

详细设计说明书



课程名称 软件工程

题目名称 需求分析报告

任课教师 杨 枨

组 号： G18

组 长： 谢子文

组 员： 梁泽生

组 员： 黄 馨

2020 年12月 9 日

目录

[1 引言 6](#_Toc58673537)

[1.1 编写目的 6](#_Toc58673538)

[1.2 背景 6](#_Toc58673539)

[1.3 定义 7](#_Toc58673540)

[1.4 参考资料 7](#_Toc58673541)

[1.5 版本历史 7](#_Toc58673542)

[2 程序系统的结构 8](#_Toc58673543)

[3 计时功能设计说明 8](#_Toc58673544)

[3.1 程序描述 8](#_Toc58673545)

[3.2 功能 8](#_Toc58673546)

[3.3 性能 10](#_Toc58673547)

[3.3.1 精度 10](#_Toc58673548)

[3.3.2 灵活性 10](#_Toc58673549)

[3.3.3 时间性 10](#_Toc58673550)

[3.4 输入项 10](#_Toc58673551)

[3.5 输出项 11](#_Toc58673552)

[3.6 算法 11](#_Toc58673553)

[3.6.1 计时模块 11](#_Toc58673554)

[3.6.2 专注组合模块 12](#_Toc58673555)

[3.7 流程逻辑 13](#_Toc58673556)

[3.8 接口 13](#_Toc58673557)

[3.9 存储分配 13](#_Toc58673558)

[3.10 注释设计 13](#_Toc58673559)

[3.11 限制条件 13](#_Toc58673560)

[3.12 测试计划 13](#_Toc58673561)

[3.13 尚未解决的问题 14](#_Toc58673562)

[4 蒸笼设计说明 14](#_Toc58673563)

[4.1 程序描述 14](#_Toc58673564)

[4.2 功能 14](#_Toc58673565)

[4.3 性能 15](#_Toc58673566)

[4.3.1 精度 16](#_Toc58673567)

[4.3.2 灵活性 16](#_Toc58673568)

[4.3.3 时间性 16](#_Toc58673569)

[4.4 输入项 16](#_Toc58673570)

[4.5 输出项 17](#_Toc58673571)

[4.6 算法 17](#_Toc58673572)

[4.7 流程逻辑 18](#_Toc58673573)

[4.8 接口 18](#_Toc58673574)

[4.9 存储分配 18](#_Toc58673575)

[4.10 注释设计 18](#_Toc58673576)

[4.11 限制条件 18](#_Toc58673577)

[4.12 测试计划 18](#_Toc58673578)

[4.13 尚未解决的问题 19](#_Toc58673579)

[5 时间轴设计说明 19](#_Toc58673580)

[5.1 程序描述 19](#_Toc58673581)

[5.2 功能 19](#_Toc58673582)

[5.3 性能 20](#_Toc58673583)

[5.3.1 精度 21](#_Toc58673584)

[5.3.2 灵活性 21](#_Toc58673585)

[5.3.3 时间性 21](#_Toc58673586)

[5.4 输入项 21](#_Toc58673587)

[5.5 输出项 22](#_Toc58673588)

[5.6 算法 22](#_Toc58673589)

[5.7 流程逻辑 23](#_Toc58673590)

[5.8 接口 23](#_Toc58673591)

[5.9 存储分配 23](#_Toc58673592)

[5.10 注释设计 23](#_Toc58673593)

[5.11 限制条件 23](#_Toc58673594)

[5.12 测试计划 24](#_Toc58673595)

[5.13 尚未解决的问题 24](#_Toc58673596)

[6 排行榜设计说明 24](#_Toc58673597)

[6.1 程序描述 24](#_Toc58673598)

[6.2 功能 24](#_Toc58673599)

[6.3 性能 25](#_Toc58673600)

[6.3.1 精度 26](#_Toc58673601)

[6.3.2 灵活性 26](#_Toc58673602)

[6.3.3 时间性 26](#_Toc58673603)

[6.4 输入项 26](#_Toc58673604)

[6.5 输出项 27](#_Toc58673605)

[6.6 算法 27](#_Toc58673606)

[6.7 流程逻辑 28](#_Toc58673607)

[6.8 接口 28](#_Toc58673608)

[6.9 存储分配 28](#_Toc58673609)

[6.10 注释设计 28](#_Toc58673610)

[6.11 限制条件 28](#_Toc58673611)

[6.12 测试计划 28](#_Toc58673612)

[6.13 尚未解决的问题 29](#_Toc58673613)

[7 成就设计说明 29](#_Toc58673614)

[7.1 程序描述 29](#_Toc58673615)

[7.2 功能 29](#_Toc58673616)

[7.3 性能 30](#_Toc58673617)

[7.3.1 精度 31](#_Toc58673618)

