|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  |  | |  |  | |  |  | |  |  | |  |  | |  |  |  实验二 **[实验目的和要求]**   1. 理解数据流图的作用及其基本概念和相关要点。 2. 了解数据流图和程序流程图之间的差别和各自的特点。 3. 掌握使用Microsoft Visio 工具绘制数据流图的方法。 4. 掌握如果从问题陈述中构造数据流图的基本思路。   **[实验内容和步骤]**   1. 观看老师给出的学习资料，理解清楚相应的概念以及相关的操作步骤。 2. 根据以下问题陈述，进行数据流图建模并编写相应的数据字典。   工资计算和制表工作繁杂并易于出错，现单位希望研发一个工资发放系统，该系统能够在每个月初根据“上海小明服装厂工资管理制度”的规定以及员工上个月的具体表现，打印出一张工资发放表，便于财务人员进行工资发放。员工的具体表现情况由行政人员出具。   1. 要求每个人独立完成，并提交相应的实验报告及结果。       数据字典  名字：**工资标准**  描述：各工资的工资标准  定义：工资标准={基本工资标准+岗位工资标准+绩效/计件工资标准+生活津贴+加班工资标准}  基本工资标准=4{数字}5  岗位工资标准=3{数字}4  绩效工资标准=2{数字}4  计件工资标准=2{数字}4  生活津贴=2{数字}3  加班工资标准=2{数字}4  名字：**员工考勤信息**  描述：员工在工作时的具体表现，比如出勤天数，加班时长等  定义：员工表现={全勤天数+出勤天数+加班天数+绩效分数/计件数}  全勤天数=2{数字}2  出勤天数=1{数字}2  加班天数=1{数字}2  绩效分数=1{数字}2  计件数=1{数字}3  名字：**员工个人信息**  描述：员工的个人信息  定义：员工的个人信息={姓名+工号+岗位+级别}  姓名=2{字符}8  工号=10{字符}10  岗位=1{字符}10  级别=1{数字}1  名字：**工资发放表**  描述：需要发放每个员工总工资的工资表  定义：工资发放表={工资标准+员工表现+员工个人信息}  名字：**基本工资**  描述：员工的基本工资  定义：基本工资={月基本工资标准/全勤天数\*出勤天数}  月基本工资标准=4{数字}5  全勤天数=2{数字}2  出勤天数=1{数字}2  名字：**岗位工资**  描述：各岗位的工资  定义：岗位工资={月岗位工资标准/全勤天数\*出勤天数}  月岗位工资=3{数字}4  全勤天数=2{数字}2  出勤天数=1{数字}2  名字：**绩效工资**  描述：各员工的绩效工资  定义：绩效工资={月绩效工资\*绩效分数}  月绩效工资=2{数字}4  绩效分数=1{数字}2  名字：**生活津贴**  描述：各员工的生活津贴  定义：生活津贴={月生活津贴标准/全勤天数\*出勤天数}  月生活津贴标准=1{数字}3  全勤天数=2{数字}2  出勤天数=1{数字}2  名字：**加班工资**  描述：员工加班的工资  定义：加班工资={月基本工资标准/全勤天数\*加班天数}  月基本工资=4{数字}5  全勤天数=2{数字}2  加班天数=1{数字}2 实验三 **[实验目的和要求]**   1. 了解软件模块设计的目的和任务 2. 理解软件模块结构图的基本元素及相关概念； 3. 掌握从数据流图获取软件模块结构图的基本方法； 4. 掌握软件模块结构图的绘制方法和模块说明的编写方法。 5. 每一级结构图的分解要清晰、连贯，并将每一级的分解图放置在实验报告中。 6. 要针对最终的结构图的各个模块进行接口说明； 7. 绘制的图形清楚，排版美   **[实验内容和步骤]**   1. 根据老师对实验二数据流图的建模成果的点评并参考其他同学的工作成果，重新修订本人的“工资发放系统”的数据流图及其数据字典。 2. 采用面向数据流方法设计“工资支付系统”的软件结构并写出每个模块的详细说明。 3. 要求每个人独立完成，并提交相应的实验报告及结果。     软件结构图  模块详细说明(node+mysql+react)   1. 录入员工信息   function addEmployees(info) `  功能： 添加员工信息  说明：info 是存储员工个人信息的 json 数据，包括（name:员工姓名、birth:出生日期、gender:性别、phone:手机号、role:所在岗位、grades:现岗位等级）等  返回值：添加成功或失败  2、录入岗位和基本薪资信息  function addJobsInfo(info)  功能：添加岗位信息  说明：info 为岗位信息和就基本薪资信息的json数据，包括（role:岗位名称、grades:岗位等级、jbase\_money:岗位基本薪资、ebase\_money:该岗位月基本薪资、perfrom\_money:基本绩效工资、live\_money:生活津贴、grow\_money:岗位增幅）等  返回值：添加成功或失败  3、录入考勤信息  function addAttendanceInfo(info)  功能：添加员工考勤信息  说明：info为员工基本考勤信息的json数据，包括（work\_day;出勤天数, total\_day:本月实际天数, over\_day:加班天数, piece\_total:计件数, score:绩效得分）等  返回值:添加成功或失败  4、计算总工资  function getSalaryInfo()  功能：计算总工资  说明：(月基本薪资\*实际天数(全勤天数)\*出勤天数) + ((该岗位基本薪资 + (该员工等级 - 1 ) \* 该岗位增幅）) / 实际天数(全勤天数) \* 出勤天数) + (月绩效工资 \* (100% / 绩效得分)) + (  月生活津贴 / 实际天数(全勤天数) \* 出勤天数) + (月基本工资 / 实际天数(全勤天数) \* 加班天数) + (月基本工资 / 实际天数(全勤天数) \* 计件数) = 总工资  返回值:无  实验四  **[实验目的和要求]**   1. 了解软件模块详细设计的目的和任务； 2. 理解软件模块详细设计的基本概念和基本要点； 3. 掌握详细设计的基本方法；   **[实验内容和步骤]**   1. 根据老师在实验三中对“工资发放系统”的软件结构设计成果的点评并参考其他同学的工作成果，重新修订本人的“工资发放系统”的软件结构图及其模块说明。 2. 根据“工资发放系统”的软件结构图及其模块说明，对每个模块进行详细设计。   实验四  **[实验目的和要求]**   1. 掌握从详细设计落地到代码的基本思路和方法； 2. 理解单元测试、集成测试的基本概念和基本要点； 3. 掌握单元测试和集成测试的基本方法；   **[实验内容和步骤]**   1. 根据老师在实验四中对“工资发放系统”的软件详细设计成果的点评并参考同学的工作成果，审阅并重新修订本人的“工资发放系统”的详细设计文稿。 2. 制定编程规范（C语言） 3. 根据详细设计，采用C语言编码实现整个系统。 4. 对系统中大部分函数采用Google Test进行单元测试。 5. 对系统进行集成测试。     Jackson图 |
