程序设计课程设计

励龙昌

Tel: 13968803107

2023/2/21

课程教学目的

- 教学目标1--能够针对给定的应用程序，根据结构化程序设计的原则与流程，进行需求分析与关键问题梳理，设计合适的数据类型，组织恰当的数据结构，选择合理的算法，进行应用程序的程序结构设计与表达。
- 教学目标2--能够以结构化设计为基础，综合运用C语言语法规则与结构化程序设计的方法，进行应用程序各功能模块的代码设计、测试与集成。

课程教学目的

- 教学目标3--能够对应用程序的设计过程进行总结，撰写实验报告，并合理设计排版总结报告。设计制作汇报PPT，并能清晰表述应用程序的设计过程。
- 教学目标4--了解常用的C++集成开发环境（IDE），能够选择合适的集成开发环境，利用其进行编码、编译和调试，给出可运行的应用程序，并能够进行评价与改进。
- 教学目标5--能够提升自主学习意识，建立程序设计的规范、求精和创新意识，强化工匠精神。

课程设计主要环节

- （1）学生信息管理信息系统：用顺序表和链表完成学生信息系统（20%），
- （2）游戏设计：贪吃蛇（30%）（含过程检查成绩）
- （3）测验：链表应用（20%）
- （4）实验总结报告：游戏设计需求分析、框架设计、程序设计、体会（20%）
- （5）实验汇报：游戏设计结果演示、设计过程汇报（10%）

C语言程序设计（结构化程序设计）的基本原则

1、 模块化程序设计：函数设计

2、 自顶向下：任务分解

3、 逐步求精：对复杂问题，应设计一些子目标作为过渡，逐步细化。

4、顺序、选择和循环结构，限制使用goto语句

复习：指针

```
#include <stdio.h>
int main()
{
        int x;
        int* px;
        px = &x;     //x 与 *px 绑定
        *px = 3;
        printf("x = %d\n", x);
        return 0;
}
```

结构体与指针

```
#include <stdio.h>
int main()
{
        ElemType x;
        ElemType* px;
        px = &x; //x 与 *px 绑定
        (*px).x = 100;
        px->y = 200;
        x.z = 300;
        px->sum = (px->x + px->y + px->z);
        x.ave = px->sum /3;
        printf("%.2f %.2f %.2f %.2f %.2f\n", x.x, px->y, px->z, x.ave, (*px).sum);
        return 0;
}
```

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
        ElemType* px;
        px = (ElemType*)malloc(sizeof(ElemType));
        (*px).x = 100;
        px->y = 200;
        px->z = 300;
        px->sum = (px->x + px->y + px->z);
        px->ave = px->sum /3;
        printf("%.2f %.2f %.2f %.2f %.2f\n", px->x, px->y, px->z, px->ave, (*px).sum);
        return 0;
}
```

不绑定变量

```
Node* initLink()
{
        node *p;
        p = (Node*)malloc(sizeof(node));
        p->pre = NULL;
        p->next = NULL;
        p->data.x = 1000;
        return p;
}
```

指针与形参

```
void swap1( ElemType x, ElemType y)
{
        ElemType temp;
        temp = x, x = y, y = temp;
}
void swap2( ElemType* px, ElemType* py)
{
        ElemType temp;
        temp = *px, *px = *py, *py = temp;
}
void swap3( ElemType* px, ElemType* py)
{
        ElemType* temp;
        temp = px, px = py, py = temp;
}
```

返回指针的函数

```
Node* initLink()
{
        node *p;
        p = (Node*)malloc(sizeof(node));
        p->pre = NULL;
        p->next = NULL;
        p->data.x = 1000;
        return p;
}
```

结构体（自定义数据类型）

```
//定义学生信息
struct ElemType {
        char id[20];   //学号
        char name[20]; //姓名
        char gend[2];   //性别
        double x, y, z;  //三门课成绩
        double ave, sum;
};
typedef struct ElemType elemType;
```

数组

```
struct ElemType data[100];
int n;      //数组长度
```

一、线性表

- 线性表（linear list）：

是数据结构的一种，一个线性表是n个具有相同特性的数据元素的有限序列。（逻辑结构）

一、线性表

- 线性表的相邻元素之间存在着序偶关系。如用（a_1，…，a_{i-1}，a_i，a_{i+1}，…，a_n）表示一个顺序表，则表中a_{i-1}领先于a_i，a_i领先于a_{i+1}，称a_{i-1}是a_i的直接前驱元素，a_{i+1}是a_i的直接后继元素。当i=1,2，…，n-1时，a_i有且仅有一个直接后继，当i=2，3，…，n时，a_i有且仅有一个直接前驱。

二、顺序表