[7.3.2 灵活性 31](#_Toc58673619)

[7.3.3 时间性 31](#_Toc58673620)

[7.4 输入项 31](#_Toc58673621)

[7.5 输出项 32](#_Toc58673622)

[7.6 算法 32](#_Toc58673623)

[7.7 流程逻辑 33](#_Toc58673624)

[7.8 接口 33](#_Toc58673625)

[7.9 存储分配 33](#_Toc58673626)

[7.10 注释设计 33](#_Toc58673627)

[7.11 限制条件 33](#_Toc58673628)

[7.12 测试计划 33](#_Toc58673629)

[7.13 尚未解决的问题 34](#_Toc58673630)

[8 商店设计说明 34](#_Toc58673631)

[8.1 程序描述 34](#_Toc58673632)

[8.2 功能 34](#_Toc58673633)

[8.3 性能 35](#_Toc58673634)

[8.3.1 精度 36](#_Toc58673635)

[8.3.2 灵活性 36](#_Toc58673636)

[8.3.3 时间性 36](#_Toc58673637)

[8.4 输入项 36](#_Toc58673638)

[8.5 输出项 37](#_Toc58673639)

[8.6 算法 37](#_Toc58673640)

[8.7 流程逻辑 38](#_Toc58673641)

[8.8 接口 38](#_Toc58673642)

[8.9 存储分配 38](#_Toc58673643)

[8.10 注释设计 39](#_Toc58673644)

[8.11 限制条件 39](#_Toc58673645)

[8.12 测试计划 39](#_Toc58673646)

[8.13 尚未解决的问题 39](#_Toc58673647)

[9 同桌设计说明 39](#_Toc58673648)

[9.1 程序描述 39](#_Toc58673649)

[9.2 功能 40](#_Toc58673650)

[9.3 性能 41](#_Toc58673651)

[9.3.1 精度 42](#_Toc58673652)

[9.3.2 灵活性 42](#_Toc58673653)

[9.3.3 时间性 42](#_Toc58673654)

[9.4 输入项 42](#_Toc58673655)

[9.5 输出项 43](#_Toc58673656)

[9.6 算法 43](#_Toc58673657)

[9.7 流程逻辑 44](#_Toc58673658)

[9.8 接口 45](#_Toc58673659)

[9.9 存储分配 45](#_Toc58673660)

[9.10 注释设计 45](#_Toc58673661)

[9.11 限制条件 45](#_Toc58673662)

[9.12 测试计划 45](#_Toc58673663)

[9.13 尚未解决的问题 45](#_Toc58673664)

[10 个人中心设计说明 46](#_Toc58673665)

[10.1 程序描述 46](#_Toc58673666)

[10.2 功能 46](#_Toc58673667)

[10.3 性能 47](#_Toc58673668)

[10.3.1 精度 48](#_Toc58673669)

[10.3.2 灵活性 48](#_Toc58673670)

[10.3.3 时间性 48](#_Toc58673671)

[10.4 输入项 48](#_Toc58673672)

[10.5 输出项 49](#_Toc58673673)

[10.6 算法 49](#_Toc58673674)

[10.7 流程逻辑 50](#_Toc58673675)

[10.8 接口 50](#_Toc58673676)

[10.9 存储分配 50](#_Toc58673677)

[10.10 注释设计 50](#_Toc58673678)

[10.11 限制条件 50](#_Toc58673679)

[10.12 测试计划 50](#_Toc58673680)

[10.13 尚未解决的问题 51](#_Toc58673681)

[11 管理员界面设计说明 51](#_Toc58673682)

[11.1 程序描述 51](#_Toc58673683)

[11.2 功能 51](#_Toc58673684)

[11.3 性能 52](#_Toc58673685)

[11.3.1 精度 53](#_Toc58673686)

[11.3.2 灵活性 53](#_Toc58673687)

[11.3.3 时间性 53](#_Toc58673688)

[11.4 输入项 53](#_Toc58673689)

[11.5 输出项 54](#_Toc58673690)

[11.6 算法 54](#_Toc58673691)

[11.7 流程逻辑 55](#_Toc58673692)

[11.8 接口 55](#_Toc58673693)

[11.9 存储分配 55](#_Toc58673694)

[11.10 注释设计 56](#_Toc58673695)

[11.11 限制条件 56](#_Toc58673696)

[11.12 测试计划 56](#_Toc58673697)

[11.13 尚未解决的问题 56](#_Toc58673698)

**详细设计说明书**

# 引言

## 编写目的

说明编写这份详细设计说明书的目的，指出预期的读者。

在前一阶段的需求分析中，已经解决了该系统在需求方面以及各个部门之间的数据传输流程等问题,包括把系统分解成几个模块以及各个模块之间的数据传递,数据结构模块结构等问题,在以下的详细设计阶段中将对本系统在本阶段中的所有环节的详细设计进行说明.

