《程序设计基础》第五次上机作业

参考答案

1、改写单词（知识点：数组作为函数参数、分支条件判断)

【问题描述】

试编写如下函数

void RegularPluralForm(char word[]);

其中参数 word（在调用该函数时，在主调函数中对应的实参是字符串数组），该函数将遵循标准英语规则将 word 修改为复数形式。规则如下：（1）如果单词以 s, x, z, ch 或 sh 结尾，单词后加 es; (2)如果单词后以 y 结尾，并且前面是一个辅音，将 y 改为 ies; (3)如果单词以 f 或 fe 结尾，则把 f 或 fe 变成 v，再加 es; (4)对于其他单词，后面加 s。

注：以 z 结尾的英文单词变复数大多数是直接加 es，如 buzz, razz, waltz 等，但是也有一些例外，其中有三个需要双写 z 再加 es，它们是 quiz, swiz, whiz，还有单复数形式相同的情况 3 个、不加 es 而是加其它的 2 个。为简单起见，本题仅仅考虑三个需要双写 z 的特殊处理即可。

参考答案如下：

```
#include <iostream>
#include <cstring>
void RegularPluralForm(char word[]) {
    int len = strlen(word);
    if (len == 0) return;
    char lastChar = word[len - 1];
    char secondLastChar = len > 1 ? word[len - 2] : '\0';

    switch (lastChar) {
        case 's':
```

```
            case 'x':
            case 'z':
                strcat(word, "es");
                break;
            case 'h':
                if (secondLastChar == 'c' || secondLastChar == 's') {
                    strcat(word, "es");
                } else {
                    strcat(word, "s");
                }
                break;
            case 'y':
                if (!(secondLastChar == 'a' || secondLastChar == 'e' || secondLastChar
== 'i' || secondLastChar == 'o' || secondLastChar == 'u')) {
                    word[len - 1] = '\0'; // Remove 'y'
                    strcat(word, "ies");
                } else {
                    strcat(word, "s");
                }
                break;
            case 'f':
                word[len - 1] = '\0'; // Remove 'f'
                strcat(word, "ves");
                break;
            case 'e':
                if (secondLastChar == 'f') {
                    word[len - 2] = '\0'; // Remove 'fe'
                    strcat(word, "ves");
                } else {
                    strcat(word, "s");
                }
                break;
            default:
                strcat(word, "s");
                break;
        }
}
```

2、字符串排序 (知识点：数组、字符串、指针、动态内存分配)

【问题描述】

要求输入若干城市的名称（每个城市的名称小于 40 个字符，但是中间可能会有空格，例如，New York, San Francisco 等；城市的总数不超过 30 个），每行输入一个城市名，以一行单独输入一个@作为输入标志。要求对这些城市按照字典顺序进行排序后输出。本题要求用字符数组保存输入的字符串，然后采用冒泡排序、选择排序或其他排序方法实现排序。

程序运行示例:

```
Please input the name of several cities, one city per line ( @ --- end of input) :
Beijing
Shanghai
New York
New Jersey
San Francisco
Atlanta
Seattle
Sorted sequence of the cities:
Atlanta
Beijing
New Jersey
New York
San Francisco
Seattle
Shanghai
```


参考答案如下:

```
#include <iostream>
#include <cstring>
using namespace std;
void bubbleSort(char arr[][40], int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (strcmp(arr[j], arr[j + 1]) > 0) {
                char temp[40];
```

```
                strcpy(temp, arr[j]);
                strcpy(arr[j], arr[j + 1]);
                strcpy(arr[j + 1], temp);
            }
        }
    }
}

int main() {
    char cities[30][40];
    int count = 0;

    cout << "Enter city names (enter '@' to stop):" << endl;
    while (true) {
        cin.getline(cities[count], 40);
        if (strcmp(cities[count], "@") == 0) {
            break;
        }
        count++;
        if (count >= 30) {
            cout << "Maximum number of cities reached." << endl;
            break;
        }
    }

    bubbleSort(cities, count);

    cout << "Cities in alphabetical order:" << endl;
    for (int i = 0; i < count; i++) {
        cout << cities[i] << endl;
    }

    return 0;
}
```


3、整数集合库的实现

【知识点：结构体，动态内存的分配，动态内存扩充、缩小】

【问题描述】

整数集合库的实现：设计一个包含int类型元素的数学集合库，完成cpp文件，实现该整数集合库。数学集合是包含不同对象的集合，例如｛1、3、9｝和｛5、11、11｝= {5, 11}. 根据下面的集合头文件，在相应的cpp文件中写入必要的用于处理集合的函数（创建、删除集合以及添加和删除元素）。