```
struct SeqList   //顺序表
{
        struct ElemType *data;
        int capacity; //顺序表容量
        int length;   //顺序表长度
};
typedef struct SeqList seqList;
struct SeqList L;          seqList* pl;
L.data      pl->data          //数组
L.n         pl-> length       //数组长度
```

二、顺序表

- 顺序表的操作
  - seqList* creatList(int capacity)
  - int findElement(seqList* pl, char id[])
  - int insertList(seqList* pl, elemType x)
  - void printList(seqList* pl)
  - int changeElement(seqList* pl, elemType x)
  - int deleteElement(seqList* pl, char id[])
  - void sortList(seqList *pl)
- 为什么必须用指针？

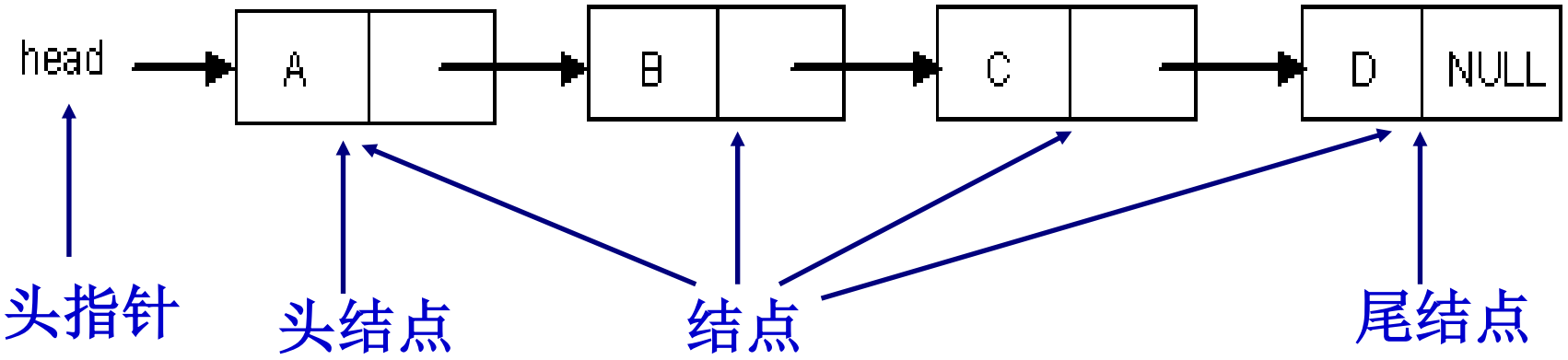
C++中的容器

- vector
- list
- deque

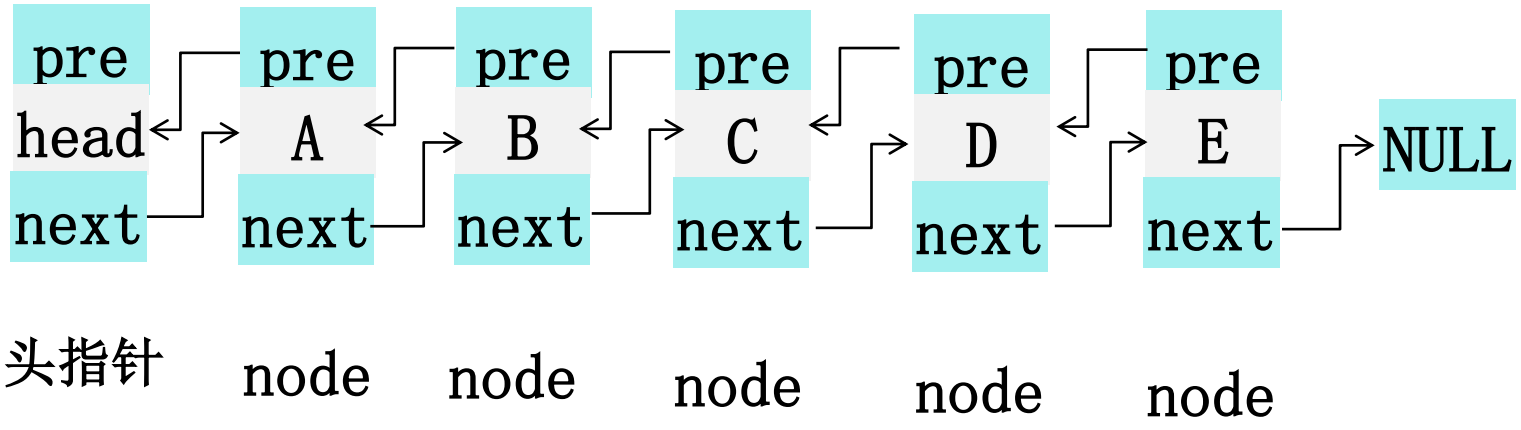
三、链表

- 链式存储：

由若干个同一结构类型的“结点”依次串接而成。单向链表、双向链表、循环链表等



三、链表：双向链表


pre
head
next
pre
A
next
pre
B
next
pre
C
next
pre
D
next
pre
E
next
NULL
头指针
node
node
node
node
node

三、链表

- 1、定义节点：

```
struct Node   //节点
{
        struct ElemType data;   //数据
        struct Node *pre;       //指向前一结点的指针
        struct Node *next;      //指向后一节点的指针
};
typedef struct Node node;
typedef struct Node *linkList;
```

三、链表

- 2、建立节点