在本阶段中,确定如何具体的实现所要求的系统,从而使其在以后的设计中可以直接被翻译成用某种特定的语言所编写的程序,主要工作有:根据在需求分析中所描述的数据,功能,运行,性能需求,以及数据流程,总体结构,和模块设计,设计软件系统的结构设计和逐个模块的程序描述,包括:包括各个模块的功能，性能，输入，输出，算法，程序逻辑，接口等等。

预期的读者：杨枨老师、各个开发者成员（即小组成员）、部分用户。

## 背景

说明：

1. 待开发软件系统的名称；
2. 本项目的任务提出者、开发者、用户和运行该程序系统的计算中心。
3. 待开发软件系统的名称：

专注包子 微信小程序

1. 此项目的任务提出者：

杨枨教授

1. 此项目的开发者：

谢子文、黄馨、梁泽生

1. 此项目的用户：

自制力不强，想要专注学习和工作的人们。

1. 以下列出目标用户：

杨枨教授、软工1802陈骁、法学1802王新、建筑1801王宇键

1. 将运行该软件的计算站（中心）：

腾讯云服务器1核2G

50G云数据库MySQL

GitHub云端版本配置管理

## 定义

列出本文件中用到专门术语的定义和外文首字母组词的原词组。

## 参考资料

列出有关的参考资料，如：

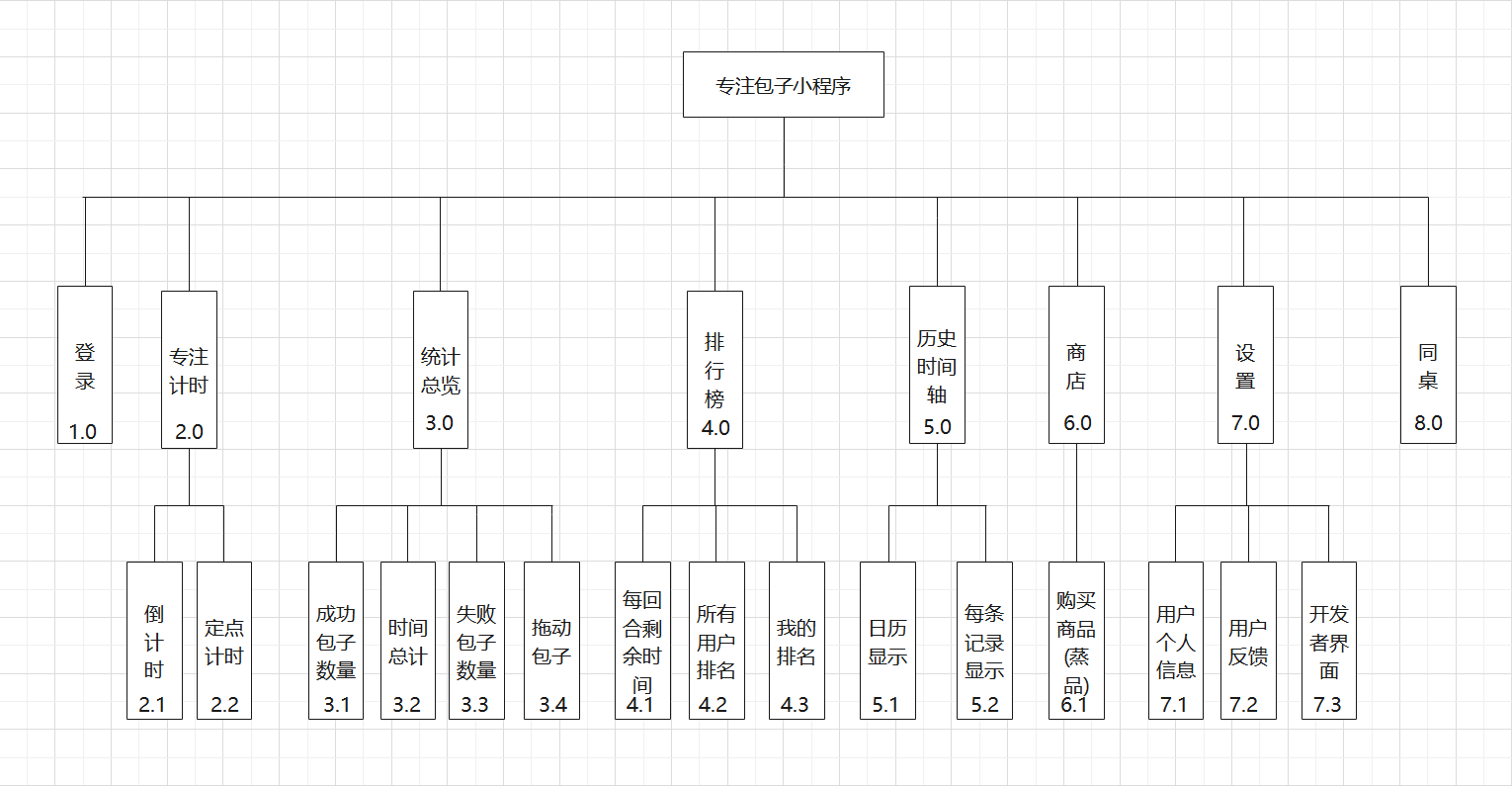
1. 本项目的经核准的计划任务书或合同、上级机关的批文；
2. 属于本项目的其他已发表的文件；
3. 本文件中各处引用到的文件资料，包括所要用到的软件开发标准。列出这些文件的标题、文件编号、发表日期和出版单位，说明能够取得这些文件的来源。

