菜单第 1 项学习功能，依次给出词库中所有单词的英文、中文、例句和发音，在主函数功能 1 被选择时调用该函数。

在打印出“单词信息”字样后，打印词库对应的表头(\n 序号\t 单词\t 中文\t 例句\t\n)，便于用户理解输出各项内容的含义。当用户选择学习一个新单词时（即输入 1），利用循环依次输出词库中每行单词相应的信息，并在信息打印结束后调用上述 pronounce 函数给出当前单词发音。当用户停止学习时（即输入-1），跳出循环并结束该函数。程序流程图如下所示：

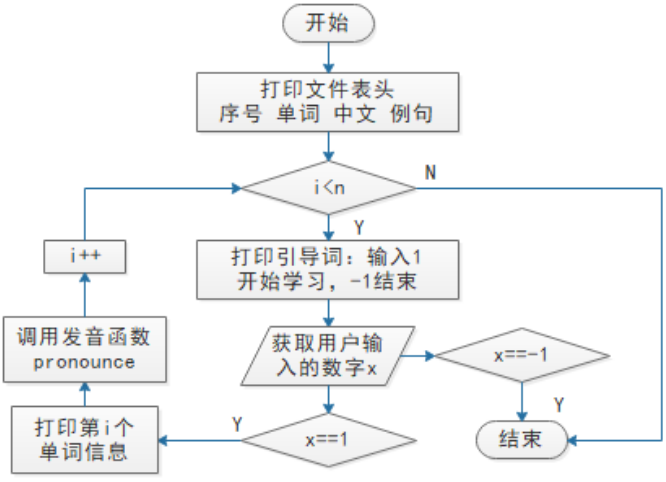


图 9.　StudyWord 模块流程图

3.6 EtoC 函数

该函数对应主菜单第 2 项功能，是背单词功能中的一项，会在界面上给出词库中随机一个单词的英文，要求用户输入正确的中文释义，输入正确得一分，输入错误不扣分并输出正确释义供使用者学习。

首先检测用户输入功能值，为 1 则开始一次学习。开始学习后，利用 rand 函数产生 0~(num-1)以内的随机数（num 为词库单词总数）， 随后通过指针调用并打印单词库中第 i 个单词英文信息，再打印“请输入对应中文”引导用户输入。

利用字符串比较函数 strcmp，判断用户输入字符是否与第 i 个单词的中文释义相同，相同打印 “Congratulations!” 提示用户输入正确，表示计分的变量自增 1；不相同则提示答案错误，并通过指针调用结构体成员中存储的数据，打印正确的单词信息，用 pronounce 函数给出该单词发音。

当检测到用户输入指令数字-1 时，打印用户本次学习得分并退出该模块。

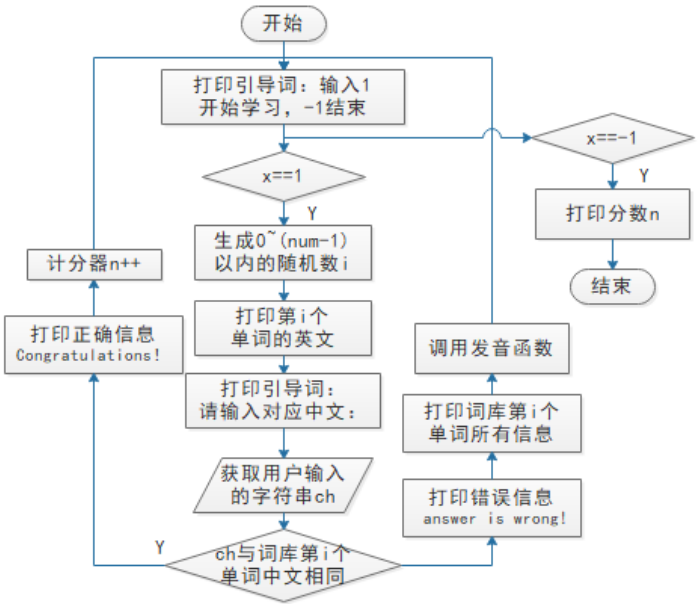


图 9. EtoC 模块流程图

3.7 EtoC2 函数

该模块对应主菜单第 3 项功能，是背单词功能中的第二项。会在界面上给出词库中随机一个单词的英文，要求用户在给出的 A,B,C,D 四个选项中选择正确的中文释义并输入程序中，正确得一分，错误不扣分并输出单词信息供使用者学习。