| **伪代码:**  计算工资 seq  获取工资标准  获取岗位工资标准  获取员工信息  获取考勤信息  程序体 iter until 循环计算各个员工结束  计算各类工资 seq  计算基本工资  计算岗位工资  计算绩效工资 seq  判断绩效得分 select 大于等于 90 分  计算大于等于 90分 seq  全额发放  计算大于等于 90分end  判断绩效得分 or 小于 90分  计算小于 90分 seq  按实际得分发放  计算小于 90分 end  判断绩效得分 end  计算绩效工资 end  计算生活津贴  计算加班工资  计算工资总和  显示员工工资计算结果 seq  显示员工工资计算结果列表  显示员工工资计算结果end  计算各类工资 end  程序体 end  选择select 保存计算结果到员工工资表  存储数据库员工工资表seq  发送请求存入员工工资表  导出员工工资excle表 seq  导出所有工资信息  导出员工工资excle表end  存储数据库员工工资表end  选择 or不保存计算结果到员工工资表  退出页面  选择end  退出系统  计算工资 end |
| 实验五  **[实验目的和要求]**   1. 掌握从详细设计落地到代码的基本思路和方法； 2. 理解单元测试、集成测试的基本概念和基本要点； 3. 掌握单元测试和集成测试的基本方法； 4. 绘制的图形清楚，排版美观 5. 算法描述清晰 6. 文档排版简洁   **[实验内容和步骤]**   1. 根据老师在实验四中对“工资发放系统”的软件详细设计成果的点评并参考同学的工作成果，审阅并重新修订本人的“工资发放系统”的详细设计文稿。 2. 制定编程规范（C语言） 3. 根据详细设计，采用C语言编码实现整个系统。 4. 对系统中大部分函数采用Google Test进行单元测试。 5. 对系统进行集成测试。 |
| 数据库：xiaominggongchang  员工表：  create table employees(  eid char(100) unique primary key comment '员工编号',  jid char(30) comment '岗位编号',  jlevel char(2) comment '岗位等级',  ename char(30) comment '员工姓名',  ebirth char(10) comment '出生日期',  egender char(2) comment '员工性别',  ephone varchar(11) comment '员工电话号码', ecreate\_time char(10) comment '入职时间',  isshow int comment '1 显示 0隐藏'  )  岗位表：  create table jobs(  jid char(30) primary key comment '岗位编号' ,  jname char(30) comment '岗位名称',  level\_total char(30) comment '该岗位的总等级',  isshow int comment '显示 1 隐藏 0'  ) |
| 基本工资表：  create table salary(  jid char(30) primary key comment '岗位编号' ,  ebase\_money float comment '基本工资',  jbase\_money float comment '岗位基本工资',  perform\_money float comment '绩效基本工资',  live\_money float comment '生活津贴基本工资',  grow\_money float comment '岗位增幅工资',  isshow int comment '1 显示 0隐藏'  )  考勤表:  create table attendance(  eid char(30) unique primary key comment '员工编号' ,  work\_day char(2) comment '出勤天数',  tota\_day char(2) comment '本月实际天数',  over\_day char(10) comment '加班天数',  piece\_total char(10) comment '计件数',  score char(3) comment '绩效得分'  isshow int comment '1 显示 0隐藏'  )  用户表:  create table admin(  aid char(30) unique primary key comment '用户id',  uname varchar(30) comment '账号',  upwd varchar(255) comment '密码',  p\_add int comment '添加权限',  p\_edit int comment '修改权限',  p\_delect int comment '删除权限',  p\_read int comment '查看权限'  isshow int comment '1 显示 0隐藏'  )  **核心代码:**  **// 登录**  exports.login = async (req, res) => {  let { uname, upwd } = req.body;  if (!uname || !upwd) {  res.jsonFail(400, "参数错误");  return;  }  getLogin({ uname, upwd: md5(upwd) }, data => {  if (data.length === 0) {  res.jsonFail(400, "账号或密码错误")  return;  }  let list = {  creator: {  uid: data[0].id,  name: data[0].uname,  add: data[0].p\_add,  edit: data[0].p\_edit,  delect: data[0].p\_delect,  read: data[0].p\_read,  },  token: setToken({ uname, uid: data[0].id })  }  return res.jsonSuccess(list)  })  }  **// 添加员工**  exports.addEmployees = (req, res) => {  const valid = valided(req.body);  if (!valid.flag) {  return