**《程序设计基础（1）》综合程序设计指导书**

《程序设计基础（1）》综合程序设计是学生在学习完《程序设计基础(1)》（C语言）课程之后的一个综合性、设计性实践教学环节。在教师的指导下，综合运用已学过的程序设计知识和技能，完成一项较为完整、并具有一定难度的综合程序设计任务。通过此任务，使学生掌握结构化程序设计的基本方法和基本步骤，培养学生的程序设计综合能力，为后续专业课程的学习打下较好的基础。

**一、时间安排**

时间： 1个寒假+4周（8学时）+12周（28学时）

**二、任务要求**

《程序设计基础（1）》综合程序设计是在完成《程序设计基础（1）》（C语言）课程学习基础上的一个综合性、设计性实践教学环节，要求运用“结构化程序设计”方法，从给定的“设计题目”中选择一个题目，完成一个规模较大的应用程序的设计与开发。任务完成包括三个阶段，即：1个寒假+前3周（15学时）+第4周。说明如下：

A．阶段1——1个寒假，选题、需求分析、系统设计。

B．阶段2——下学期前4周（8学时），学习链表和文件知识。

C． 阶段3——下学期第5～12周，系统实现（编码、调试）与运行，第12～16周，整理系统，提交验收作品。作品提交要求：

♦ 内容：每生一个文件夹，命名规则“学号+姓名”，例如：201641402101张三，提交内容包括：系统和报告。交到FTP。

♦ 时间：下学期第14～16周。

**三、考核与评分**

根据学生提交作品和报告，综合考核基本功能实现、系统整体运行、代码编写风格等几个方面。

**成绩评定参考标准：**

**A、优**：软件完全实现设计方案，设计方案先进，软件可靠性好；程序运行顺畅，功能完善，界面友好。说明：相应的百分制成绩为95分。

**B、良**：软件完全实现设计方案，设计方案较先进；程序运行正确，主要功能正确，界面良好。说明：相应的百分制成绩为85分。

**C、中**：软件基本实现设计方案，设计方案正确；程序运行有小错误，主要功能基本完成，界面一般。说明：相应的百分制成绩为75分。

**D、及格**：软件基本实现设计方案，设计方案基本正确；程序运行不顺畅，大部分功能基本完成，界面不友好。说明：相应的百分制成绩为65分。

**E、不及格**：软件没有基本实现设计方案，设计方案不正确。程序无法运行。说明：根据提交作品，视情况给出[0，55]范围的百分制成绩。

**注意**：软件必须是由学生自己独立完成，雷同者视情况扣分或记零分。

**四、选题范围**

**1、员工工资管理系统**

**（一）、公司现状分析**

1、公司主要有4类岗位人员：经理、技术员、销售员、销售经理，他们都有基本信息：员工号、姓名、性别、部门、岗位、年龄、当月工资等。一个部门最多只能有一个销售经理，一个部门可以有多个销售员，一个部门可以有若干个经理和若干技术员。

2、员工月工资的计算办法。

A．经理：固定月薪为8000；

B．技术员：工作时间＊小时工资（100元每小时）；

C．销售员：销售额＊4%提成；

D．销售经理：底薪（5000）＋所辖部门销售额总额＊0.5%。

请使用结构化的程序设计方法，开发一个员工工资管理系统对员工工资进行有效管理。

**（二）、系统的功能要求**

1. 添加数据：添加一个员工，只输入该员工的基本信息（姓名、年龄、性别、部门、岗位）。
2. 修改数据：
3. 根据员工号来修改任意员工的基本信息（姓名、年龄、性别、部门、岗位）。
4. 批量计算所有技术员的当月工资额（需填写技术员当月工作时间）。
5. 批量计算所有销售员的当月工资额（需填写销售员当月销售额）；然后批量计算所有销售经理的当月工资额。
6. 删除数据：根据员工号删除一个员工。
7. 查询数据：
8. 根据工号或者姓名查询员工信息。
9. 按部门显示本部门全部员工信息。
10. 分别显示4种岗位的员工信息。
11. 分页显示全部员工的信息。分页功能：每页显示10条员工的信息，有上一页、下一页、首页和尾页的功能。
12. 排序：
    1. 所有员工信息按当月工资从高到低并显示。
    2. 某个岗位的员工信息按当月工资从高到低显示。
13. 统计：
14. 统计并显示某个部门的平均月工资、最低月工资、最高月工资。
15. 统计并显示某个部门超出本部门平均月工资的人数与员工信息。
16. 统计并显示所有员工中的最低月工资和最高月工资员工的信息。
17. 统计并显示所有员工超出平均月工资的人数与员工信息。

**（三）、系统的实现技术要求**

1. 员工用结构体表示，全体员工用一个链表来表示。某一岗位的员工，可以只使用员工结构体的部分成员，比如：经理只使用基本信息成员，而不使用销售额成员、工作时间成员。
2. 必须用文件来存储员工信息：开始运行程序时从文件中读取员工信息到链表，退出程序时将员工信息保存到文件中。
3. 数据约束：
   1. 员工号：整型、不能重复、从1开始依次递增、由软件自动计算产生，不能修改。
   2. 岗位：枚举类型，分别表示经理、技术员、销售员、销售经理。
   3. 年龄：18-100岁。
   4. 性别：男或女。
   5. 姓名、性别、部门、岗位、年龄不能为空。
4. 其他：至少有两层菜单。