首先检测用户输入功能值，为 1 则开始一次学习。开始学习后，利用 rand 函数产生 0~(num-1)以内的随机数（num 为词库单词总数），表示本次学习对应的单词为词库中第 i 个单词，将 i 的值放入四维数组 s[]的第一个元素中。

再利用循环，调用 rand 函数生成 0~(num-1)以内的随机数，并赋给 s[]数组的后三个元素，作为干扰选项的单词序号。

每次赋值后检测该值是否与数组中已有的值相同，若相同则舍弃该随机数并再次生成一个随机数作为 s[] 数组同一位置元素的新值，代替被舍弃的值，对于新生成的值也需进行上述检测。该过程可保证后续给出的 4 个选项中有且仅有一个正确答案，且三个干扰选项不相同。

得到存放了 4 个随机数的数组 s[]后，利用冒泡法对该数组中的值从大到小排序，并依次打印 s[] 数组中 4 个数值对应的单词信息。冒泡法排序可以保证本来位于 s[] 数组第一位的随机数 i，即对应正确单词信息的数字，在 s 数组中的位

置不再固定，进而保证正确选项随机出现在 A,B,C,D 四个选项中。

打印“请在下列选项中选出正确选项：”引导用户输入认为正确的选项（A or B or C or D），考虑到 A,B,C,D 中每一项依次对应 s[]数组中的值，因此若某一选项对应 s[]数组值与本次生成的 i 相同，则表示该选项是正确选项。可利用 switch 函数对用户输入的选项进行选择，对于每句 case 语句，判断字符对应的 s[]数组值是否等于 i，是则表示用户回答正确，计分变量自增 1 并输出正确提示；反之表示回答错误，输出错误信息并给出该单词的正确信息、