res.jsonFail(400, valid.msg)  }  addEmployeesInfo({ uid: createRandomId(), ...req.body, createTime: formatDate(new Date()) }, data => {  try {  if (!data) {  return res.jsonFail(400, "请检查添加内容是否符合规范")  }  res.jsonSuccess(null)  }  catch (err) {  console.log("添加员工失败" + err);  return res.jsonFail(400, "未知错误")  }  })  }  // 添加岗位  exports.addJobsInfo = (req, res) => {  try {  let { role: name, grades: level\_total, ebase\_money, jbase\_money, perform\_money, live\_money, grow\_money } = req.body;  if (!name || !level\_total || !ebase\_money || !jbase\_money || !perform\_money || !live\_money || !grow\_money) {  return res.jsonFail(400, '参数错误')  }  addJobs({ id: createRandomId(), name, level\_total, ebase\_money, jbase\_money, perform\_money, live\_money, grow\_money }, data => {  if (data) {  res.jsonSuccess(data)  } else {  res.jsonFail(400, '添加岗位失败')  }  })  } catch (error) {  console.log("erro", error);  res.jsonFail(400, '添加失败');  }  } |
| **// 修改考勤信息**  exports.editAttendanceInfo = (req, res) => {  try {  let { work\_day, total\_day, over\_day, piece\_total, score, uid } = req.body;  let flag = true;  for (const key in req.body) {  if (!['work\_day', 'total\_day', 'over\_day', 'piece\_total', 'score', 'uid'].includes(key)) {  flag = false;  }  }  if (!flag) {  return res.jsonFail(400, '参数错误')  }  editAttendance({ work\_day, total\_day, over\_day, piece\_total, score: score || 0, uid }, data => {  if (data && data.affectedRows > 0) {  res.jsonSuccess(null);  } else {  res.jsonFail(400, "修改失败")  }  })  }  catch (err) {  console.log("修改考勤信息失败", err);  res.jsonFail(400, "修改失败")  }  }  **// 计算薪资**  exports.getSalaryInfo = (req, res) => {  try {  let { page, limit } = req.query  if (!page || !limit) {  page = 1, limit = 15  }  page = parseInt(page)  limit = parseInt(limit)  getSalaryList({ page: (page - 1) \* limit, limit }, data => {  res.jsonSuccess({  data: data?.map(v => {  const baseMoney = Number(((v.ebase\_money / v.total\_day) \* v.work\_day).toFixed(2));  const jobMoney = Number(((v.jbase\_money / v.total\_day) \* v.work\_day).toFixed(2));  const performMoney = Number((v.perform\_money \* (v.score / 100) + (Number(v.jlevel) - 1) \* v.grow\_money).toFixed(2));  const liveMoney = Number(((v.live\_money / v.total\_day) \* v.work\_day).toFixed(2));  const overMoney = Number(((v.ebase\_money / v.total\_day) \* v.over\_day).toFixed(2));  const totalMoney = Number((baseMoney + jobMoney + performMoney + liveMoney + overMoney).toFixed(2));  return {  uid: v.eid,  name: v.ename,  ebase\_money: v.ebase\_money,  jbase\_money: v.jbase\_money,  perform\_money: v.perform\_money,  live\_money: v.live\_money,  grow\_money: v.grow\_money,  work\_day: v.work\_day,  total\_day: v.total\_day,  over\_day: v.over\_day,  piece\_total: v.piece\_total,  jlevel: v.jlevel,  score: v.score,  baseMoney,  jobMoney,  performMoney,  liveMoney,  overMoney,  totalMoney  }  }  ),  page,  limit,  total: 10,  })  })  }  catch (err) {  console.log("获取考勤信息失败", err);  res.jsonFail(400, "获取失败")  }  } |