

重庆邮电大学

课程综合学习设计报告

学年学期： 2022-2023学年 ☐春 ☒秋学期

课程名称： 软件技术基础

设计题目： 软件开发研究、基于链式结构的疫情期间物资配送系统

学生学院： 自动化学院

专业班级： 08122104

学生学号： 2021212981

学生姓名： 王忠全

学生成绩：

指导教师： 刘平

时间： 2022 年 12 月 18 日

摘要： 基于软件开发研究，探寻数据库的背后价值、发展趋势，与自动化专业控制类软件开发学习。数据库是存放数据的仓库。它的存储空间很大，可以存放百万条、千万条、上亿条数据。其价值包括：商业价值、社会价值、决策价值、预测价值。数据库以及渗透进我们日常生活了，和我们息息相关。而数据库的发展也必定会影响与改变我们的日常生活。随着互联网、物联网、云计算、三网融合等 I T 与通信技术的迅猛发展，数据的快速增长成了许多行业共同面对的严峻挑战和宝贵机遇，因而信息社会已经进入了大数据时代。大数据的涌现不仅改变着人们的生活与工作方式、企业的运作模式，甚至还引起科学研究模式的根本性改变。软件开发是根据用户要求建造出软件系统或者系统中的软件部分的过程。软件开发是一项包括需求捕捉、需求分析、设计、实现和测试的系统工程。软件一般是用某种程序设计语言来实现的。通常采用软件开发工具可以进行开发。控制类软件专业性高、需要用到许多数据结构内容、并且与当今万物互联，动动手机就可以让家里面开灯等等，应用场景非常广泛，在软件开发时候，也需要具备多方面知识技能协调调用

关键词： 软件开发研究 数据库 大数据时代 控制类软件开发

目 录

摘要	
关键词	
第一部分 文献学习报告	1
1.1 数据库 – 当今大数据时代的潮儿	1
1.2 数据库的发展 – 任重道远	3
1.3 软件开发中的算法要求 – 养成好习惯	4
1.4 控制类软件开发 – 万物互联	6
第二部分 综合编程设计	7
2.1 需求分析	7
2.2 系统设计	7
2.3 编程实施	9
2.4 测试结果分析	14
2.5 总结与展望	20
2.6 源代码	20
第三部分 感想和收获	26
参考文献	27

第一部分 文献学习报告

1.1 数据库 - 当今大数据时代的潮儿

1.1.1 数据库的含义

数据库是存放数据的仓库。它的存储空间很大，可以存放百万条、千万条、上亿条数据。但是数据库并不是随意地将数据进行存放，是有一定的规则的，否则查询的效率会很低。当今世界是一个充满着数据的互联网世界，充斥着大量的数据。即这个互联网世界就是数据世界。数据的来源有很多，比如出行记录、消费记录、浏览的网页、发送的消息等等。除了文本类型的数据，图像、音乐、声音都是数据。

1.1.2 数据的价值

在开始研究软件前，我不得好奇软件所服务的“数据”有什么价值。数据即是有含义的信息，能够代表反映事物的规律与内涵，于是每条数据有它的含义，而数据与数据之间也反映一定现状与含义。其价值包括：商业价值、社会价值、决策价值、预测价值。

比如身份证、信用卡、手机号码、cookie、车牌号等。在一般的场景下，有多个账号可能不是什么问题，但是一旦涉及反欺诈等需要识别到“人”的时候，则必须将之识别出来。能够还原用户真实身份和真实行为的数据，就越能够让企业在大数据竞争中保持战略优势。

又比如对某一个单品进行预测，比如推荐系统推荐了一款T恤，它有多大可能性被点击的大数据计算预测。对经营状况的预测，即对公司的整体经营进行预测，并能够用预测的结论指导公司的经营策略。

以及中国现代化进程，工业、农业、生产业产生的各行各业数据进行指标分析，来判断当今社会的发展，以及利用数据进行规划等等。可见数据小到你的手机号码，身份证，大到各行各业数据，这都是数据产生的价值。

1.1.3 国内数据库代表公司

OceanBase：支付宝蚂蚁集团原生分布式数据库【金融数据】

OceanBase 已在中国建设银行、南京银行、西安银行、中国人寿、网商银行等多家金融机构上线，同时 OceanBase 支持江西人社、中国石化、国家电网、中国移动、中国联通等企业及机构的数字化转型升级。全国 200 家头部金融客户中，1/4 的客户将 OceanBase 作为核心系统升级首选。

X-DB：阿里巴巴集团新一代分布式数据库【购物数据】

2018 年双 11 是 X-DB 的第一次大考，本次双 11 X-DB 服务于天猫 / 淘宝核心交易系统、核心物流系统、核心 IM 系统，经受了零点业务 32.5 万笔 / 秒峰值的性能考验（对应数据库峰值每秒破亿次的 SQL 调用）；同时 X-DB 支撑起了新一代单元化架构，在分布式一致性算法 Paxos 的统一框架下，第一次提供了跨 Region 分布式强一致能力，实现高效的跨 Region 数据同步、跨 Region 容灾，保证金融级的数据质量服务。













PolarDB：阿里云团队开发的数据库产品【数据库产品】

POLARDB 是阿里云自研的下一代关系型分布式数据库，POLARDB 是阿里云增速最快的数据库产品，广泛应用于互联网金融、政府便民工程、新零售、教育、游戏、社交直播等行业。