当检测到用户输入指令数字-1 时，打印用户本次学习得分并退出该模块。

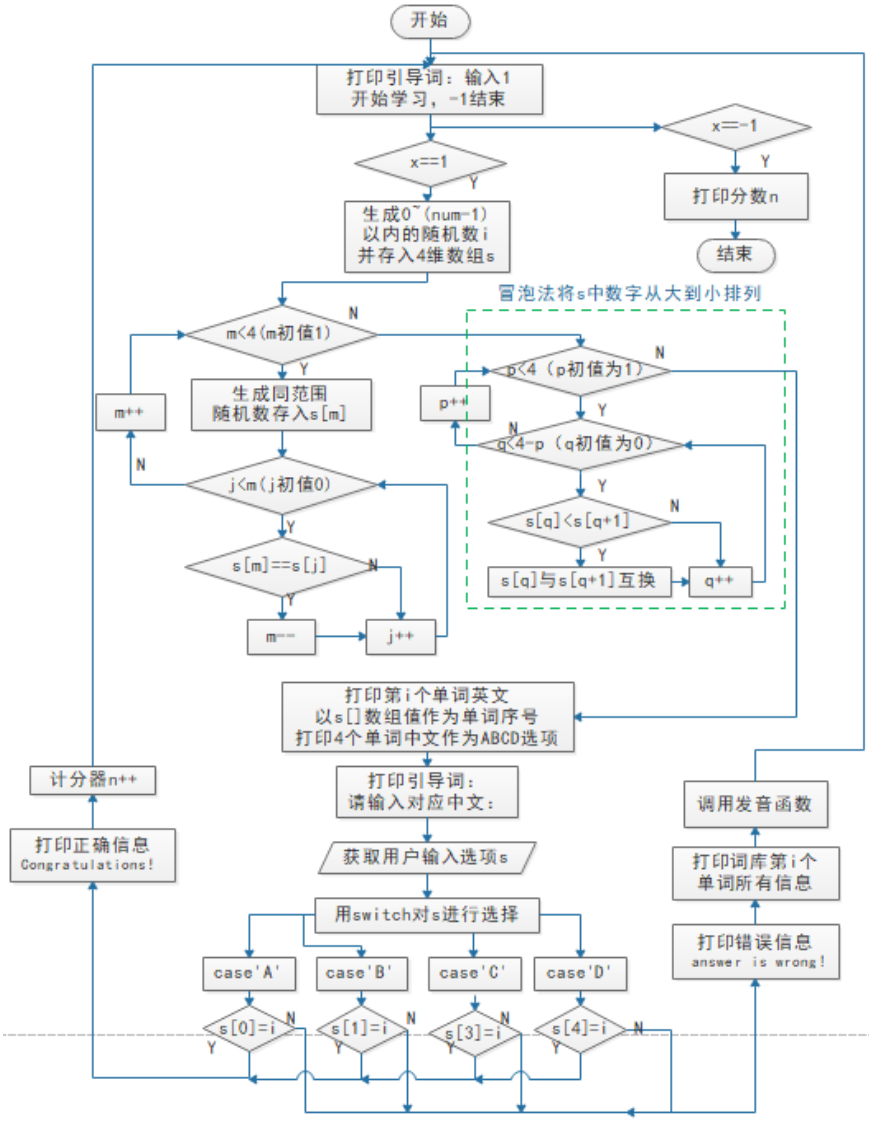


图 10.　EtoC2 模块流程图

3.8　CtoE 函数

该模块对应主菜单第 4 项功能，是背单词功能中第三项，会在界面上给出词库中随机一个单词的中文释义，要求用户输入正确的英文，输入正确得一分，输入错误不扣分并输出该单词信息供使用者学习。

首先检测用户输入功能值，为 1 则开始一次学习。开始学习后，与上述“英译汉”功能相类似，利用 rand 函数产生 0~(num-1)以内的随机数（num 为词库单词总数），　随后打印单词库中第 i 个单词的中文释义，再打印提示语“请输入对

应单词”，引导使用者输入该单词的英文。

同 3.6，通过比较函数 strcmp 判断用户输入的英文是否与第 i 个单词的英文相同，相同打印正确信息，计分结果的变量自增 1；反之提示答案错误，打印正确的单词信息，并调用 pronounce 函数给出发音。

当检测到用户输入指令数字-1 时，打印用户本次学习得分并退出该模块。

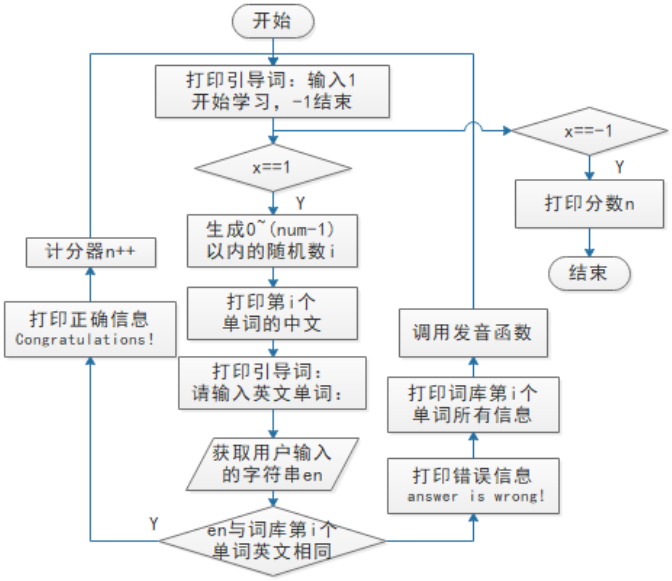


图 11.　CtoE 模块流程图

3.9　SearchWord 函数

该模块对应主菜单第 5 项功能，可查找用户输入的英文单词，并将当前词库中对应的信息输出（包括英文、中文释义、例句和发音）。

首先检测用户输入的功能值，为 1 则开始一次查找。首先打印提示语“请输入您要查找的单词”，引导使用者输入待查找的英文单词。

循环遍历词库中所有单词的英文信息，并利用 strcmp 函数依次将词库中英文单词与用户输入的字符比较，相同则输出该单词所有信息和发音并跳出循环；若遍历词库后未找到相同者，则表示词库中不存在该单词，输出这一提示。