## 版本历史

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 版本修订日期 | 版本修改人 | 是否通过审核 |
| 0.1.0 | 2020/11/29 | 谢子文 | 是 |
| 0.1.1 | 2020/12/01 | 谢子文、黄馨、梁泽生 | 是 |
| 0.1.2 | 2020/12/06 | 谢子文、黄馨、梁泽生 | 是 |
| 0.1.3 | 2020/12/08 | 谢子文、黄馨、梁泽生 | 是 |
| 0.1.4 | 2020/12/09 | 黄馨 | 是 |
| 0.1.5 | 2020/12/10 | 梁泽生 | 是 |
| 0.1.6 | 2020/12/10 | 黄馨 | 是 |
| 0.1.7 | 2021/1/3 | 黄馨 | 是 |
| 0.1.8 | 2021/01/20 | 谢子文 | 是 |

# 程序系统的结构

用一系列图表列出本程序系统内的每个程序（包括每个模块和子程序）的名称、标识符和它们之间 的层次结构关系。



# 计时功能设计说明

从本章开始，逐个地给出各个层次中的每个程序的设计考虑。以下给出的提纲是针对一般情况的。对于一个具体的模块，尤其是层次比较低的模块或子程序，其很多条目的内容往往与它所隶属的上一层 模块的对应条目的内容相同，在这种情况下，只要简单地说明这一点即可。

## 程序描述

给出对该程序的简要描述，主要说明安排设计本程序的目的意义，并且，还要说明本程序的特点（如 是常驻内存还是非常驻？是否子程序？是可重人的还是不可重人的？有无覆盖要求？是顺序处理还是并发处理等）。

## 功能

说明该程序应具有的功能，可采用IPO图（即输入一处理一输出图）的形式。

以IPO图形式表现输入输出：

图表待修改

|  |  |  |  |
| --- | --- | --- | --- |
| 功能名称 | 输入input | 处理过程process | 输出output |
| 用户登录 | 用户点击小程序并且授权登陆。 | 系统服务器检测到登录情况并且允许登录操作。 | 进入小程序初始页面。 |
| 设置 | 点击 | 读取信号并切换页面 | 进入设置页面 |
| 设定专注时长 | 想要专注多长时间 | 经过汇编语言处理 | 生成一个番茄钟 |
| 开始（倒）番茄钟计时 | 发出专注番茄钟开始的信号 | 读取信号并判断 | 开始蒸包子（专注），番茄钟开始计时 |
| 蒸包子（专注）中 | 无 | 判断是否在专注，是否在不玩手机 | 蒸包子中，随着时间长度的变化，包子的形态也会随着改变 |
| 查看排行榜 | 点击排行榜的按钮 | 按照数据库中储存的用户专注时长来进行排序 | 排行榜上各个用户的排名 |
| 个性化商店 | 点击商店 | 加载数据库存储的商品信息 | 各个商品可在界面上展示出来 |
| 查看总览计时 | 点击统计总览 | 读取数据库中用户的专注信息 | 显示用户的蒸笼（也就是专注信息） |
| 查看时间历程（也叫历史时间轴） | 点击时间历程 | 加载日历，并从数据库中提取专注信息记录 | 界面上半部分是日历，日历下面是今日的专注记录 |
| 退出小程序 | 点击退出 | 系统检测到退出操作并允许执行 | 退出到微信界面 |