```
linkList p;
p = (node*)malloc(sizeof(node));
p->data = x;
p->pre =   ;
p->next = ;
```

三、链表

- 3、头节点、尾节点
  - 申请头节点

    node* head;

    p = (node*)malloc(sizeof(node));
  - 头节点：　没在前驱（一般单独设头节点）

    head->pre = NULL
  - 尾节点：　没有后继

    p->next = NULL
  - 空链表：　head->next = NULL

与数组比较

- 在用数组存放数据时，一般需要事先定义好固定长度的数组，在数组元素个数不确定时，可能会发生浪费内存空间的情况。
- 链表是动态存储分布的数据结构。根据需要动态地开辟内存空间，可以比较自由方便地插入新元素（结点），故使用链表可以节省内存，操作效率高。

- 动态分配相关函数
  - void *malloc(unsigned size)

功能：在内存的动态存贮区中分配一块长度为size的连续空间。

返回值：指针，存放被分配内存的起始地址。若未申请到空间，则返回 NULL(0)。

例如：(int *) malloc(sizeof(int))
(node*) malloc(sizeof(node))

  - void free(void *ptr)

功能：释放由malloc()申请的动态内存空间，ptr存放该空间的首地址。

返回值：无。

例如：free(p);

四、链表操作

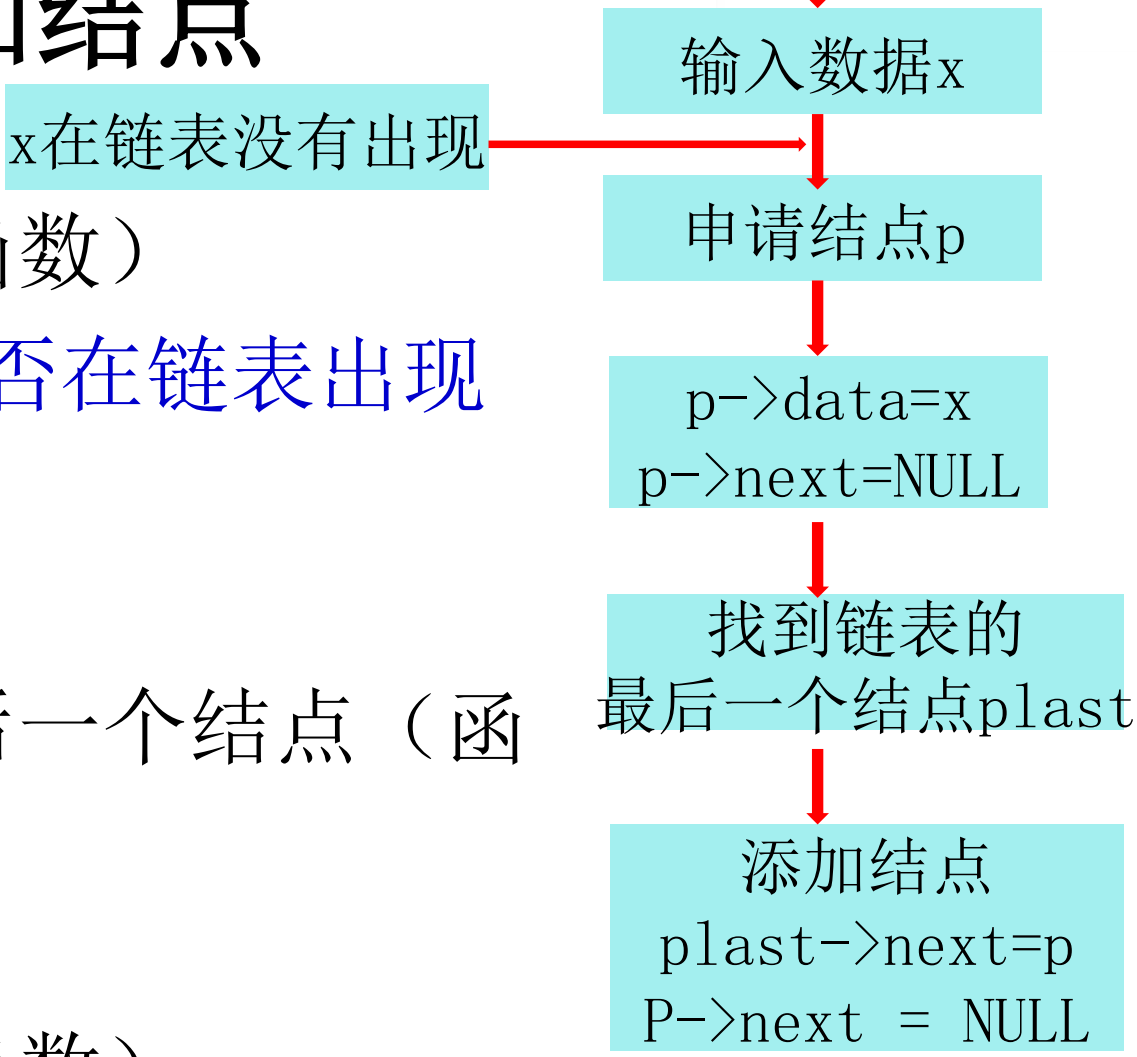
- 1. 链表初始化
- 2. 链表的添加结点
- 3. 链表的遍历、打印
- 4. 插入结点
- 5. 删除结点
- 6. 查找结点
- 7. 修改结点
- 8. 排序
- 9.交换二个elemType的值

1、链表初始化

- 定义一个头结点
- 申请内存空间
- head->next = NULL
- head->pre = NULL
- 链表初始化（函数）

2. 链表添加结点

- 输入数据（函数）
- 查找数据x是否在链表出现（函数）
- 申请空间p
- 找到链表最后一个结点（函数）
- 添加结点
- 输出数据（函数）

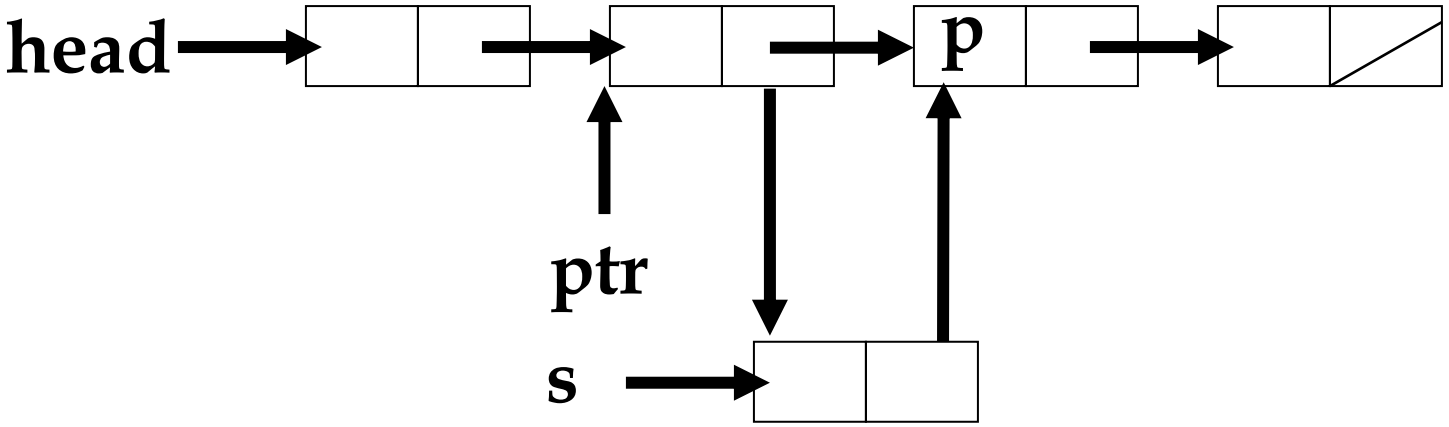


3、链表遍历