由此可见，在我们现实生活中，数据库以及渗透进我们日常生活各个方面了，和我们息息相关。而数据库的发展也必定会影响与改变我们的日常生活。

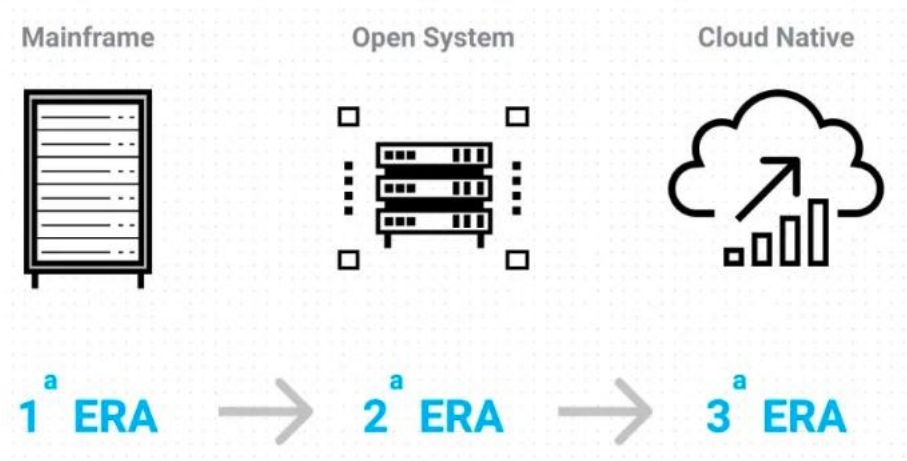
1.1.4 国外数据库代表公司

330 systems in ranking, August 2017

Rank			DBMS	Database Model	Score		
Aug 2017	Jul 2017	Aug 2016			Aug 2017	Jul 2017	Aug 2016
1.	1.	1.	Oracle 	Relational DBMS	1367.88	+7.00	-59.85
2.	2.	2.	MySQL 	Relational DBMS	1340.30	-8.81	-16.73
3.	3.	3.	Microsoft SQL Server 	Relational DBMS	1225.47	-0.52	+20.43
4.	4.	5.	PostgreSQL 	Relational DBMS	369.76	+0.32	+54.51
5.	5.	4.	MongoDB 	Document store	330.50	-2.27	+12.01
6.	6.	6.	DB2 	Relational DBMS	197.47	+6.22	+11.58
7.	7.	8.	Microsoft Access	Relational DBMS	127.03	+0.90	+2.98
8.	8.	7.	Cassandra 	Wide column store	126.72	+2.60	-3.52
9.	9.	10.	Redis 	Key-value store	121.90	+0.38	+14.57
10.	10.	11.	Elasticsearch 	Search engine	117.65	+1.67	+25.16
11.	11.	9.	SQLite	Relational DBMS	110.85	-3.02	+0.99
12.	12.	12.	Teradata	Relational DBMS	79.23	+0.86	+5.59
13.	14.	14.	Solr	Search engine	66.96	+0.93	+1.18
14.	13.	13.	SAP Adaptive Server	Relational DBMS	66.92	+0.00	-4.13
15.	15.	15.	HBase	Wide column store	63.52	-0.10	+8.01
16.	16.	17.	Splunk	Search engine	61.46	+1.17	+12.56
17.	17.	16.	FileMaker	Relational DBMS	59.65	+1.00	+4.84
18.	18.	20.	MariaDB 	Relational DBMS	54.70	+0.23	+17.82
19.	19.	19.	SAP HANA 	Relational DBMS	47.02	+0.00	+5.24
20.	20.	18.	Hive 	Relational DBMS	47.30	+1.18	-0.51

国外数据库热度排行统计

1.2 数据库发展 - 任重道远



1.2.1 数据库发展现状

近年来，随着互联网、物联网、云计算、三网融合 等 I T 与通信技术的迅猛发展，数据的快速增长成了许多行业共同面对的严峻挑战和宝贵机遇，因而信息社会已经进入了大数据时代。大数据的涌现不仅改变着人们的生活与工作方式、企业的运作模式，甚至还引起科学研究模式的根本性改变。

而推动数据库研究发展的动力是相关技术的成熟。比如，在过去的几十年里，数据挖掘技术已经成为数据库系统重要的一个组成部分。Web 搜索引擎导致了信息检索的商品化，并需要和传统的数据库查询技术集成。许多人工智能领域的研究成果也和数据库技术融合起来，这些新的技术使得我们可以处理语音、自然语言，进行不确定性推理和机器学习等。

而数据库的主流发展，主要是从信息集成、数据流管理、传感器数据库技术、

半结构化数据与 XML 数据管理、网格数据管理、DBMS 自适应管理、移动数据管理、微小型数据库、数据库用户界面等方面进行。

1.2.2 数据库发展阶段

前关系型阶段（1960-1970）：网状层次数据库初尝探索

前关系型阶段数据库的数据模型主要基于网状模型和层次模型，代表产品为 IDS 和 IMS，该类产品在当时较好地解决了数据集中存储和共享的问题，但在数据抽象程度和独立性上存在明显不足。

关系型阶段（1970-2008）：关系型数据库大规模应用

关系型阶段以 IBM 公司研究员 E.F.Codd 提出关系模型概念，论述范式理论作为开启标志，期间诞生了一批以 DB2、Sybase、Oracle、SQLServer、MySQL、PostgreSQL 等为代表的广泛应用的关系型数据库，该阶段技术脉络逐步清晰、市场格局趋于稳定。

后关系型阶段（2008-至今）：模型拓展与架构解耦并存

谷歌的三篇论文开启后关系型数据库阶段，该阶段由于数据规模爆炸增长、数据类型不断丰富、数据应用不断深化，技术路线呈现多样化发展。随着各行业数字化转型不断深入，5G、云计算等新兴技术快速发展，传统数据库的应用系统纷纷优化升级。全球市场格局剧烈变革，我国数据库产业进入重大发展机遇期。

1.2.3 数据库发展问题

- (1) 升级变配慢
- (2) 新增只读节点慢
- (3) 存储空间有上限
- (3) 存储空间有上限

1.3 软件开发中算法要求 - 养成好习惯

1.3.1 算法

算法是求解一类问题的随意一种特殊方法，一个算法是对特定问题求解步骤的一种描写叙述。

1.3.2 软件开发

软件开发是根据用户要求建造出软件系统或者系统中的软件部分的过程。软件开发是一项包括需求捕捉、需求分析、设计、实现和测试的系统工程。软件一般是用某种程序设计语言来实现的。通常采用软件开发工具可以进行开发。软件分为系统软件和应用软件，并不只是包括可以在计算机上运行的程序，与这些程序相关的文件一般也被认为是软件的一部分。软件设计思路和方法的一般过程，包括设计软件的功能和实现的算法和方法、软件的总体结构设计和模块设计、编程和调试、程序联调和测试，然后进行编写再提交程序。

1.3.3 软件开发中算法要求

①正确性

永不给用户错误的结果

②可读性

描述在其他开发人员没有进行太多联想或猜测的情况下就能理解代码的含义的难易程度。

③健壮性

健壮性是指软件对于规范要求以外的输入情况的处理能力。所谓健壮的系统是指对于规范要求以外的输入能够判断出这个输入不符合规范要求，并能有合理的处理方式。另外健壮性有时也和容错性，可移植性，正确性有交叉的地方。

④可移植性

可移植性是软件质量之一，良好的可移植性可以提高软件的生命周期。

代码的可移植性主题是软件，软件可移植性指与软件从某一环境转移到另一环境下的难易程度。这里的环境包括软件环境，硬件环境和系统的组织环境。

为获得较高的可移植性，在设计过程中常采用通用的程序设计语言和运行支撑环境。尽量不用与系统的底层相关性强的语言。

⑤可维护性

系统的可维护性是衡量一个系统的可修复(恢复)性和可改进性的难易程度。所谓可修复性是指在系统发生故障后能够排除(或抑制)故障予以修复，并返回到原来正常运行状态的可能性。而可改进性则是系统具有接受对现有功能的改进，增加新功能的可能性。

因此，可维护性实际上也是对系统性能的一种不可缺少的评价体系，它主要包括两个方面：首先是评价一个系统在实施预防型和纠正型维护功能时的难易程度，其中包括对故障的检测、诊断、修复以及能否将该系统重新进行初始化等功能；其次，则是衡量一个系统能接受改进，甚至为了进一步适应外界(或新的)环境而进行功能修改的难易程度。事实上，可维护性是可信性属性中一项相当重要的评价标准。可维护性的优劣可能直接影响到系统的可靠性和可信性。