当检测到用户输入指令数字-1 时，退出该模块。该子模块的流程图如下所示：

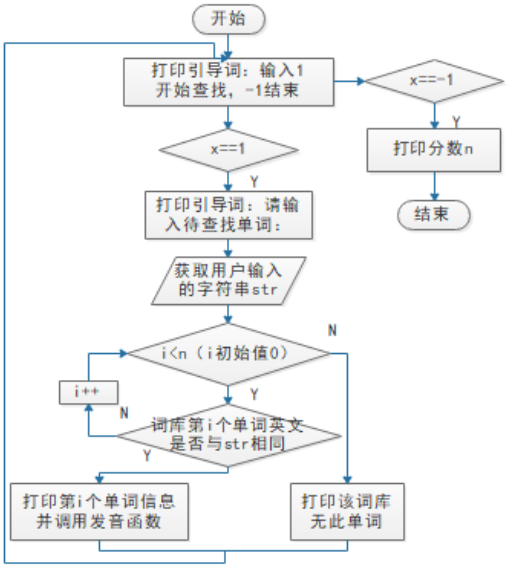


图 12.　SearchWord 模块流程图

四、 软件完整代码

```
#include<stdio.h>
#include<stdlib.h>
#include <string.h>
#include <time.h>
#include <algorithm>
#include<conio.h>
#include<windows.h>
#include<iostream>
#include<string>
#include<vector>
using namespace std;
vector<string>vec[200];//定义个一个字符串容器
#define ture 1
#define false -1
#pragma warning(disable:4996)
#pragma comment(lib, "Winmm.lib")
struct word
{
    char english[200];
    char chinese[200];
    char example[2000];
};

void wait()
{
    int c;
    rewind(stdin);
    printf("Press down any key to continue...\n");
    c = getch();
}

int ReadWordInfo(const char *filename, struct word **pwords)
{
    FILE *filer;
    int n = 0, i = 0, len = 0;
    filer = fopen(filename, "r"); //加载文件至缓存区
    if (filer == nullptr)         //如果文件加载失败，输出提示，并返回错误
    {
        printf("Failed to load file!\n");
        return false;
    }
```

八、 附录清单

1. 功能测试.mp4

采取录屏的方式，测试了简易背单词软件的 5 项基本功能和进入或退出某项功能的完整操作；

为便于老师检验，该视频中除发音外的测试环节已调整为 1.8 倍速播放。

2. wordData.txt

程序中所调用的一个单词库，该 txt 文件中按行存储了高中常用单词 200 个；

包含单词的英文、汉语翻译和例句，每行中的不同内容用空格隔开；

该文件的编码格式为 ANSI.

3. wav 文件夹

该文件夹中存放了 200 个 wav 格式的单词美式发音，文件名序号与单词库中的单词顺序相对照。

4. programmer.cpp

本次程序的完整代码，适用于 Visual Studio 2017 软件和该软件其他较高版本。

该文件中对单词库和语音包的位置做了相应的规定，运行 exe 前请把”wordData.txt” 和” wav”这两个文件放在指定位置。文件路径和要求如下所示：

- wordData.txt 路径：D:\wordData.txt
- “wav”文件夹改名为”1”后，放置路径：D:\1

5. programmer- improved.cpp

优化过后的完整代码，适用于词库内单词数量较大的情况。对 programmer.cpp 程序中的词库检索方法做了改进，优化了随机数查重避免单词重复。相比 programmer.cpp 代码的复杂程度有所提高，但是功能 5 的计算速度和内存占用有所降低。