```
node *p = head->next;
while( p != NULL)
{
        //程序
        p=p->next;
}
```

4、插入结点

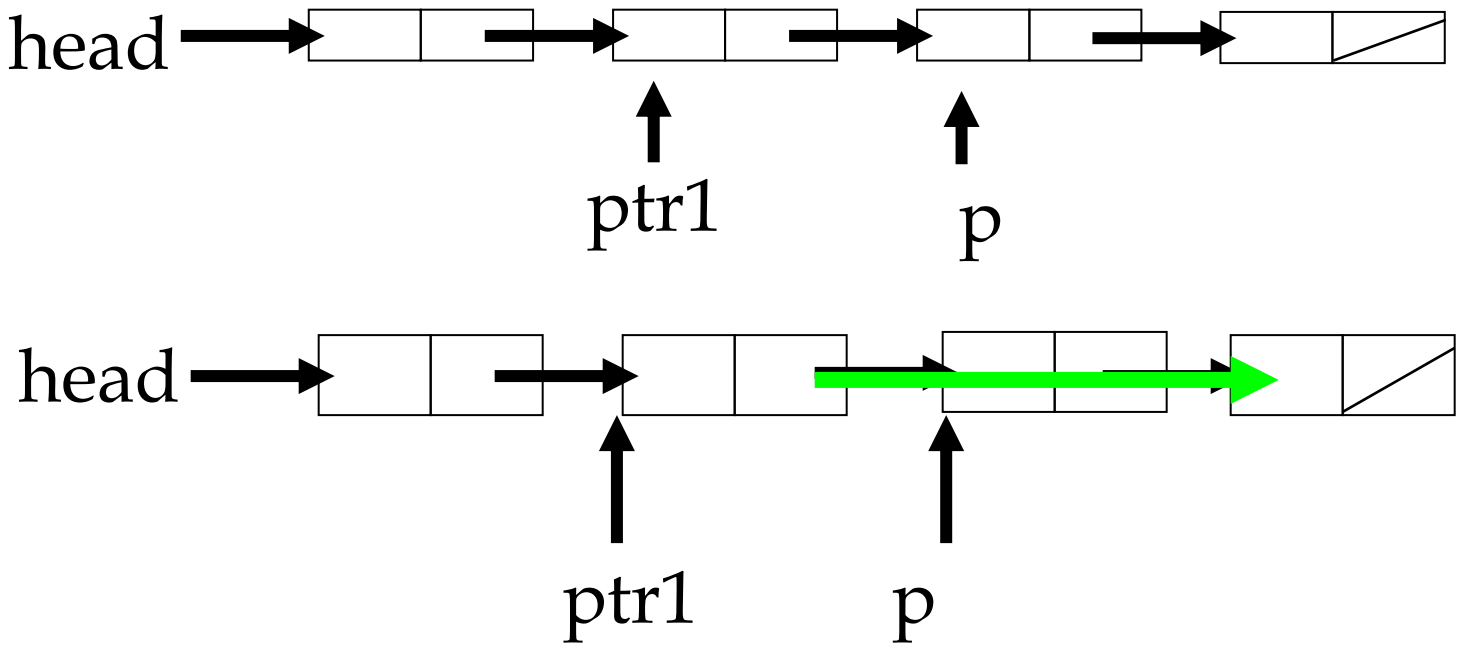
将节点s插入到节点p之前



- 1找：找到节点p的前一节点ptr
- 2连: s->next = ptr->next;
- 3断: ptr->next = s;

- 插入结点s
- 找到插入位置的前一个结点ptr（函数）
- s->next=ptr->next
- ptr->next=s

5、删除结点p



- 1找：找p的前一节点ptr1
- 2接:ptr1->next=p->next;
- 3删:free(p);

⇔ **ptr1->next = ptr1->next->next;**

8、链表排序

- 选择法：用链表形式表示