⑥效率与低存储量需求

⑦算法效率的度量

1.4 控制类软件开发 - 万物互联

1.4.1 我的未来

学习电工技术、电子技术、控理论、自动检测与仪表、信息处理、系统工程、计算机技术与应用和网络技术等较宽广领域的工程技术基础和一定的业知识，具有自动化系统分析、设计、开发与研究的基本能力，综合素质高，具有坚实理论基础和创新能力。

电路、信号与系统、PLC 编程应用、模拟电子技术、数字电子技术、自动控制原理、现代控理论、微机原理及应用、软件技术基础、电机与拖动、电力电子技术、计算机控技术、系统仿真、计算机网络、运动控、过程控、单片机与嵌入式系统原理、计算机辅助设计、专业英语、智能控制、C 语言程序设计、C++语言、软件技术基础、数据结构等等。

工业控制方向：plc 语言

偏硬件的方向：hdl 语言，Multisim 等数字电路设计仿真工具，keil、cube 等嵌入式开发工具

机器人方向：ROS

工业设计方向：DSPACE 等软硬结合系统

控制类软件专业性高、需要用到许多数据结构内容、并且与当今万物互联，动手机就可以让家里面开灯等等，应用场景非常广泛，在软件开发时候，也需要具备多方面知识技能协调调用，当然最基础的还是刘老师的数据结构这门课程。

第二部分 综合编程设计

2.1 需求分析

本次综合编程设计《基于链式结构的疫情期间物资配送系统》。

结在疫情期间，各行业都受到影响，物流也不例外，但是物流，却变得更加不可或缺，物流不仅承担着物品的运输，还在成为每个人与外部世界的链接者。

作为基础设施的物流在未来将变得更加重要，尤其在非常时期，物流成为一个国家经济流通、社会流动、生活保障最不可或缺的基础依赖。

从这次疫情看到，物流的价值正在不断上升。

疫情期间，消费者的紧张情绪、各地防疫封锁措施、物资紧急需求之下，大量的线下消费转移到线上，原本处于淡季的各大电商配送平台面临着各种需求同时爆发的情况。

2.2 系统设计

2.2.1 整个系统

1. 录入物资配送信息功能

根据商家/物流公司的实际物流配送需求，将物资录入到物资配送信息系统。

2. 查找物资配送信息功能

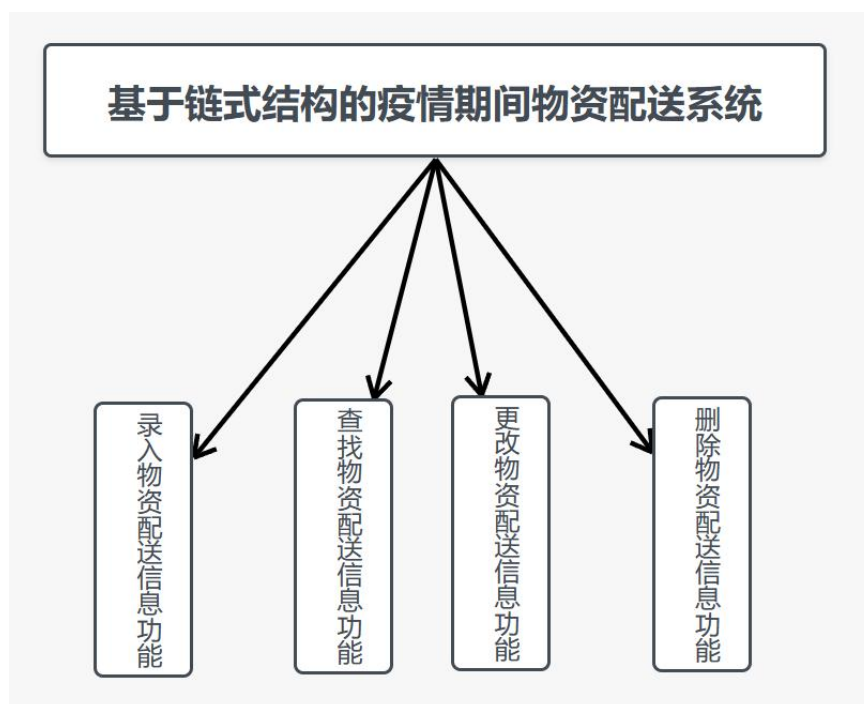
商家/物流公司可以根据实际需要查找的物流订单信息进行查找。

3. 更改物资配送信息功能

商家/物流公司可以根据实际需要查找的物流订单信息进行动态更改信息。

4. 删除物资配送信息功能

商家/物流公司可以根据实际需要查找的物流订单信息，进行动态删除。



图一 功能板块

```

*****
*****基于链式结构的疫情期间物资配送系统*****
*****
***      请选择系统功能:      ***
***      >输入1录入物资信息      ***
***      >输入2查找物资信息      ***
***      >输入3更改物资信息      ***
***      >输入4删除物资信息      ***
***      >输入-1结束系统          ***
*****
  
```

图二 预期设计系统展示图

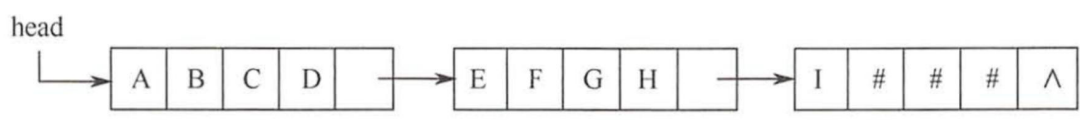
2.22 解决办法

采用链式的存储结构，将每个物资配送信息按照链表存储，如同外卖商家的订单系统，并且链式存储结构的存储空间具有随机性，链表的存储空间是动态分配，插入、删除运算方便，内存地址不连续。

即：

1. 存储空间动态分配，可以根据实际需要使用
2. 不需要地址连续的存储空间

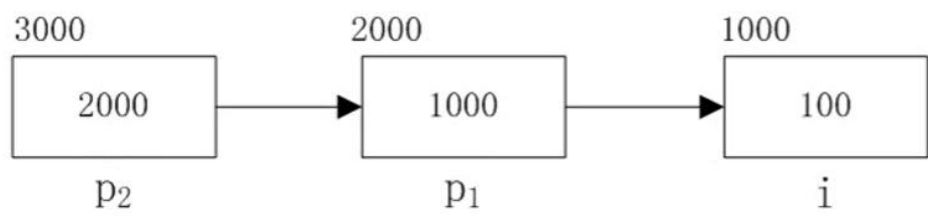
3. 插入/删除操作只须通过修改指针实现，不必移动数据元素，操作时间效率高（无论位于链表何处无论链表长度如何，操作的时间都是 $O(1)$ ）