6. 完整代码.txt

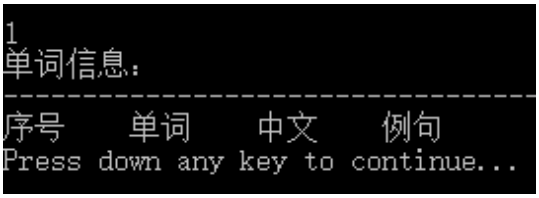
该文件中存储了本次程序的完整代码。

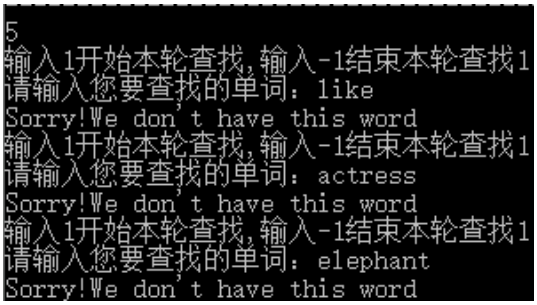
九、 维护手册（用户版）

对于用户使用过程中可能出现的一些异常情况，本次设计特编写维护手册，便于用户快速排除错原因、快速定位错误发生位置并做出相应修改。

异常 1：词库未载入

如果用户在使用过程中选择功能 1 后，随即出现下述图 13(a)所示情况；在选择功能 5 后，即使输入已存入词库的单词，仍出现下述图 13(b)所示情况：





(a) 功能 1 不输出单词便结束　　(b) 功能 5 所有单词查找不到

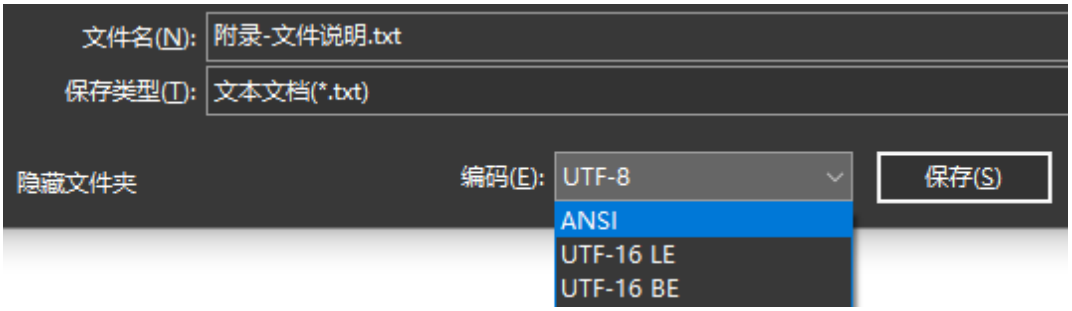
图 13. 异常 1 使用时可能出现的情况

上述情况的出现，均是词库未放在正确的目录下造成的，可参照“附录-文件说明”中相应文件应存放的路径进行文件路径的修改，或更改代码中第 72 行和第 372 行，下述位置的路径为用户所需路径。

```
72    string str[] = { "D:/1/1", "D:/1/2","D:/1/3","D:/1/4","
372   const char *filename = "D:/wordData.txt";
```

异常 2：输出信息乱码

例如程序无报错，但单词信息出现乱码，可能是由于词库文件 wordData.txt 编码格式非 ANSI 造成。可在“文件-另存为”中选择正确的编码格式，如下图所示，或是直接将词库复制到本次程序所附带的 wordData.txt 文件中保存即可。



异常 3：运行弹出异常窗口

Str 堆栈损坏错误可能是由于词库中某一例句过长，而对应结构体成员变量 example 长度过短造成的，需修改代码中该变量的规定长度。

十、 用户词库生成和完善

用户若想使用自己的词库，在此库资料齐全时，只需将文件按照规定方式放置指定目录即可。如果用户自己的词库有待完善，存在缺少英文单词的汉语翻译，缺少相应的发音，发音文件的命名不符合程序要求等情况，用户只需按照下述步骤运行相应的下方程序，即可完善自己的词库。

情况 1：缺少发音文件

有时用户自己拥有较为完整的词库，但却缺少相应的发音文件，下述代码可实现在 project 中开辟两个文件夹，作为英音和美音的本地语音库。同时调用有道词典的在线发音库，首先判断本地语音库中是否有对应的 MP3 文件：如果有就不下载，返回 MP3 绝对路径地址；如果没有，就下载 MP3 到本地，同时返回 MP3 绝对路径地址。

此外，代码中 api 仅有两个参数，即发音类型和单词，在 audio=后面加上相应单词即可，type=0 为美国发音，type=1 为英国发音。

美音：http://dict.youdao.com/dictvoice?type=0&audio=

英音：http://dict.youdao.com/dictvoice?type=1&audio=

例如，输入该地址 http://dict.youdao.com/dictvoice?type=1&audio=act，会得到单词”act”的“英国”发音。

在程序中涉及到的主要功能函数和相应作用如下表所示：

附表 1. 发音下载程序各模块子函数简介

功能函数名	作用
`setAccent(self, type=0)`	调整语音库
`getAccent(self)`	获取什么语音库(英式 or 美式)
`down(self, word)`	下载 MP3 格式的文件
`_getURL(self)`	生成发音的目标 URL
`_getWordMp3FilePath(self, word)`	获取单词的 MP3 本地文件路径，如果有 MP3 文件，返回路径(绝对路径)，如果没有，返回 None