```
for(i=0;i<n-1; i++)
{
      k=i;
      for(j=i+1;j<n;j++)
      {
            if (a[k]>a[j])    k=j;
      }
      //如果 k!=i, 交换a[k]与a[i]的elemType数据项
}
```

9、完成相关题目

温州大學 WENZHOU UNIVERSITY

项目1-1：学生信息管理系统

题号	标题	通过率
0533	动态链表：插入	0% (0 /0)
2080	动态链表：修改	0% (0 /0)
2079	动态链表：查找	0% (0 /0)
2077	命令式菜单设计	100% (1 /1)
0538	动态链表：排序	100% (1 /1)
0537	动态链表：删除	0% (0 /0)
0534	动态链表：输出	0% (0 /0)
3324	输入输出一个学生信息	50% (1 /2)

程序框架

```
/*
主函数项目：学生信息管理系统
要求：
1、数据结构必须用链表
2、尽可能用函数
*/
int main()
{
        //1、链表或顺序表初始化

        while(1)
        {
                //2、读入命令字符串
                //3、如果 插入
                if (strcmp(c, "Insert") == 0)
                {
                        //3.1、输入学生信息
                        inputElement(&x);
                        //3.2如果成功插入信息（函数），则输出学生信息，
                        //否则，输出 "Failed\n"
                }
```

要求：必须编写以下函数

（1）输入一个学生信息函数

（2）输出一个学生信息函数

（3）交换二个节点的 elemType的数据

（4）链表初始化函数