图一 链表数据结构示意图

采用指向指针的指针即 `int**` 类型来进行修改各个函数间的信息，其使用方法如下：

`*p2`等价于`p1`，`*(p2)`等价于`*p1`，也就等价于`i`



图二 指向指针的指针示意图

2.3 编程实施



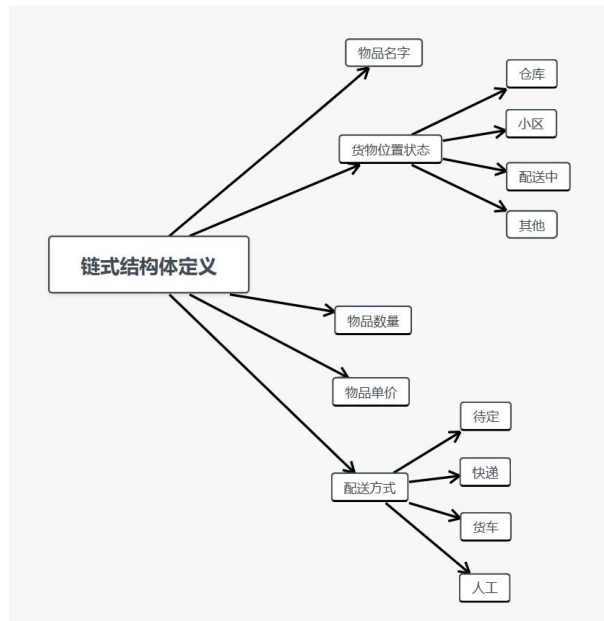
2.3.1 头文件声明、结构体定义

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define N 100

// 疫情下物资结构体定义
typedef struct object
{
    char name[N]; // 物品名字
    char location[N]; // 货物位置状态 -仓库 / 小区 / 配送中
    int num; // 物品数量
    float per_money; // 物品单价
    char delivery[N]; // 配送方式 待定 / 快递 / 货车 / 人工
    struct object *next; // 指针
}object;

```



定义结构体 object，结构体 object 里面存储了物品名字、货物位置状态（仓库 / 小区 / 配送中）不限制于三个状态的货物状态、物品数量、物品单价、配送方式（待定 / 快递 / 货车 / 人工）等不限制于四种方式的配送方式，以及链式存储结构的指针 next。

2.3.2 系统界面展示函数 System_Show

```

// 系统界面显示函数
int System_Show(int x)
{
    printf("***** 基于链式结构的疫情期间物流配送系统 *****\n");
    printf("***** 请选择系统功能 *****\n");
    printf("***** >输入1录入物资信息 *****\n");
    printf("***** >输入2查找物资信息 *****\n");
    printf("***** >输入3更改物资信息 *****\n");
    printf("***** >输入4删除物资信息 *****\n");
    printf("***** >输入-1结束系统 *****\n");
    printf("***** 我的选择是: *****");
    scanf("%d", &x); // 用户选择
    return x;
}

```

展示界面，每次选择功能后都展示此界面，返回用户选择的功能。

2.3.3 创建链表函数 Creat_list

```
// 创建链表函数
object* Creat_list(object* rearl)
{
    object* p = (object*)malloc(sizeof(object));

    getchar();
    printf("          ***请输入货物名字: ");
    scanf("%s",p->name);
    printf("          ***请输入货物位置状态 (仓库 / 小区 / 配送中): ");
    getchar();
    scanf("%s",p->location);
    printf("          ***请输入货物数量: ");
    getchar();
    scanf("%d",p->num);
    printf("          ***请输入货物单价: ");
    getchar();
    scanf("%f",p->per_money);
    printf("          ***请输入货物配送方式(待定 / 快递 / 货车 / 人工): ");
    scanf("%s",p->delivery);
    printf("          *****物品信息录入成功! *****\n");

    p->next = NULL;
    rearl->next = p;
    rearl = p;
    return rearl;
}
```

根据链表结构体创建函数，使用 object* rearl 作为尾指针，使用 malloc 函数申请了 p 节点内存空间，不断输入物资的各项信息，p 节点作为新节点链接到前面的尾节点，然后尾指针后移，返回尾指针，通过函数调用，如此循环往复，如同系统不断创建新订单。

2.3.4 物品信息展示函数 List_ALL_show

```
// 物品信息显示
void List_ALL_show(object* p)
{
    printf("          *****\n");
    printf("          *****疫情期间物流配送系统物资表 *****\n");
    while(p!= NULL)
    {
        printf("          *****\n");
        printf("          物品: %s          ***\n",p->name);
        printf("          货物状态: %s          ***\n",p->location);
        printf("          货物数量: %d          ***\n",p->num);
        printf("          货物单价: %.2f          ***\n",p->per_money);
        printf("          货物配送方式: %s          ***\n",p->delivery);
        printf("          *****\n");
        p = p->next;
    }
}
```

作为可视化程序最重要的部分，使用 while 循环不断遍历程序，使链表全部信息得以展示，其他函数都可以调用具有普适性。

2.3.5 物品信息展示函数 List_single_show

```
// 单次物品展示
void List_single_show(object* p)
{
    printf("          *****\n");
    printf("    ***物品: %s          ***\n",p->name);
    printf("    ***货物状态: %s      ***\n",p->location);
    printf("    ***货物数量: %d       ***\n",p->num);
    printf("    ***货物单价: %.2f     ***\n",p->per_money);
    printf("    ***货物配送方式: %s   ***\n",p->delivery);
    printf("          *****\n");
}
}
```

考虑到某些步骤需要单独考虑可视化链表信息，由此设计单独物品信息展示函数，并且在其他函数都可以调用。

2.3.6 物品信息查询函数 find

```
// 查找函数 找到物品函数 ( 多函数链表 )
void find(object* p, char F[],int find_choice)
{
    int find_flag = 0; // 找到
    // 物品名称
    if(find_choice == 1)
    {
        while(p != NULL)
        {
            if(strcmp(p->name, F) == 0)
            {
                List_single_show(p); // 显示
                find_flag = 1; // 找到
            }
            p = p->next;
        }
        if( find_flag == 0)
        {
            printf("          ***没找到!\n");
        }
    }
    // 物品位置
    if(find_choice == 2)
    {
        while(p != NULL)
        {
            if(strcmp(p->location, F) == 0)
            {
                List_single_show(p); // 显示
                find_flag = 1; // 找到
            }
            p = p->next;
        }
        if(find_flag == 0)
        {
            printf("          ***没找到!\n");
        }
    }
    // 物品配送方式
    if(find_choice == 3)
    {
        while(p != NULL)
        {
            if(strcmp(p->delivery, F) == 0)
            {
                List_single_show(p); // 显示
                find_flag = 1; // 找到
            }
            p = p->next;
        }
        if(find_flag == 0)
        {
            printf("          ***没找到!\n");
        }
    }
}
}
```

首先形参是头节点*p 、查询内容 char F[]字符串数组、用户选择的查询分类。使用 if 语句划分三个查询程序，每个查询程序使用 strcmp() 字符串处理函数来比较链表物资信息和待查物资信息。使用 while(p!=NULL) 循环函数遍历链表，p=p->next 语句不断更新指针。

2.3.7 更改链表函数 change

