

作业 12

必做题

第一题

源代码:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <malloc.h>

#define CANDIDATE *(pStr+cLocCmp) /*被比较字符串*/
#define NORM *(pStr+cRound) /*参考字符串*/
#define MAX cAmount-1 /*指针数组最大有效下标*/
char sTemp[80]; /*暂时保存输入字符串的字符数组*/
char *pStr[128]; /*存放各个字符串的指针的数组*/

/*输入录入字符串函数*/
char cfInput(void) {
    char cLoc=0; /*表征保存字符串指针的下标*/
    while (1) {
        if (cLoc==128) break; /*超容保护*/
        printf("请输入字符串: ");
        gets(sTemp); /*暂时存储输入字符串*/
        if (sTemp[0]==0) break; /*不再输入即退出*/
        /*为字符串分配适合大小的空间并保存指针*/
        *(pStr+cLoc)=(char*)malloc(strlen(sTemp)+1);
        /*内存是否申请成功判定*/
        if (*(pStr+cLoc)!=NULL) strcpy(*(pStr+cLoc++), sTemp);
        else { /*内存过满申请失败*/
            printf("内存过满, 储存失败。\\n");
            break;
        }
    }
    return cLoc; /*返回指针数组有效指针数量*/
}

/*字符串排序函数*/
int ifCmp(char cLocCmp) {
    if (cLocCmp==0) return 0; /*退出条件*/
    else {
        char cRound=0; /*循环变量*/
        ifCmp(cLocCmp-1);
        for (; cRound<cLocCmp; cRound++) {
            char cCharRound=0; /*内部循环变量*/
            /*判断哪个字符串更大 (strcmp需要const str), 并交换顺序*/
            for (; ((int)cCharRound<=(int)strlen(CANDIDATE) &&
```

```

(int) cCharRound <= (int) strlen(NORM)); cCharRound++) {
    /*后面的字符串更大*/
    if (*(CANDIDATE+cCharRound) > *(NORM+cCharRound)) break;
    /*后面的字符串更小，交换位置*/
    else if (*(CANDIDATE+cCharRound) < *(NORM+cCharRound)) {
        char * pTemp=CANDIDATE;
        CANDIDATE=NORM;
        NORM=pTemp;
        break;
    }
}
}
}
return 1;
}

```

/*主函数*/

```

int main() {
    char cMainRound=0; /*循环变量*/
    char cAmount=cfInput(); /*总字符串数量*/
    ifCmp(MAX);
    for (; cMainRound < cAmount; cMainRound++) {
        printf("%s\n", *(pStr+cMainRound));
        free(*(pStr+cMainRound));
    }
    system("pause");
    return 0;
}

```

运行结果：

```

请输入字符串: hydrogen
请输入字符串: helium
请输入字符串: lithium
请输入字符串: beryllium
请输入字符串: boron
请输入字符串: carbon
请输入字符串: nitrogen
请输入字符串: oxygen
请输入字符串: fluorine
请输入字符串: neon
请输入字符串:
beryllium
boron
carbon
fluorine
helium
hydrogen
lithium
neon
nitrogen
oxygen
请按任意键继续. . . |

```

第二题

源代码:

```
#include <stdio.h>
#include <stdlib.h>
typedef short* atom;
/*打印某个对角线的函数*/
int iPrintDiag (short sMat[20][20], char cSize, char cLoc){
    /*sMat为处理的矩阵（定义得大了些用于后续写任意矩阵）
    cSize为矩阵的大小，cLoc为需要打印的对角线编号*/
    short (*sRow) [20]=sMat;
    int iSum=0;
    for (;(sRow-sMat)<cSize;sRow++){
        atom sColumn=(atom)sRow;
        sColumn+=cLoc+(char)(sRow-sMat);
        if (sColumn>=((atom)sRow+cSize)) sColumn-=cSize;
        iSum+=*(sColumn);
    }
    printf("第%d对角线元素和为%d\n",cLoc,iSum);
    return 0;
}
/*矩阵初始化*/
int iInitMat (short sMat[20][20], char cSize){
    short iOne=0, iTen=0;
    for (;iTen<cSize;iTen++){
        /*每行初始化*/
        for (iOne=0;iOne<cSize;iOne++)
            /*每个元素初始化*/
            sMat[iTen][iOne]=iTen*cSize+iOne;
    }
    return 0;
}
/*主函数*/
int main(){
    short sMat[20][20],sRound=0;
    iInitMat(sMat,10);
    for (;sRound<10;sRound++)
        iPrintDiag(sMat,10,(char)sRound);
    system("pause");
    return 0;
}
```

运行结果:

```
第0对角线元素和为495
第1对角线元素和为495
第2对角线元素和为495
第3对角线元素和为495
第4对角线元素和为495
第5对角线元素和为495
第6对角线元素和为495
第7对角线元素和为495
第8对角线元素和为495
第9对角线元素和为495
请按任意键继续. . . |
```

第三题

源代码:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
char cText[140];/*输入的文字*/
/*打印表情名称函数*/
int iPrint(char *cpLoc, char cEnd) {
    for (cpLoc++;*cpLoc!=cEnd;cpLoc++)
        printf("%c",*cpLoc);
    printf("\n");
    return 0;
}
/*寻找转义符函数*/
int iEsc (char *cpLoc, char cEnd, char cEsc) {
    for (;(cpLoc-cText)>=0;cpLoc--) {
        if (*cpLoc==cEsc) {
            iPrint(cpLoc,cEnd);
            break;
        }
    }
    return 0;
}
/*寻找终止符函数*/
int iEnd (char *cpLoc, char cEnd, char cEsc) {
    for (;*cpLoc!=0;cpLoc++) {
        if (*cpLoc==cEnd) {
            iEsc(cpLoc++, cEnd, cEsc);/*找表情名称*/
            iEnd(cpLoc, cEnd, cEsc);/*找下一个表情名称*/
            break;
        }
    }
}
```

```

        return 0;
    }
    /*主函数*/
    int main() {
        char cEnd, cEsc;
        printf("请输入转义符: ");
        scanf_s("%c", &cEsc);
        getchar();
        printf("请输入终止符: ");
        scanf_s("%c", &cEnd);
        getchar();
        printf("请输入一段半角文字: ");
        gets(cText);
        iEnd(cText, cEnd, cEsc);
        system("pause");
        return -52;
    }

```

运行结果:

```

请输入转义符: [
请输入终止符: ]
请输入一段半角文字: Programming[ is [Happy] very very FUN [Exclaim]!!!
Happy
Exclaim
请按任意键继续. . . |

```

选做题

第一题

源代码:

```

#include <stdio.h>
#include <stdlib.h>
/*静态区变量定义*/
char cOrder; /*阶数*/
short sMat[128][128]; /*系数矩阵*/
short (*pRow)[128];
short *pColumn;
/*函数声明*/
int fiUp(void);
int fiDown(void);
int fiInput(void);
int fiInit(void);
/*右上方向扫描*/
int fiUp () {
    char cFlag=0; /*变为1时跳出循环*/
    printf("%hd ", *pColumn);

```

```

if (pColumn<(*pRow+cOrder-1)) {
    do{
        if (pColumn==(*pRow+cOrder-1)) { /*到右侧*/
            pRow++;
            pColumn+=128;
            fiDown();
            cFlag=1;
        }
        else if (pRow==sMat) { /*到上方*/
            pColumn++;
            fiDown();
            cFlag=1;
        }
        else { /*正常进行*/
            pRow--;
            pColumn-=128;
            printf("%hd ",*(++pColumn));
        }
    } while (cFlag==0);
}
return 0;
}
/*左下方向扫描*/
int fiDown () {
    char cFlag=0; /*变为1时跳出循环*/
    printf("%hd ",*pColumn);
    do{
        if (pRow==sMat+cOrder-1) { /*到下方*/
            pColumn++;
            fiUp();
            cFlag=1;
        }
        else if (pColumn==*pRow) { /*到左侧*/
            pRow++;
            pColumn+=128;
            fiUp();
            cFlag=1;
        }
        else { /*正常进行*/
            pRow++;
            pColumn+=128;
            printf("%hd ",*(--pColumn));
        }
    } while (cFlag==0);
}

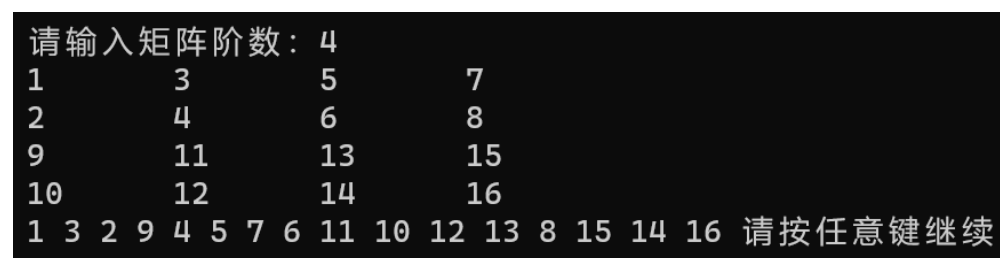
```

```

        return 0;
    }
    /*输入矩阵*/
    int fiInput() {
        for (;pRow<=sMat+cOrder-1;pRow++) {
            for (pColumn=*pRow;pColumn<=*pRow+cOrder-1;pColumn++)
                scanf("%hd", pColumn);
        }
        return 0;
    }
    /*初始化指针*/
    int fiInit() {
        pRow=sMat;
        pColumn=*pRow;
        return 0;
    }
    /*主函数*/
    int main() {
        printf("请输入矩阵阶数: ");
        scanf("%d",&cOrder);
        fiInit();
        fiInput();
        fiInit();
        fiUp();
        system("pause");
        return 0;
    }

```

运行结果截图:



The screenshot shows the program's execution on a black background with white text. It starts with the prompt '请输入矩阵阶数: 4'. Below this, a 4x4 matrix is displayed with values ranging from 1 to 16. At the bottom, the text '请按任意键继续' (Press any key to continue) is shown.

```

请输入矩阵阶数: 4
1      3      5      7
2      4      6      8
9      11     13     15
10     12     14     16
1 3 2 9 4 5 7 6 11 10 12 13 8 15 14 16 请按任意键继续

```