（5）查链表中按学号查找函数:顺序表中按学号查找,如果找到了返回该结点的指针，没找到，则返回NULL

（6）输出链表中所有元素的函数

（7）删除链表中指定节点的函数

（8）按学号排序的函数

（9）main函数

要求：函数说明

/*

功能：

形参：

返回值：

编写者：

日期：

版本：

*/

四、windows应用程序设计

1、windows应用程序框架

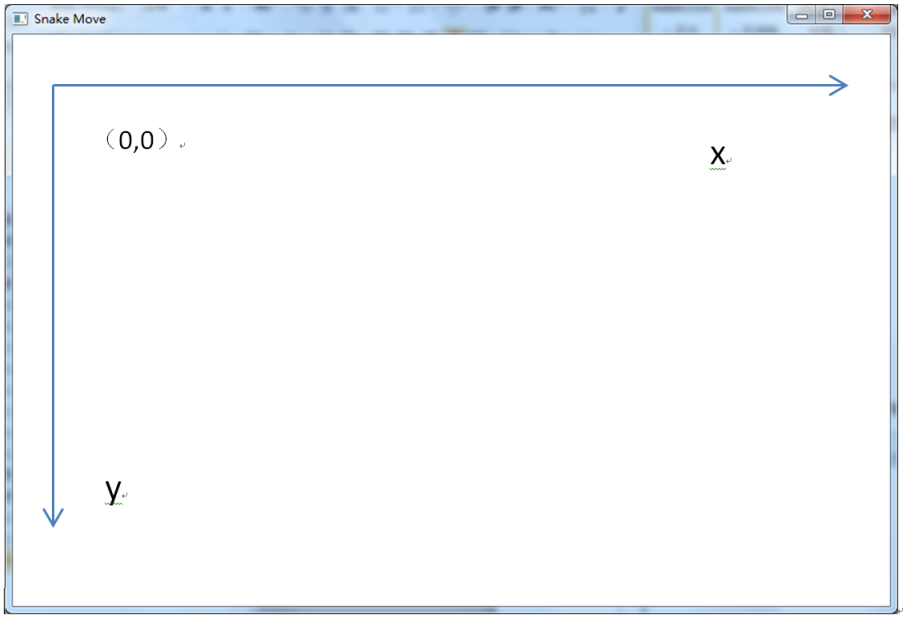
（1）主程序 WinMain ()

（2）消息处理

四、windows应用程序设计

1、windows应用程序框架

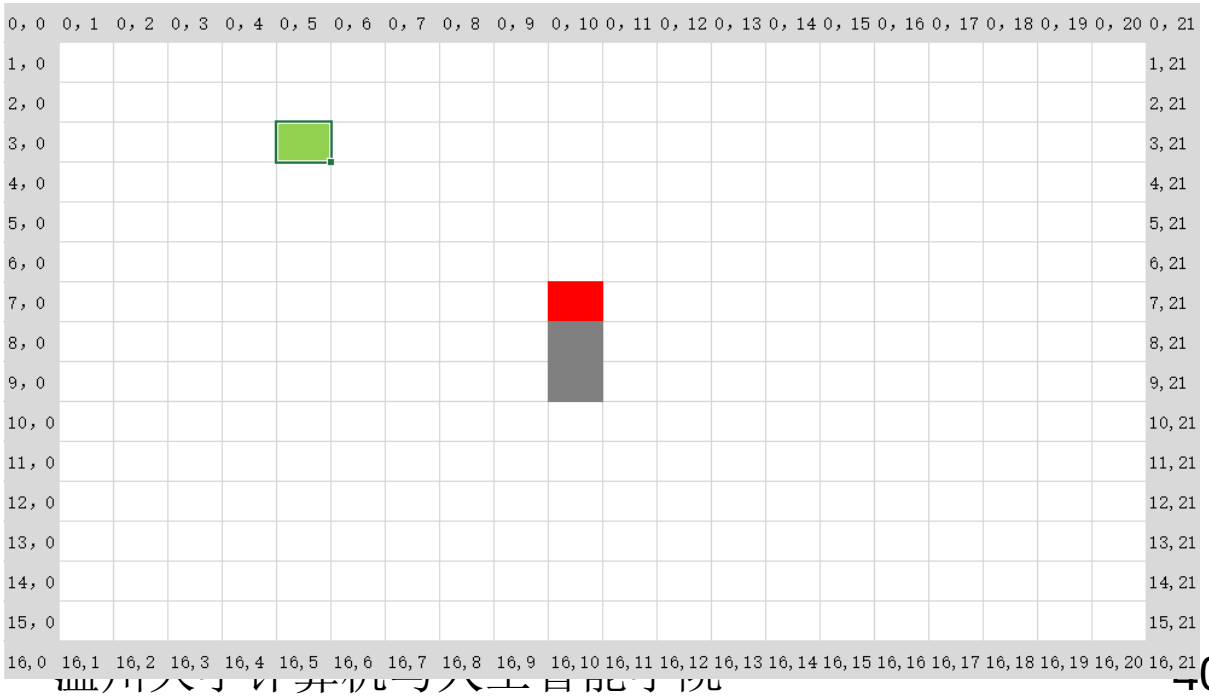
（3）windows坐标系统



四、windows应用程序设计

2、windows应用程序设计

（1）view：用户界面（背景、食物、蛇）

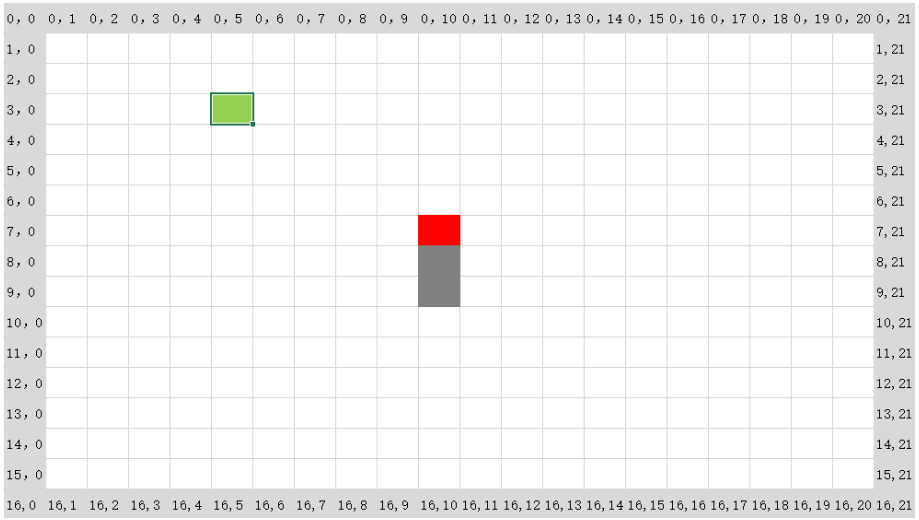


四、windows应用程序设计

2、windows应用程序设计

（2）数据模型：业务规则（Model）

背景、食物、蛇



四、windows应用程序设计

2、windows应用程序设计

（3）程序控制（Controller）：

开始、方向控制、定时器、程序结束

（4）动画：原理

（5）程序流程：吃食物、食物产生、等级变化、速度变化

四、windows应用程序设计

2、windows应用程序设计

（6）定时器：每间隔一定时间，重复执行一段程序（或函数）

（7）全局变量、宏定义

四、windows应用程序设计

2、windows应用程序设计

（8）封装的函数

- 颜色RGB(r, g, b)
- 画矩形函数
- 输出文本
- 输出数字
- 输出图形

四、windows游戏设计

2、windows应用程序设计

（9）文件

- winapp.cpp为应用程序入口，包含winapp.h、myfile.h二个头文件。在消息处理中编写代码，如要调用函数，则将函数写在myfile.h 中。
- winapp.h：（不要修改该文件）封装了相关的应用函数，具体参看函数说明