```

// 修改链表函数
void change(object** p)
{
    int find_flag = 0, n = 0;
    char things[N] = {'\0'}; // 需要更改的物品
    char change_Name[N] = {'\0'}; // 更改名字
    char change_Location[N] = {'\0'}; // 更改位置
    int change_Num = 0;
    float change_Per_money = 0;
    char change_Delivery[N] = {'\0'};

    printf("    ***输入需要更改的物资分类:\n");
    printf("    ***输入更改物品的名称:\n");
    getch();
    scanf("%s", things);
    *p = (*p) ->next;
    while(*p != NULL)
    {
        if(strcmp((*p) ->name, things) == 0)
        {
            list_single_show(*p); // 显示
            printf("    ***更改其名字:\n");
            scanf("%s", n);
            switch(n)
            {
                case 1:
                {
                    printf("    ***输入更改名字:");
                    scanf("%s", change_Name);
                    strcpy((*p) ->name, change_Name);
                    break;
                }
                case 2:
                {
                    printf("    ***输入更改状态:");
                    scanf("%s", change_Location);
                    strcpy((*p) ->location, change_Location);
                    break;
                }
                case 3:
                {
                    printf("    ***输入更改数量:");
                    scanf("%d", &change_Num);
                    (*p) ->num = change_Num;
                    break;
                }
                case 4:
                {
                    printf("    ***输入更改单价:\n");
                    scanf("%f", &change_Per_money);
                    (*p) ->per_money = change_Per_money;
                    break;
                }
                case 5:
                {
                    printf("    ***输入更改配送方法:");
                    scanf("%s", change_Delivery);
                    strcpy((*p) ->delivery, change_Delivery);
                    break;
                }
            }
            printf("    ***输入错误!\n");
            printf("    ***更改成功!\n");
            list_single_show(*p);
            find_flag = 1; // 成功
        }
        *p = (*p) ->next;
    }
    if(find_flag == 0)
    {
        printf("    ***没找到!\n");
    }
}

```

使用指向指针的指针 object** p，使传入的参数能够改变主函数的值，*p 就是改变主函数链表的值。因为头节点是不存储信息的，即要从 *p->next 开始，使用 while(P!=NULL) 不断循环，找到需要改变的链表物资信息所在存储空间，使用 switch 函数划分为 5 个需要改变的部分，由用户输入来自行选择需要改变信息。使用 (*p) = (*p)->next 不断后退查找需改变信息。使用 strcmp((*p)->name, things) == 0 字符串处理判断条件来对需要修改的信息进行对比。使用 strcpy((*p)->name, change_Name) 字符串处理来进行更改。

2.3.8 删除链表函数 delted


```
// 删除链表函数
void delted(object** q)
{
    char things[N] = {'\0'}; // 查找需要更改的物品
    int k = 0;
    int find_flag = 0;

    printf("        ***请输入需要删除的物资分类...\n");
    printf("        ***输入删除物品的名称...\n");
    getchar();
    scanf("%s", things);
    while ((*q)->next != NULL)
    {
        if (strcmp((*q)->next->name, things) == 0)
        {
            printf("        ***是否删除以下物资? \n");
            List_single_show((*q)->next);
            printf("        ***输入1删除输入0取消: ");
            scanf("%d", &k);
            if (k == 1)
            {
                (*q)->next = (*q)->next->next; // 删除
                printf("        ***已成功删除!\n");
            }
            find_flag = 1;
        }
        (*q) = (*q)->next;
    }
    if (find_flag == 0)
    {
        printf("        ***没找到!\n");
    }
}
}
```

使用指向指针的指针 `object** p`，同理，使传入的参数能够改变主函数的值，`*p` 就是改变主函数链表的值。使用 `while ((*q)->next != NULL)` 循环判断条件，不断遍历链表，使用字符串判断条件 `strcmp((*q)->next->name, things) == 0` 找到需要删除的物资名称，使用 `(*q)->next = (*q)->next->next` 语句将要删除的物资信息下一个节点地址于链接到前面一个，实现链表重建，即删除了链表中的某个物资信息。

2.4 测试结果分析

对设计的功能进行测试，分别给出运行结果截图，分析是否满足功能需求。

2.4.1 录入功能测试

```
"C:\Users\Akaxi\Desktop\Akaxi sys4\bin\Debug\Akaxi sys4.exe"

*****基于链式结构的疫情期间物资配送系统*****
*****
*** 请选择系统功能: ***
*** >输入1录入物资信息 ***
*** >输入2查找物资信息 ***
*** >输入3更改物资信息 ***
*** >输入4删除物资信息 ***
*** >输入-1结束系统 ***
*****
***我的选择是: 1
*****录入物资信息*****
***请输入货物名字: 口罩
***请输入货物位置状态(仓库 / 小区 / 配送中): 仓库
***请输入货物数量: 100
***请输入货物单价: 0.5
***请输入货物配送方式(待定 / 快递 / 货车 / 人工): 货车
*****物品信息录入成功! *****
***输入1继续增加货物
***输入0停止增加货物: 1
***请输入货物名字: 消毒水
***请输入货物位置状态(仓库 / 小区 / 配送中): 小区
***请输入货物数量: 58
***请输入货物单价: 4.6
***请输入货物配送方式(待定 / 快递 / 货车 / 人工): 快递
*****物品信息录入成功! *****
***输入1继续增加货物
***输入0停止增加货物: 1
***请输入货物名字: 蔬菜
***请输入货物位置状态(仓库 / 小区 / 配送中): 配送中
***请输入货物数量: 60
***请输入货物单价: 10.5
***请输入货物配送方式(待定 / 快递 / 货车 / 人工): 人工
*****物品信息录入成功! *****
***输入1继续增加货物
***输入0停止增加货物: 1
***请输入货物名字: 水果
***请输入货物位置状态(仓库 / 小区 / 配送中): 仓库
***请输入货物数量: 50
***请输入货物单价: 28.8
***请输入货物配送方式(待定 / 快递 / 货车 / 人工): 人工
*****物品信息录入成功! *****
***输入1继续增加货物
***输入0停止增加货物: 0
*****
```

输入各个物资的相关信息，比如口罩、消毒水、蔬菜、水果等，录入信息成功，并且调用物品信息展示函数 List_ALL_show 打印链表成功。