**（四）、建议定义如下的主要数据结构**

1、枚举型岗位：销售员、技术员、销售经理、经理

enum workpost { SALESMAN,TECHNICIAN,SALESMANAGER,MANAGER };

2、员工类型：链表结点

typedef struct node{

int id; //员工号：主键，唯一

char name[10]; //姓名

char sex[3]; //性别：男或女

int age; //年龄

char department[20]; //部门

enum workpost post; //枚举型岗位

int workHour; //当月工作小时数

double sales; //当月月销售额

double salary; //月工资

struct node \* next;

}Employee;

**2、学生成绩管理系统**

**（一）、学校现状分析**

1、某学校有本科生和研究生两类学生。

A. 本科生信息：学号、姓名、性别、专业、班级、高数成绩、英语成绩、C语言、总成绩、班级排名、校级排名。其中：学号、姓名、性别、专业、班级、高数成绩、英语成绩、C语言成绩为输入项，总成绩是计算项，班级排名是总成绩在班内的名次，校级排名是总成绩在校内的名次。

B. 研究生信息：学号、姓名、性别、专业、班级、课程综合成绩、论文成绩、总成绩、班级排名、校级排名。其中：学号、姓名、性别、专业、班级、课程综合成绩、论文成绩为输入项，总成绩是计算项，班级排名是总成绩在班内的名次，校级排名是总成绩在校内的名次。

2、学生总成绩计算方法。

A．本科生：总成绩=高数成绩+英语成绩+C语言成绩；

B．研究生：总成绩=课程综合成绩+论文成绩；

请使用结构化的程序设计方法，开发一个学生成绩管理系统对学生成绩进行有效管理。

**（二）、系统的功能要求**

1. 添加功能：分本科生和研究生两类人员，实现下列添加功能。

A．本科生：能够添加学生信息，如果高数成绩、英语成绩、C语言成绩都输入，则系统自动计算总成绩。

B．研究生：能够添加学生信息，如果课程综合成绩、论文成绩都输入，则系统自动计算总成绩。

1. 修改功能：分本科生和研究生两类人员，实现下列修改功能。

A．本科生：根据学号来修改任意学生的除学号外的信息。如果高数成绩、英语成绩、c语言成绩都存在，则系统自动计算总成绩。

B．研究生：根据学号来修改任意学生的除学号外的信息。如果课程综合成绩、论文成绩都存在，则系统自动计算总成绩

1. 删除功能：分本科生和研究生两类人员，能够根据学号删除一个学生。
2. 排名功能：分本科生和研究生两类人员，实现下列排名功能。

**说明**：排名包括班级排名和校级排名，排名规则按体育竞赛规则处理，若出现两个并列第1名，下个名次为第3名，依此类推。

**A：班级排名**：分本科生和研究生两类学生，计算每个学生总成绩在班级中的名次。

**B：校级排名**：分本科生和研究生两类学生，计算每个学生总成绩在全校中的名次。

1. 查询功能：分本科生和研究生两类人员，实现下列查询功能。
2. 分页显示全部学生的信息。分页功能：每页显示10条学生的信息，有上一页、下一页、首页和最后一页的功能。
3. 能够按班级显示本班全部学生信息。
4. 能够根据学号或者姓名查询学生信息。
5. 能够在某个班级中查询某门课成绩不及格学生信息。
6. 排序功能：分本科生和研究生两类人员，实现下列排序功能。
   1. 所有学生信息按总成绩从高到低排序并显示。
   2. 某个班学生信息按总成绩从高到低排序并显示。
7. 统计功能：分本科生和研究生两类人员，实现下列统计与显示功能。
8. 统计某班级某课程的平均成绩、最高成绩、最低成绩。如果学生该门课没有成绩，统计平均成绩时忽略该生。
9. 统计某班级某课程超过课程平均成绩的学生名单及人数。
10. 统计某班级某课程不及格学生名单及人数。
11. 统计某班级某课程不同等级的学生人数。等级标准：优—大于等于90；良—大于等于80且小于90；中：大于等于70且小于80；及格：大于等于60且小于70；不及格：小于60。

**（三）、系统的实现技术要求**

1. 学生用结构体表示，全体学生用一个链表来表示。某一类别的学生，可以只使用学生结构体的部分成员，比如：本科生只使用属于本科生的信息成员，研究生只使用属于研究生的信息成员。
2. 必须用文件来存储学生信息：开始运行程序时从文件中读取学生信息到链表，退出程序时将学生信息保存到文件中。
3. 数据约束：
4. 学号：整型、不能重复、从1开始依次递增、由软件自动计算产生，不能修改。
5. 姓名、专业、班级：不能为空。
6. 类别：枚举类型，分别表示本科生、研究生。
7. 性别：男或女。
8. 各种成绩：整型，取值范围为0-100。-1表示未有此项成绩。
9. 其他：至少有两层菜单。

**（四）、建议定义如下的主要数据结构**

1、枚举型学生类别：本科生、研究生

enum studentcategory { UNDERGRADUATE,POSTGRADUATE };

2、学生类型：链表结点

typedef struct node{

int id; //学号：主键，唯一

char name[10]; //姓名

char sex[3]; //性别：男或女

char specialty[20]; //专业

char classes[20]; //班级：唯一，不同班级不同

enum studentcategory category; //枚举型学生类别；本科生、研究生

int math; //高数

int english; //英语

int clanguage; //C语言

int comprehensive; //课程综合

int thesis; //毕业论文

int totalscore; //总成绩

int classrank; //班级排名

int schoolrank; //校级排名

struct node \* next;

}Student;