四、windows应用程序设计

2、windows应用程序设计

（9）文件

- myfile.h自己在此编写相关函数
- 按回车健开始执行程序

（10）数据

20行、30列

map1.txt

四、windows应用程序设计

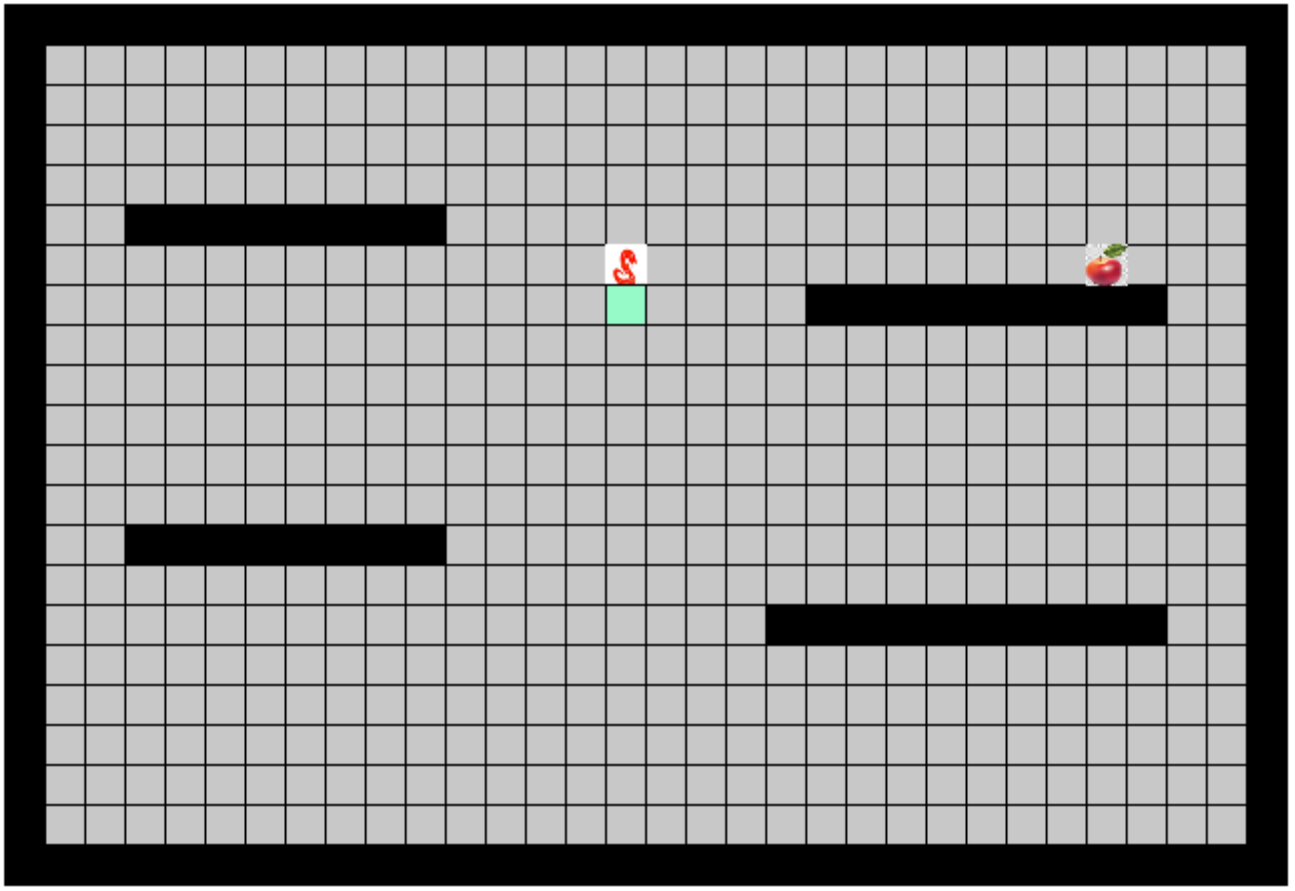
3、老鼠走迷宫

完成示例程序功能

四

按回车键开始　　当前分数　0　　游戏暂停，按空格键继续
按方向键控制方向并移　　等级　1　　按 ESC 暂停

4、



四、windows游戏设计

5、visual studio 2019程

（一）解压压缩文件至

（二）打开项目

（1）启动VS2019

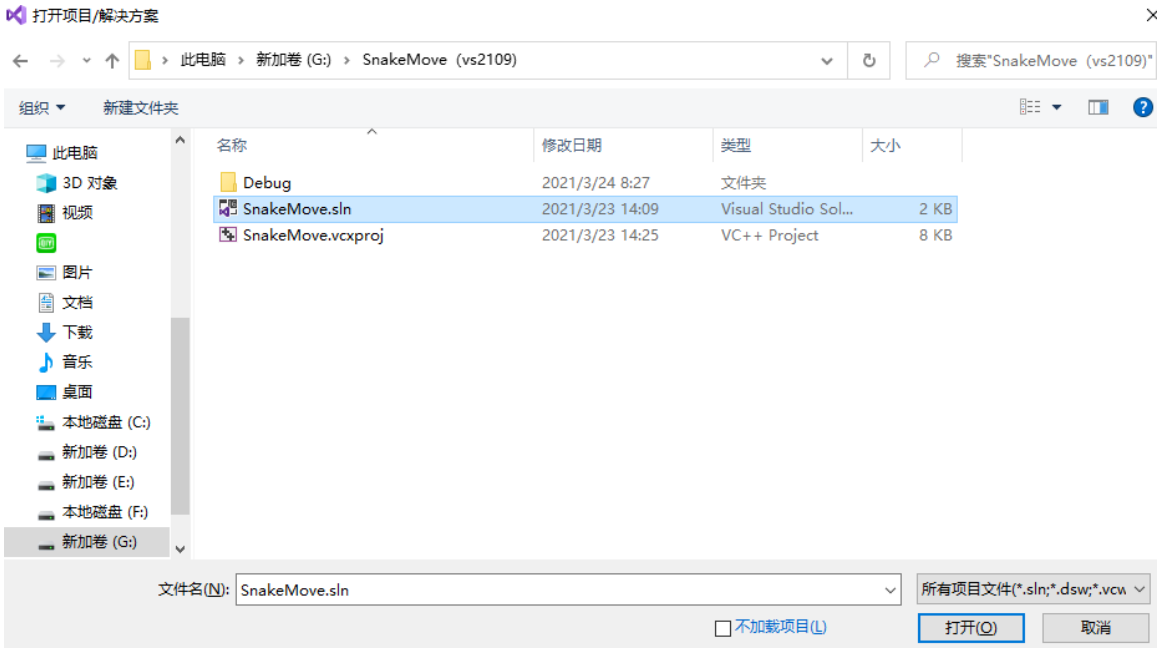
（2）选择“打开项目或

（3）引导至指定文件夹

四、windows游戏设计

5、visual studio 2019程序框架

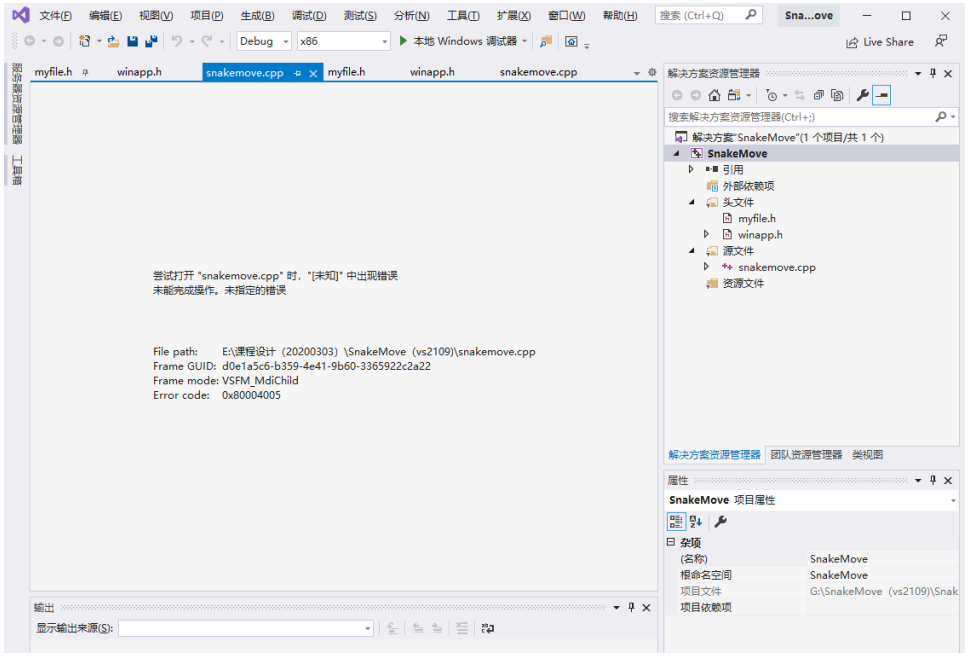
（3）引导至指定文件夹，打开



四、windows游戏设计

5、visual studio 2019程序框架

（4）可能会出现如图提示，但不影响使用



四、windows游戏设计

5、visual studio 2019程序框架

（5）调试→开始执行

四、windows应用程序设计

6、要求

- 理解软件设计的过程：需求分析、总体设计、详细设计、调试
- 理解程序框架：理解MVC（Model View Controller）结构
- 老鼠走迷宫可以用顺序表，贪吃蛇必须用链表设计数据结构。

四、windows应用程序设计

6、要求

- 完成指定游戏程序设计。
- 程序编写规范（有适当的注释、对齐和缩进、变量和函数命名符合规则）
- 课程设计日志：评分时的依据（每星期检查）

课程设计参考标准（1）：学生信息管理系统

- AC
- 1）、输入一个学生信息函数
- 2）、输出一个学生信息函数
- 3）、交换二个 elemType的数据
- 4）、链表初始化函数

课程设计参考标准（1）：学生信息管理系统