```
*****疫情期间物资配送系统物资表*****
*****
***物品: 口罩 ***
***货物状态: 仓库 ***
***货物数量: 100 ***
***货物单价: 0.50 ***
***货物配送方式: 货车 ***
*****
***物品: 消毒水 ***
***货物状态: 小区 ***
***货物数量: 58 ***
***货物单价: 4.60 ***
***货物配送方式: 快递 ***
*****
***物品: 蔬菜 ***
***货物状态: 配送中 ***
***货物数量: 60 ***
***货物单价: 10.50 ***
***货物配送方式: 人工 ***
*****
***物品: 水果 ***
***货物状态: 仓库 ***
***货物数量: 50 ***
***货物单价: 28.80 ***
***货物配送方式: 人工 ***
*****
```

2.4.2 查找物资信息测试

```

*****
*****基于链式结构的疫情期间物资配送系统*****
*****
***      请选择系统功能:      ***
***      >输入1录入物资信息    ***
***      >输入2查找物资信息    ***
***      >输入3更改物资信息    ***
***      >输入4删除物资信息    ***
***      >输入-1结束系统      ***
*****
***我的选择是: 2
*****查找物资信息*****
***请输入需要查找的分类:
***输入1查找物品名称
***输入2查找物品状态
***输入3查找物品配送方式 : 1
***请输入需要查找的物品名称 / 状态 / 配送方式: 消毒水
*****
***物品: 消毒水      ***
***货物状态: 小区    ***
***货物数量: 58      ***
***货物单价: 4.60    ***
***货物配送方式: 快递 ***
*****

```

输入 2 进入查找功能，并且输入 1 按照名字查找消毒水，结果打印成功。

```

***输入1继续查找货物
***输入0停止查找货物: 1
***请输入需要查找的分类:
***输入1查找物品名称
***输入2查找物品状态
***输入3查找物品配送方式 : 2
***请输入需要查找的物品名称 / 状态 / 配送方式: 仓库
*****
***物品: 口罩      ***
***货物状态: 仓库  ***
***货物数量: 100   ***
***货物单价: 0.50  ***
***货物配送方式: 货车 ***
*****
***物品: 水果      ***
***货物状态: 仓库  ***
***货物数量: 50    ***
***货物单价: 28.80 ***
***货物配送方式: 人工 ***
*****

```

继续查找，按照物品状态查找在仓库的物品，结果是口罩与水果，成功。

```

***输入1继续查找货物
***输入0停止查找货物: 1
***请输入需要查找的分类:
***输入1查找物品名称
***输入2查找物品状态
***输入3查找物品配送方式 : 3
***请输入需要查找的物品名称 / 状态 / 配送方式: 人工
*****
***物品: 蔬菜      ***
***货物状态: 配送中 ***
***货物数量: 60     ***
***货物单价: 10.50  ***
***货物配送方式: 人工 ***
*****
***物品: 水果      ***
***货物状态: 仓库  ***
***货物数量: 50     ***
***货物单价: 28.80 ***
***货物配送方式: 人工 ***
*****

```

最后按照物品配送方式进行查找，查找人工配送，结果是蔬菜和水果，成功。

2.4.3 更改物资信息测试

```
*****
*****基于链式结构的疫情期间物资配送系统*****
*****
***      请选择系统功能:                ***
***      >输入1录入物资信息              ***
***      >输入2查找物资信息              ***
***      >输入3更改物资信息              ***
***      >输入4删除物资信息              ***
***      >输入-1结束系统                  ***
*****
***我的选择是: 3
*****更改物资信息*****
*****疫情期间物资配送系统物资表*****
*****
***物品: 口罩                            ***
***货物状态: 仓库                        ***
***货物数量: 100                         ***
***货物单价: 0.50                        ***
***货物配送方式: 货车                    ***
*****
***物品: 消毒水                          ***
***货物状态: 小区                        ***
***货物数量: 58                          ***
***货物单价: 4.60                        ***
***货物配送方式: 快递                    ***
*****
***物品: 蔬菜                            ***
***货物状态: 配送中                      ***
***货物数量: 60                          ***
***货物单价: 10.50                       ***
***货物配送方式: 人工                    ***
*****
***物品: 水果                            ***
***货物状态: 仓库                        ***
***货物数量: 50                          ***
***货物单价: 28.80                       ***
***货物配送方式: 人工                    ***
*****
***请输入需要更改的物资分类:
***输入更改物品的名称: 消毒水
*****
***物品: 消毒水                          ***
***货物状态: 小区                        ***
***货物数量: 58                          ***
***货物单价: 4.60                        ***
***货物配送方式: 快递                    ***
*****
***更改其名字1? 状态2? 数量3? 单价4? 配送方式5? 1
***输入更改名字: 温度计
更改成功!
*****
***物品: 温度计                          ***
***货物状态: 小区                        ***
***货物数量: 58                          ***
***货物单价: 4.60                        ***
***货物配送方式: 快递                    ***
*****
```



```

*****
*****疫情期间物资配送系统物资表*****
*****
***物品：口罩***
***货物状态：仓库***
***货物数量：100***
***货物单价：0.50***
***货物配送方式：货车***
*****
***物品：温度计***
***货物状态：小区***
***货物数量：58***
***货物单价：4.60***
***货物配送方式：快递***
*****
***物品：蔬菜***
***货物状态：配送中***
***货物数量：60***
***货物单价：10.50***
***货物配送方式：人工***
*****
***物品：水果***
***货物状态：仓库***
***货物数量：50***
***货物单价：28.80***
***货物配送方式：人工***
*****

```

输入 3 进入更改物资功能，输入名字消毒水，输入改名字 1，将消毒水名字改为温度计，结果成功且原链表其他物品名字不变。

2.4.4 删除物资信息测试

```

*****
*****基于链式结构的疫情期间物资配送系统*****
*****
***      请选择系统功能:      ***
***      >输入1录入物资信息      ***
***      >输入2查找物资信息      ***
***      >输入3更改物资信息      ***
***      >输入4删除物资信息      ***
***      >输入-1结束系统      ***
*****
***我的选择是: 4
*****删除物资信息*****
*****疫情期间物资配送系统物资表*****
*****
***物品: 口罩      ***
***货物状态: 仓库      ***
***货物数量: 100      ***
***货物单价: 0.50      ***
***货物配送方式: 货车      ***
*****
***物品: 温度计      ***
***货物状态: 小区      ***
***货物数量: 58      ***
***货物单价: 4.60      ***
***货物配送方式: 快递      ***
*****
***物品: 蔬菜      ***
***货物状态: 配送中      ***
***货物数量: 60      ***
***货物单价: 10.50      ***
***货物配送方式: 人工      ***
*****
***物品: 水果      ***
***货物状态: 仓库      ***
***货物数量: 50      ***
***货物单价: 28.80      ***
***货物配送方式: 人工      ***
*****
***请输入需要删除的物资分类:
***输入删除物品的名称: 蔬菜
***是否删除以下物资?
*****
***物品: 蔬菜      ***
***货物状态: 配送中      ***
***货物数量: 60      ***
***货物单价: 10.50      ***
***货物配送方式: 人工      ***
*****
***输入1删除输入0取消: 1
***已成功删除!

```