## 性能

说明对该程序的全部性能要求，包括对精度、灵活性和时间特性的要求。

### 精度

在进行向数据库提取所要信息时，只要输入相关的关键字，就能查找出相对应的信息；向数据库输入信息时，要确保输入的信息准确的并且数据类型要符合定义的数据类型。

### 灵活性

a．操作方式上的变化：暂无

b．运行环境的变化：暂无

c．同其他软件的接口的变化：暂无

d．精度和有效时限的变化：精度在必须调整的条件下，有效时限则依据现实的测试情况允许稍大范围的变化。

e．计划的变化或改进： 工作时间安排会存在必然的浮动，这部分要协同分工和组员一同来进行商定，前期的计划可以稍微有些变动，后期的安排尽量按照计划执行。

### 时间性

a启动该系统后，系统就能运行。

b.更新处理时间：由系统运行状态来决定

c.数据的转换和传送时间：能够在15秒钟内完成

## 输入项

给出对每一个输入项的特性，包括名称、标识、数据的类型和格式、数据值的有效范围、输入的方式。数量和频度、输入媒体、输入数据的来源和安全保密条件等等。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 格式 | 有效范围 | 输入方式 |
| 选择计时模式 |  |  |  |  |  |
| 设定时间 | 2.6.0 |  | 按钮 | 0和1 | 按钮 |
| 选择蒸品 | 2.7.0 |  | 按钮 | 0和1 | 按钮 |
| 开始按钮 | 2.5.0 |  | 按钮 | 0和1 | 按钮 |
| 菜单按钮 |  |  |  |  |  |

## 输出项

给出对每一个输出项的特性，包括名称、标识、数据的类型和格式，数据值的有效范围，输出的形式、数量和频度，输出媒体、对输出图形及符号的说明、安全保密条件等等。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 格式 | 有效范围 | 输出方式 |
| 做好的蒸品信息 |  |  |  |  |  |

## 算法

startTiming\_0: function () {

    var that = this;

    that.setData({

        isTiming: 1,

        totalsecond: that.data.timesetArray\_num[that.data.timeindex] \* 60 - 1

    })

    var totalsecond = that.data.totalsecond

    var minute = that.data.timesetArray\_num[that.data.timeindex] //分钟数

    var second = totalsecond % 60 //代表的是总时间去掉分钟的秒数零头

    /\* 设定一个定时器。按照指定的周期（以毫秒计）来执行注册的回调函数 \*/

    var i = setInterval(function () {

        //每次定时器执行时，更新变量

        var totalsecond = that.data.totalsecond

        var isBindBtnStop = that.data.isBindBtnStop

        second = totalsecond % 60

        minute = (totalsecond - second) / 60

        totalsecond--

        //判断 是继续计时并更新界面 还是停止计时

        if (totalsecond >= 0 && (isBindBtnStop == 0)) {

            that.setData({

                totalsecond: totalsecond,

                UI\_Timeset: that.zeroFill(minute.toString(), 2) + ":" + that.zeroFill(second.toString(), 2)

            })

        } else {

            clearInterval(i)

            that.setData({

                UI\_Timeset: that.data.timesetArray[that.data.timeindex],

                isTiming: 0

            })

            //用户主动取消计时

            if (isBindBtnStop == 1) {

                console.log("弹窗告知用户专注失败")

                that.setData({

                    isBindBtnStop: 0

                }) //刷新状态

                // 向event表中插入此次计时事件

                app.sqlOpreate("insert into event(user\_id,event\_discribe,event\_time,event\_date) values(" + app.globalData.currentUserInfo.user\_id + ",'失败',0,'" + that.timeStampGet() + "')")

                // 向steamer中插入此次计时事件

                app.sqlOpreate("update steamer set steamer\_main\_count= steamer\_main\_count + 1, steamer\_dietree\_count = steamer\_dietree\_count + 1 where user\_id = " + app.globalData.currentUserInfo.user\_id)

            }

            //用户计时自然结束

            else {

                console.log("弹窗告知用户专注成功")

                // 向event表中插入此次计时事件

                app.sqlOpreate("insert into event(user\_id,event\_discribe,event\_time,event\_date) values(" + app.globalData.currentUserInfo.user\_id + ",'成功'," + that.data.timesetArray\_num[that.data.timeindex] + ",'" + that.timeStampGet() + "')")

                // 向steamer中插入此次计时事件

                app.sqlOpreate("update steamer set steamer\_main\_count= steamer\_main\_count + 1, steamer\_livetree\_count = steamer\_livetree\_count + 1,steamer\_time\_count =  steamer\_time\_count + " + that.data.timesetArray\_num[that.data.timeindex] + " where user\_id = " + app.globalData.currentUserInfo.user\_id)

                // 加钱

                app.sqlOpreate("update user set user\_money = user\_money + " + that.data.timesetArray\_num[that.data.timeindex] + " where user\_id = " + app.globalData.currentUserInfo.user\_id)

                app.globalData.currentUserInfo.user\_money += that.data.timesetArray\_num[that.data.timeindex]

            }

        }

    }, 10)