代码：

```
import os
import urllib.request

class youdao():
    def __init__(self, type=0, word='hellow'):
        '''''
        调用 youdao API
        type = 0: 美音
        type = 1: 英音
```

```
        判断当前目录下是否存在两个语音库的目录
        如果不存在，创建
        '''
        word = word.lower()  # 小写
        self._type = type  # 发音方式
        self._word = word  # 单词
         # 文件根目录
        self._dirRoot = os.path.dirname(os.path.abspath(__file__))
        if 0 == self._type:
            self._dirSpeech = os.path.join(self._dirRoot, 'Speech_US')  # 美音库
        else:
            self._dirSpeech = os.path.join(self._dirRoot, 'Speech_EN')  # 英音库
          # 判断是否存在美音库
        if not os.path.exists('Speech_US'):
            # 不存在，就创建
            os.makedirs('Speech_US')
        # 判断是否存在英音库
        if not os.path.exists('Speech_EN'):
            # 不存在，就创建
            os.makedirs('Speech_EN')

    def setAccent(self, type=0):
        '''''
        type = 0: 美音
        type = 1: 英音
        '''
        self._type = type  # 发音方式
         if 0 == self._type:
            self._dirSpeech = os.path.join(self._dirRoot, 'Speech_US')  # 美音库
        else:
            self._dirSpeech = os.path.join(self._dirRoot, 'Speech_EN')  # 英音库

    def getAccent(self):
        '''''
        type = 0: 美音
        type = 1: 英音
        '''
        return self._type

    def down(self, word):
        '''''
        下载单词的 MP3
        判断语音库中是否有对应的 MP3
```

```
53.         如果没有就下载
54.         '''
55.         word = word.lower()  # 小写
56.         tmp = self._getWordMp3FilePath(word)
57.         if tmp is None:
58.             self._getURL()  # 组合 URL
59.             # 调用下载程序，下载到目标文件夹
60.             # print('不存在 %s.mp3 文件\n 将 URL:\n' % word, self._url, '\n 下载
    到:\n', self._filePath)
61.             # 下载到目标地址
62.             urllib.request.urlretrieve(self._url, filename=self._filePath)
63.             print('%s.mp3 下载完成' % self._word)
64.         else:
65.             print('已经存在 %s.mp3, 不需要下载' % self._word)
66.           # 返回声音文件路径
67.         return self._filePath
68.
69.     def _getURL(self):
70.         '''''
71.         私有函数，生成发音的目标 URL
72.         http://dict.youdao.com/dictvoice?type=0&audio=
73.         '''
74.         self._url = r'http://dict.youdao.com/dictvoice?type=' + str(
75.             self._type) + r'&audio=' + self._word
76.
77.     def _getWordMp3FilePath(self, word):
78.         '''''
79.         获取单词的 MP3 本地文件路径
80.         如果有 MP3 文件，返回路径(绝对路径)
81.         如果没有，返回 None
82.         '''
83.         word = word.lower()  # 小写
84.         self._word = word
85.         self._fileName = self._word + '.mp3'
86.         self._filePath = os.path.join(self._dirSpeech, self._fileName)
87.
88.         # 判断是否存在这个 MP3 文件
89.         if os.path.exists(self._filePath):
90.             # 存在这个 mp3
91.             return self._filePath
92.         else:
93.             # 不存在这个 MP3，返回 none
94.             return None
```

```
95.
96.
97. if __name__ == "__main__":
98.
99.     sp = youdao()
100.    sp.down('zoom')
```

情况 2：缺少汉语翻译

如果用户具有词库相应的英文单词，但缺少英文单词对应的汉语翻译，可以通过该部分程序完善用户词库.