```

*****
*****疫情期间物资配送系统物资表*****
*****
***物品: 口罩      ***
***货物状态: 仓库      ***
***货物数量: 100      ***
***货物单价: 0.50      ***
***货物配送方式: 货车      ***
*****
***物品: 温度计      ***
***货物状态: 小区      ***
***货物数量: 58      ***
***货物单价: 4.60      ***
***货物配送方式: 快递      ***
*****
***物品: 水果      ***
***货物状态: 仓库      ***
***货物数量: 50      ***
***货物单价: 28.80      ***
***货物配送方式: 人工      ***
*****

```

输入 4 进入删除功能，输入删除物品名字蔬菜，结果删除成功，原链表输出成功。

2.5 总结与展望

总结：虽然部分功能实现还并未完善，但是在敲代码时切身体会到了数据结构的魅力，使用数据结构就像是有了工具箱，在工具里找出合适的方法来实现各种功能，在刘老师的课堂上面认真真学习了，知道了栈、队列、链表、二叉树、图等等数据结构，体会到了行行代码之间的逻辑于关联。

在实现功能时也常常报错，我通过查询知乎、CSDN、论坛等等途径来寻找自己的问题，同时拓宽自己的知识，查漏补缺，完善自己。

当然通过本次《基于链式结构的疫情期间物资配送系统》的撰写，我切实感受到，从一个想法到一个步骤到一行代码再到整个程序写出，通过实践初尝代码世界的奥妙，也感受到刘平老师传授数据结构这门课程的实用性，最后感谢刘老师的孜孜教诲，能让我一行一行敲代码，学完数据结构课程后能够灵活编写属于自己的一个模块化程序。

展望：可以在基于《基于链式结构的疫情期间物资配送系统》的许多功能上进行调整补充，比如分为两个用户配送员以及顾客，分别对此物资配送系统进行操作；再如给系统增加统计函数，统计整个数据以及处理结构进行大数据分析，提供预期数据为商家以及顾客作为参考建议等。

2.6 源代码

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define N 100

// 疫情下物资结构体定义
typedef struct object
{
    char name[N];    // 物品名字
    char location[N]; // 货物位置状态 -仓库 / 小区 / 配送中
    int num;          // 物品数量
    float per_money;  // 物品单价
    char delivery[N]; // 配送方式 待定 / 快递 / 货车 / 人工

    struct object *next; // 指针
}object;

// 系统界面展示函数
int System_Show(int x)
{
    printf("
    *****
    *****\n");
    printf("          ***** 基于链式结构的疫情期间物资配送系统
    *****\n");
    printf("
    *****
    *****\n");
}
```

```

        printf("        ***        请选择系统功能:");
        printf("        ***        >输入 1 录入物资信息");
        printf("        ***        >输入 2 查找物资信息");
        printf("        ***        >输入 3 更改物资信息");
        printf("        ***        >输入 4 删除物资信息");
        printf("        ***        >输入-1 结束系统");
        printf("        ***\n");
        printf("        *****\n");
        printf("        *****\n");
        printf("        ***我的选择是: ");
        scanf("%d",&x); // 用户选择
        return x;
    }

// 创建链表函数
object* Creat_list(object* rear1)
{
    object* p = (object*)malloc(sizeof(object*));

    getchar();
    printf("        ***请输入货物名字: ");
    scanf("%s",p->name);
    printf("        ***请输入货物位置状态(仓库 / 小区 / 配送中): ");
    getchar();
    scanf("%s",p->location);
    printf("        ***请输入货物数量: ");
    getchar();
    scanf("%d",&p->num);
    printf("        ***请输入货物单价: ");
    getchar();
    scanf("%f",&p->per_money);
    getchar();
    printf("        ***请输入货物配送方式(待定 / 快递 / 货车 / 人工):");
    scanf("%s",p->delivery);
    printf("        *****物品信息录入成功!*****\n");

    p->next = NULL;
    rear1->next = p;
    rear1 = p;
    return rear1;
}

// 物品信息展示
void List_ALL_show(object* p)
{
    printf("        *****\n");
    printf("        *****疫情期间物资配送系统物资表*****\n");
    while(p!= NULL)
    {
        printf("        *****物品: %s\n",p->name);
        printf("        ***货物状态: %s\n",p->location);
        printf("        ***货物数量: %d\n",p->num);
        printf("        ***货物单价: %.2f\n",p->per_money);
        printf("        ***货物配送方式: %s\n",p->delivery);
        printf("        *****\n");

        p = p->next;
    }
}

// 单次物品展示
void List_single_show(object* p)
{
    printf("

```



```

*****
*****\n");
    printf("          *** 物品： %s
***\n",p->name);
    printf("          *** 货物状态： %s
***\n",p->location);
    printf("          *** 货物数量： %d
***\n",p->num);
    printf("          *** 货物单价： %.2f
***\n",p->per_money);
    printf("          *** 货物配送方式： %s
***\n",p->delivery);
    printf("
*****
*****\n");
}

// 查询链表函数 找到物资函数( 很多函数都是
基于此)
void find(object* p, char F[],int find_choice)
{
    int find_flag = 0; // 默认没找到
    // 找物品名字
    if(find_choice == 1 )
    {
        while(p != NULL)
        {
            if(strcmp(p->name, F) == 0 )
            {
                List_single_show(p); // 展示
                find_flag = 1;      // 标志
找到了
            }
            p = p->next;
        }
        if( find_flag == 0)
        {
            printf("          ***没找到! \n");
        }
    }
    // 找物品状态
    if(find_choice == 2)
    {
        while(p != NULL )
            {
                if(strcmp(p->location, F) == 0)
                {
                    List_single_show(p); // 展示
                    find_flag = 1;      // 标志
找到了
                }
                p = p->next;
            }
        if(find_flag == 0)
        {
            printf("          ***没找到! \n");
        }
    }
    // 找物品配送方式
    if(find_choice == 3)
    {
        while(p != NULL )
        {
            if(strcmp(p->delivery, F) == 0)
            {
                List_single_show(p); // 展示
                find_flag = 1;      // 标志
找到了
            }
            p = p->next;
        }
        if(find_flag == 0)
        {
            printf("          ***没找到! \n");
        }
    }
}

// 更改链表函数
void change(object** p)
{
    int find_flag = 0 , n = 0 ;
    char things[N] = {'\0'}; // 查找需要更改的物品
    char change_Name[N] = {'\0'}; // 更改的名字
    char change_Location[N] = {'\0'};
    int change_Num = 0;
    float change_Per_money = 0;