},

## 流程逻辑

用图表（例如流程图、判定表等）辅以必要的说明来表示本程序的逻辑流程。

插入图片

## 接口

用图的形式说明本程序所隶属的上一层模块及隶属于本程序的下一层模块、子程序，说明参数赋值和调用方式，说明与本程序相直接关联的数据结构（数据库、数据文卷）。

## 存储分配

根据需要，说明本程序的存储分配。

## 注释设计

说明准备在本程序中安排的注释，如：

1. 加在模块首部的注释；
2. 加在各分枝点处的注释；
3. 对各变量的功能、范围、缺省条件等所加的注释；
4. 对使用的逻辑所加的注释等等。

## 限制条件

说明本程序运行中所受到的限制条件。

## 测试计划

说明对本程序进行单体测试的计划，包括对测试的技术要求、输入数据、预期结果、进度安排、人员职责、设备条件驱动程序及桩模块等的规定。

## 尚未解决的问题

说明在本程序的设计中尚未解决而设计者认为在软件完成之前应解决的问题。

# 蒸笼设计说明

从本章开始，逐个地给出各个层次中的每个程序的设计考虑。以下给出的提纲是针对一般情况的。对于一个具体的模块，尤其是层次比较低的模块或子程序，其很多条目的内容往往与它所隶属的上一层 模块的对应条目的内容相同，在这种情况下，只要简单地说明这一点即可。

## 程序描述

给出对该程序的简要描述，主要说明安排设计本程序的目的意义，并且，还要说明本程序的特点（如 是常驻内存还是非常驻？是否子程序？是可重人的还是不可重人的？有无覆盖要求？是顺序处理还是并发处理等）。

## 功能

说明该程序应具有的功能，可采用IPO图（即输入一处理一输出图）的形式。

以IPO图形式表现输入输出：

图表待修改

|  |  |  |  |
| --- | --- | --- | --- |
| 功能名称 | 输入input | 处理过程process | 输出output |
| 用户登录 | 用户点击小程序并且授权登陆。 | 系统服务器检测到登录情况并且允许登录操作。 | 进入小程序初始页面。 |
| 设置 | 点击 | 读取信号并切换页面 | 进入设置页面 |
| 设定专注时长 | 想要专注多长时间 | 经过汇编语言处理 | 生成一个番茄钟 |
| 开始（倒）番茄钟计时 | 发出专注番茄钟开始的信号 | 读取信号并判断 | 开始蒸包子（专注），番茄钟开始计时 |
| 蒸包子（专注）中 | 无 | 判断是否在专注，是否在不玩手机 | 蒸包子中，随着时间长度的变化，包子的形态也会随着改变 |
| 查看排行榜 | 点击排行榜的按钮 | 按照数据库中储存的用户专注时长来进行排序 | 排行榜上各个用户的排名 |
| 预设任务番茄钟 | 输入预设事项和相关专注时长 | 添加相关的事项和时长到数据库中存储 | 预设事项形成 |
| 个性化商店 | 点击商店 | 加载数据库存储的商品信息 | 各个商品可在界面上展示出来 |
| 查看总览计时 | 点击统计总览 | 读取数据库中用户的专注信息 | 显示用户的蒸笼（也就是专注信息） |
| 查看时间历程（也叫历史时间轴） | 点击时间历程 | 加载日历，并从数据库中提取专注信息记录 | 界面上半部分是日历，日历下面是今日的专注记录 |
| 警告用户 | 用户在专注时期退出小程序界面 | 系统检测到界面切换出去的操作 | 弹出提示框，警告用户不专注了，请尽快回来 |
| 退出小程序 | 点击退出 | 系统检测到退出操作并允许执行 | 退出到微信界面 |

## 性能

说明对该程序的全部性能要求，包括对精度、灵活性和时间特性的要求。

### 精度

在进行向数据库提取所要信息时，只要输入相关的关键字，就能查找出相对应的信息；向数据库输入信息时，要确保输入的信息准确的并且数据类型要符合定义的数据类型。

### 灵活性

a．操作方式上的变化：暂无

b．运行环境的变化：暂无

c．同其他软件的接口的变化：暂无

d．精度和有效时限的变化：精度在必须调整的条件下，有效时限则依据现实的测试情况允许稍大范围的变化。

e．计划的变化或改进： 工作时间安排会存在必然的浮动，这部分要协同分工和组员一同来进行商定，前期的计划可以稍微有些变动，后期的安排尽量按照计划执行。

### 时间性

a启动该系统后，系统就能运行。

b.更新处理时间：由系统运行状态来决定

c.数据的转换和传送时间：能够在15秒钟内完成

## 输入项

给出对每一个输入项的特性，包括名称、标识、数据的类型和格式、数据值的有效范围、输入的方式。数量和频度、输入媒体、输入数据的来源和安全保密条件等等。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 格式 | 有效范围 | 输入方式 |
| 选择月份 |  |  |  |  |  |
| 蒸品图片左右滑动 | 2.6.0 |  | 按钮 | 0和1 | 按钮 |
| 上下滑动 | 2.7.0 |  | 按钮 | 0和1 | 按钮 |