- 5）、按学号查找函数:链表中按学号查找，如果找到了返回该结点的指针，没找到，则返回NULL
- 6）、输出链表中所有元素的函数
- 7）、删除链表中指定节点的函数
- 8）、按学号排序的函数
- 9）、程序编写规范（有适当的注释、对齐和缩进、变量和函数命名符合规则）

命名规则

- 一、匈牙利命名法：`void jiedian(List* plist,student s)`
- 二、驼峰命名法：
  - 使用大小写字母来构成标识符的名字。其中第一个单词首字母小写，余下的单词首字母大写。
    例如：printEmployeePaychecks();
- 三、帕斯卡（Pascal）命名法
- 四、下划线命名法。
- https://zhidao.baidu.com/question/4557268731544842 05.html

课程设计参考标准（2）：应用程序设计

- 1）、程序能正确运行
- 2）、实现全部游戏功能
- 3）、界面设计合理
- 4）、用链表组织数据结构
- 5）、程序编写规范（有适当的注释、对齐和缩进、变量和函数命名符合规则）

课程设计参考标准（3）：总结报告

- 1）、排版合理：文档结构
- 2）、全面完整的总结了软件设计的过程，设计中数据模型、数据显示、程序控制的相互关系，及相关函数分析设计。

  可重复性。
- 3、附录：代码
  - 学生信息管理系统（排序）
  - 贪吃蛇代码

学生信息管理系统（排序）代码

- 进入到ACM系统
- 定位学生信息管理系统排序
- 我的提交
- 生成实验报告（pdf）格式或其他形式
- 转换成图片或其他形式插入到总结报告

课程设计参考标准（4）：工作日志

- 1、学生信息管理系统（至少一次）
- 2、贪吃蛇（至少2 次）
- 3、对于已解决的问题，要有深入的理解，到时做为答辩的依据

课程设计提交材料清单

- 1、课程设计总结报告
- 2、汇报PPT
- 3、贪吃蛇的二个文件

  snakeMove.cpp（可选）

  myfile.h
- 4、以文件夹的形式提交，以“学号+姓名”命名文件夹

点评三份总结报告

测验安排

- 时间：
- 地点：