下述程序中，在 while 循环中，对表格中的单词进行分别匹配，设置翻译展现形式，对每一行的内容进行逐次翻译，完成后将翻译结果另存为一个与原单词位置相对应的文件。

附表 2. 汉语翻译程序各模块子函数简介

功能函数名	作用
fetch(query_str)	取得 html 数据，传入 parse 函数的参数
parse(html, num)	从第一行的单词开始，逐一翻译成中文

代码：

```
1.  # -*- coding:utf-8 -*-
2.  from openpyxl import load_workbook
3.  from openpyxl import Workbook
4.  import json
5.  import sys
6.  from urllib.parse import urlparse, quote, urlencode, unquote
7.  from urllib.request import urlopen
8.  import re
9.
10. def fetch(query_str):
11.     query = {'q': "".join(query_str)}  # list --> str: "".join(list)
12.     url = 'https://fanyi.youdao.com/openapi.do?keyfrom=11pegasus11&key=2
    73646050&type=data&doctype=json&version=1.1&' + urlencode(
13.         query)
14.     response = urlopen(url, timeout=3)
15.     html = response.read().decode('utf-8')
16.     return html
17.
18.   def parse(html, num):
19.     d = json.loads(html)
20.     try:
21.         if d.get('errorCode') == 0:
22.             explains = d.get('basic').get('explains')
```

```
            result = str(explains).replace('\'', "").replace('[', "").replace(']', "")  # .replace 真好用~
            sheet.cell(row=num, column=2).value = result
            num = num + 1
            for i in explains:
                print(i)
        else:
            print('无法翻译!****')
            sheet.cell(row=num, column=2).value = ' '  # 若无法翻译，则空出来
            num = num + 1
    except:
        print('****翻译出错!')  # 若无法翻译，则空出来
        sheet.cell(row=num, column=2).value = ' '
        num = num + 1

def main():
    Sheet1 = ExcelFile['Sheet1']
    num = 1
    while (1):
        word = Sheet1.cell(row=num + 2, column=1).value
        if (word != None):
            print('正在翻译第', end='')
            print(num, end='')
            print('个单词')
            print(word)
            parse(fetch(word), num)
            num += 1
            print()
        else:
            print('翻译结束! ')
            break
    ExcelFile.close()
    out.save('out.xlsx')

if __name__ == '__main__':
    ExcelFile = load_workbook('D:\word.xlsx')  # 输入文件
    out = Workbook()
    sheet = out.active
    sheet.title = "out"
    main()
```

情况 3：发音文件文件名待修改

背单词软件程序中对发音文件的发音名有相应的要求，发音文件命名应跟词库中单词的顺序标号相同，但实际上通过情况 1 中代码所得到的翻译文件命名是单词本身的英文，文件夹里的文件是按照 a-z 的顺序排列的，逐个修改文件名过于繁琐，通过下述代码对文件夹里的文件进行批量重命名，完成于与主代码中单词结构体发音文件部分的匹配，从而实现单词发音的功能。

程序思路和命名规则：

第一个文件为 1.wav，之后文件的名字数字逐次加一。

代码：

```
import os
import re
import sys
def renameall():
    fileList = os.listdir(r"D:\FFOutput\wav")      #待修改文件夹
    print("修改前："+str(fileList))        #输出文件夹中包含的文件
    currentpath = os.getcwd()      #得到进程当前工作目录
    os.chdir(r"D:\FFOutput\wav")     #将当前工作目录修改为待修改文件夹的位置
    num=1  #名称变量
    for fileName in fileList:      #遍历文件夹中所有文件
        pat=".+\.(wav)"        #匹配文件名正则表达式
        pattern = re.findall(pat,fileName)     #进行匹配
        os.rename(fileName,(str(num)+'.'+pattern[0]))      #文件重新命名
        num = num+1        #改变编号，继续下一项
    print("--------------------------------------------------")
    os.chdir(currentpath)      #改回程序运行前的工作目录
    sys.stdin.flush()      #刷新
    print("修改后："+str(os.listdir(r"D:\FFOutput\wav")))#输出修改后文件夹中包含的文件
renameall()
```

该部分参考文献：

[1] 基础内容: https://www.runoob.com/python/python-object.html
[2] urllib.request 库：
https://docs.python.org/3.7/library/urllib.request.html#module-urllib.request
[3]Source code: Lib/os.py：
https://docs.python.org/3.7/library/os.html?highlight=os#module-os
[4] js 库：https://docs.python.org/3.7/library/json.html?highlight=json#module-json
[5] sys：https://docs.python.org/3.7/library/sys.html?highlight=sys#module-sys
[6] urllib.parse: https://docs.python.org/3.7/library/urllib.parse.html?highlight=urllib
[7] urllib.request:
https://docs.python.org/3.7/library/urllib.request.html?highlight=urllib