## 输出项

给出对每一个输出项的特性，包括名称、标识、数据的类型和格式，数据值的有效范围，输出的形式、数量和频度，输出媒体、对输出图形及符号的说明、安全保密条件等等。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 格式 | 有效范围 | 输出方式 |
| 选中的月份 |  |  |  |  |  |
| 显示蒸品大概的个数 | 2.6.0 |  | 按钮 | 0和1 | 按钮 |
| 显示总专注时间 | 2.7.0 |  | 按钮 | 0和1 | 按钮 |
| 显示完成个数 | 2.5.0 |  | 按钮 | 0和1 | 按钮 |
| 显示失败个数 |  |  |  |  |  |
| 显示图表 |  |  |  |  |  |

## 算法

var that = this;

//获取蒸笼的数据

wx.cloud.callFunction({

    name: "mysqlOpreate",

    data: {

        sql: "select steamer\_time\_count,steamer\_livetree\_count,steamer\_dietree\_count from steamer where user\_id = " + app.globalData.currentUserInfo.user\_id

    },

    success: function (res) {

        console.log(res)

        that.setData({

            totalTime: res.result[0].steamer\_time\_count, // 总时间

            completeNum: res.result[0].steamer\_livetree\_count, // 完成数

            failedNum: res.result[0].steamer\_dietree\_count // 失败数

        })

    },

    fail: console.error

})

//获取蒸笼的数据

wx.cloud.callFunction({

    name: "mysqlOpreate",

    data: {

        sql: "select event\_time,event\_date from event where user\_id = " + app.globalData.currentUserInfo.user\_id

    },

    success: function (res) {

        if (res) {

            console.log(res)

            var length = res.result.length //循环控制

            var lengthC = length //更改长度

            for (var i = 0; i < length; i++) {

                console.log(res.result[i].event\_date.substring(0, 7))

                //最后一个

                if (i == length - 1 && res.result[i].event\_date.substring(0, 7) != that.data.date) {

                    lengthC--

                } else if (res.result[i].event\_date.substring(0, 7) != that.data.date) {

                    res.result[i] = res.result[i + 1]

                    lengthC--

                }

            }

            res.result.length = lengthC

            console.log(res)

            that.setData({

                bunsList: res.result,

                bunsList\_length: res.result.length

            })

        }

    },

    fail: console.error

})

},

## 流程逻辑

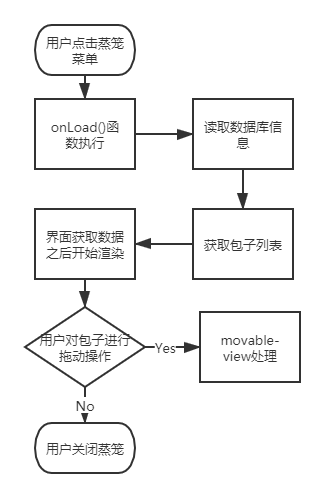


图 1 蒸笼模块流程图

## 接口

用图的形式说明本程序所隶属的上一层模块及隶属于本程序的下一层模块、子程序，说明参数赋值和调用方式，说明与本程序相直接关联的数据结构（数据库、数据文卷）。

## 存储分配

根据需要，说明本程序的存储分配。

## 注释设计

说明准备在本程序中安排的注释，如：

1. 加在模块首部的注释；
2. 加在各分枝点处的注释；
3. 对各变量的功能、范围、缺省条件等所加的注释；
4. 对使用的逻辑所加的注释等等。

## 限制条件

说明本程序运行中所受到的限制条件。

## 测试计划

说明对本程序进行单体测试的计划，包括对测试的技术要求、输入数据、预期结果、进度安排、人员职责、设备条件驱动程序及桩模块等的规定。

## 尚未解决的问题

说明在本程序的设计中尚未解决而设计者认为在软件完成之前应解决的问题。

# 时间轴设计说明

从本章开始，逐个地给出各个层次中的每个程序的设计考虑。以下给出的提纲是针对一般情况的。对于一个具体的模块，尤其是层次比较低的模块或子程序，其很多条目的内容往往与它所隶属的上一层 模块的对应条目的内容相同，在这种情况下，只要简单地说明这一点即可。

## 程序描述

给出对该程序的简要描述，主要说明安排设计本程序的目的意义，并且，还要说明本程序的特点（如 是常驻内存还是非常驻？是否子程序？是可重人的还是不可重人的？有无覆盖要求？是顺序处理还是并发处理等）。