```

```

char change_Delivery[N] = {'\0'};

printf("        ***请输入需要更改的物资
分类: \n");
printf("        ***输入更改物品的名称:
");
getchar();
scanf("%s",things);
*p = (*p)->next ;
while((*p) != NULL )
{
    if(strcmp((*p)->name, things) == 0)
    {
        List_single_show(*p); // 展示
        printf("        ***更改其名字
1? 状态 2? 数量 3? 单价 4? 配送方式 5? ");
        scanf("%d",&n);
        switch( n )
        {
            case 1:
                {
                    printf("        ***输入
更改名字: ");
                    scanf("%s",change_Name);
                    strcpy((*p)->name,
change_Name);
                    break;
                }
            case 2:
                {
                    printf("        ***输入
更改状态: ");
                    scanf("%s",change_Location);
                    strcpy((*p)->location,
change_Location);
                    break;
                }
            case 3:
                {
                    printf("        ***输入
更改数量: ");
                    scanf("%d",&change_Num);
                    (*p)->num = change_Num;

break;
}
}
}
}

case 4:
{
    printf("        ***输入
更改单价: ");
    scanf("%f",&change_Per_money);
    (*p)->per_money =
change_Per_money;
    break;
}
}

case 5:
{
    printf("        ***输入
更改配送方式: ");
    scanf("%s",change_Delivery);
    strcpy((*p)->delivery,
change_Delivery);
    break;
}
}

default:
    printf("        ***输入错误!
");
}

printf("更改成功! \n");
List_single_show(*p);
find_flag = 1; // 标志找到
了
}

(*p) = (*p)->next ;
}

if(find_flag == 0)
{
    printf("        ***没找到! \n");
}
}

// 删除链表函数
void delted(object** q)
{
    char things[N] = {'\0'}; // 查找需要更改的物品
    int k = 0;

```

```

int find_flag = 0;

printf("          ***请输入需要删除的物资
分类: \n");
printf("          ***输入删除物品的名称:
");
getchar();
scanf("%s",things);
while((*q)->next!= NULL )
{
    if(strcmp((*q)->next->name, things) == 0)
    {
        printf("          ***是否删除以下
物资? \n");
        List_single_show((*q)->next);
        printf("          ***输入 1 删除输
入 0 取消: ");
        scanf("%d",&k);
        if( k ==1 )
        {
            (*q)->next = (*q)->next->next; //
删除
            printf("          ***已成功删
除! \n");
        }
        find_flag = 1;
    }
    (*q) = (*q)->next ;
}
if(find_flag == 0)
{
    printf("          ***没找到! \n");
}
}

```

```

int main()
{
    object* head = (object*)malloc(sizeof(object*));
    head->name[N] = '\0';
    head->location[N] = '\0';
    head->num = 0;
    head->per_money = 0;
    head->delivery[N] = '\0';
    head->next = NULL;

```

```

object* rear1 = head;
object* rear2 = head;
object* rear3 = head;

int choice = 0 ;    // 用户选择
int Creat_flag = 0; // 创建链表选择
int Find_flag = 0;  // 查找选择
int find_choice = 0; // 查询分类
char F[N] = {'\0'}; // 查询的字符串
choice = System_Show(choice); // 界面展示

while(choice != -1) // 系统大循环 choice = -1
结束程序
{
    if(choice == 1)
    {
        printf("
***** 录 入 物 资 信 息
*****\n");
        do
        {
            rear1 = Creat_list(rear1); // 创建
链表
            printf("          ***输入 1 继
续增加货物\n          ***输入 0 停止增加货物:
");
            scanf("%d",&Creat_flag);
        }while(Creat_flag != 0);

        List_ALL_show(head->next); // 展 示
链表
    }
}

```

```

if(choice == 2)
{
    printf("
***** 查 找 物 资 信 息
*****\n");
    do
    {
        printf("          ***请输入需
要查找的分类: \n");

```

```

printf("          ***输入 1 查
找物品名称\n          ***输入 2 查找物品状态
\n          ***输入 3 查找物品配送方式  : ");
scanf("%d",&find_choice);
printf("          ***请输入需
要查找的物品名称 / 状态 / 配送方式: ");
getchar();
scanf("%s",F);

find(head->next,F,find_choice);

printf("\n          ***输入 1
继续查找货物\n          ***输入 0 停止查找货
物: ");

scanf("%d",&Find_flag);
}while(Find_flag != 0);
}

if(choice == 3)
{
printf("
***** 更改物资信息
*****\n");
List_ALL_show(head->next); // 展示
链表

rear2 = head;
change(&rear2); // 更改物品信息
List_ALL_show(head->next);

}

if(choice == 4)
{
printf("
***** 删除物资信息
*****\n");
List_ALL_show(head->next); // 展示
链表

rear3 = head;
delted(&rear3);
List_ALL_show(head->next);

}

choice = System_Show(choice); // 展示
界面，下一次循环
}

List_ALL_show(head->next); // 展示系统
printf("          *****
系统程序正常结束! *****\n");

}

```

第三部分 感想和收获

①在一定程度上改变了我的写代码习惯。之前我是先写主函数，而且喜欢把所有功能堆砌在主函数里面，缺乏分块编写的思维。通过学习刘平老师的课程，我逐渐养成了“分而治之”的思想，与系统化模块化思想，每个独立函数功能，主函数与函数之间的关系，形参与变量等等，这对于我的写代码习惯有莫大帮助。

②在一定程度上纠正了我的错误。以前总喜欢写全局变量，但是通过本次实验，我的两个程序设计都没有用到全局变量，都是主函数之间的参数传递，地址引用等等，这对于我理解计算机存储结构，以及计算机的数据结构有莫大帮助。

③在一定程度上培养了我的性格与品质。编写这两个程序时，难免会出现报错，于是需要反复不停的 debug，根据运行结果以及逻辑判断，一次又一次的调试，记得调试【基于链式结构的疫情期间物资配送系统】时，就不知不觉从晚上 20:00 调试到了凌晨 1:00 并且孜孜不倦。我知道坚持与从错误中学习的重要性，这才能使我学懂，学好。对于代码的浓厚兴趣也使我在其他语言有所帮助，比如 python、C++ 等等，写代码不能草草了事模模糊糊，更多的是要注入心血与精力才能浇灌出一个良好系统程序。

④致谢。感谢刘平老师的孜孜不倦教导，通过一个学期的数据结构知识学习，真真正正把书本上的理论运用到计算机实践，并且从手中实现一个又一个程序，是一趟奇妙的代码世界的旅行，感谢老师的一路陪伴与指导，我会带着这份美好在接下的学习中不断向前。

参考文献

- [1] 王元卓,靳小龙,程学旗.网络大数据:现状与展望[J].计算机学报,2013,36(06):1125-1138.
- [2] 孟小峰,周龙骧,王珊.数据库技术发展趋势[J].软件学报,2004(12):1822-1836.
- [3] 中国信息与通信研究院,数据库发展研究报告(2021年)
- [4] 陈黎.我国数据库的发展现状与趋势[J].现代情报,2006(11):138-140.
- [5] 杨芙清.软件工程技术发展思索[J].软件学报,2005(01):1-7.
- [6] 王巧玲,王军茹,付兴建,刘丽华.自动化控制类专业课实践教学模式的探讨[J].科技视界,2016(25):122+98.DOI:10.19694/j.cnki.issn2095-2457.2016.25.086.