该整数集合库的头文件和测试程序如下。

```
// 集合库头文件
#ifndef USET_H
#define USET_H
#define INITSETSIZE 64 // Initial memory allocated for the set
/* elem: list of elments; card: cardinal of the set; */
struct uset {
   int *elem;
   int card;
};

/* Initialize an empty set and allocate the initial memory: INITSETSIZE*sizeof(int) bytes,
   Which is enabling to hold INITSETSIZE integers.
 */
void newSet(uset **set);

// Free the memory allocated by newSet
void deletSet(uset **set);

/* add elem to the set: check whether it is already in the set;
resize memory if card = allocated memory; new memory = previous+64
e.g. before: mem=128, card=128, after: men=192, card=129 */
void addElem(int *elem, uset *set);

/*【提示】 一旦发现card已经是64的倍数，则加一个元素一定会超出原来的内存空间能力，此时，就应该申请一个新的空间，这个新空间的大小（元素个数）应该是previous+64 （即：先前的元素个数+64）*/

/* remove elem from the set; do nothing if the set does not contain this elem;
resize memory if "too much memory" is used; new = previous-64
e.g. before: mem=192, card=129, after: card=128, mem=128 */
void remElem(int *elem, uset *set);

/*【提示】 一旦发现card-1 是64的倍数，则删除一个元素后，集合中元素的个数与原来申请的内存空间的大小之差，刚好是64，此时有64个元素大小的内存空间未被使用。因此，此时，就可以申请一个较小的新的空间，这个新空间的大小（元素个数）应该是previous-64 （即：
```

```
先前的元素个数-64）*/

// display the number of elements and all the elements
void displaySet(uset *pset);

//* check if the set contains a specific element, if it contains the element return true, otherwise, return false */
bool contains(int *elem, uset *pset);

   /* findPos finds the position (index) of an element in the set, if not exist, return -1 */
int findPos(int *elem, uset *pset);


#endif

// 驱动程序文件
#include <iostream>
#include "uset.h"
using namespace std;
int main()
{
      uset *myset;

      newSet(&myset);

      // 注： 这里将&myset 传入函数newSet，目的是可以修改myset的值
      // 这里&myset 的类型是 uset **

      displaySet(myset);
      for(i=1; i<80;i++)
            addElem(&i, myset);

      displaySet(myset);
      for(i=50; i<100;i++)
            addElem(&i, myset);

      displaySet(myset);
      for(i=30;i<100;i++)
          remElem(&i,myset);

      displaySet(myset);
      deleteSet(&myset);
      return 0;
}
```

参考答案：

```
// uset.cpp
#include <iostream>
#include <vector>
#include "uset.h"

void newSet(uset **set) {
	*set = new uset;
	(*set)->elem = new int[INITSETSIZE];
	(*set)->card = 0;
}

void deleteSet(uset **set) {
	delete[] (*set)->elem;
	delete *set;
	*set = nullptr;
}

void addElem(int *elem, uset *set) {
	if (contains(elem, set)) return;

	if (set->card % INITSETSIZE == 0) {
		int *newElem = new int[set->card + INITSETSIZE];
		for (int i = 0; i < set->card; i++) {
			newElem[i] = set->elem[i];
		}
		delete[] set->elem;
		set->elem = newElem;
	}

	set->elem[set->card] = *elem;
	set->card++;
}

void remElem(int *elem, uset *set) {
	int pos = findPos(elem, set);
	if (pos == -1) return;

	for (int i = pos; i < set->card - 1; i++) {
		set->elem[i] = set->elem[i + 1];
	}
	set->card--;
```

```
    if (set->card % INITSETSIZE == 0 && set->card != 0) {
        int *newElem = new int[set->card];
        for (int i = 0; i < set->card; i++) {
            newElem[i] = set->elem[i];
        }
        delete[] set->elem;
        set->elem = newElem;
    }
}

void displaySet(uset *pset) {
    std::cout << "Number of elements: " << pset->card << std::endl;
    for (int i = 0; i < pset->card; i++) {
        std::cout << pset->elem[i] << " ";
    }
    std::cout << std::endl;
}

bool contains(int *elem, uset *pset) {
    return findPos(elem, pset) != -1;
}

int findPos(int *elem, uset *pset) {
    for (int i = 0; i < pset->card; i++) {
        if (pset->elem[i] == *elem) {
            return i;
        }
    }
    return -1;
}
```