## 功能

说明该程序应具有的功能，可采用IPO图（即输入一处理一输出图）的形式。

以IPO图形式表现输入输出：

图表待修改

|  |  |  |  |
| --- | --- | --- | --- |
| 功能名称 | 输入input | 处理过程process | 输出output |
| 用户登录 | 用户点击小程序并且授权登陆。 | 系统服务器检测到登录情况并且允许登录操作。 | 进入小程序初始页面。 |
| 设置 | 点击 | 读取信号并切换页面 | 进入设置页面 |
| 设定专注时长 | 想要专注多长时间 | 经过汇编语言处理 | 生成一个番茄钟 |
| 开始（倒）番茄钟计时 | 发出专注番茄钟开始的信号 | 读取信号并判断 | 开始蒸包子（专注），番茄钟开始计时 |
| 蒸包子（专注）中 | 无 | 判断是否在专注，是否在不玩手机 | 蒸包子中，随着时间长度的变化，包子的形态也会随着改变 |
| 查看排行榜 | 点击排行榜的按钮 | 按照数据库中储存的用户专注时长来进行排序 | 排行榜上各个用户的排名 |
| 预设任务番茄钟 | 输入预设事项和相关专注时长 | 添加相关的事项和时长到数据库中存储 | 预设事项形成 |
| 个性化商店 | 点击商店 | 加载数据库存储的商品信息 | 各个商品可在界面上展示出来 |
| 查看总览计时 | 点击统计总览 | 读取数据库中用户的专注信息 | 显示用户的蒸笼（也就是专注信息） |
| 查看时间历程（也叫历史时间轴） | 点击时间历程 | 加载日历，并从数据库中提取专注信息记录 | 界面上半部分是日历，日历下面是今日的专注记录 |
| 警告用户 | 用户在专注时期退出小程序界面 | 系统检测到界面切换出去的操作 | 弹出提示框，警告用户不专注了，请尽快回来 |
| 退出小程序 | 点击退出 | 系统检测到退出操作并允许执行 | 退出到微信界面 |

## 性能

说明对该程序的全部性能要求，包括对精度、灵活性和时间特性的要求。

### 精度

在进行向数据库提取所要信息时，只要输入相关的关键字，就能查找出相对应的信息；向数据库输入信息时，要确保输入的信息准确的并且数据类型要符合定义的数据类型。

### 灵活性

a．操作方式上的变化：暂无

b．运行环境的变化：暂无

c．同其他软件的接口的变化：暂无

d．精度和有效时限的变化：精度在必须调整的条件下，有效时限则依据现实的测试情况允许稍大范围的变化。

e．计划的变化或改进： 工作时间安排会存在必然的浮动，这部分要协同分工和组员一同来进行商定，前期的计划可以稍微有些变动，后期的安排尽量按照计划执行。

### 时间性

a启动该系统后，系统就能运行。

b.更新处理时间：由系统运行状态来决定

c.数据的转换和传送时间：能够在15秒钟内完成

## 输入项

给出对每一个输入项的特性，包括名称、标识、数据的类型和格式、数据值的有效范围、输入的方式。数量和频度、输入媒体、输入数据的来源和安全保密条件等等。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 格式 | 有效范围 | 输入方式 |
| 用户选择日期 |  |  |  |  |  |
| 上下滑动 | 2.7.0 |  | 按钮 | 0和1 | 按钮 |
| 补加记录 |  |  |  |  |  |

## 输出项

给出对每一个输出项的特性，包括名称、标识、数据的类型和格式，数据值的有效范围，输出的形式、数量和频度，输出媒体、对输出图形及符号的说明、安全保密条件等等。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 格式 | 有效范围 | 输出方式 |
| 显示被选中的日期 |  |  |  |  |  |
| 显示记录的时间日期 | 2.6.0 |  | 按钮 | 0和1 | 按钮 |
| 显示记录完成的蒸品个数 | 2.7.0 |  | 按钮 | 0和1 | 按钮 |

## 算法

var that = this

wx.cloud.callFunction({

    name: "mysqlOpreate",

    data: {

        sql: "select event\_discribe,event\_time,event\_date from event where user\_id = " + app.globalData.currentUserInfo.user\_id

    },

    success: function (res) {

        console.log(res)

        that.setData({

            recordList: res.result

        })

        console.log(that.data.recordList)

    },

    fail: console.error

})

## 流程逻辑

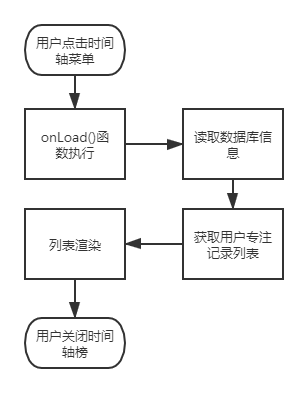


图 2 时间轴模块流程图

## 接口

用图的形式说明本程序所隶属的上一层模块及隶属于本程序的下一层模块、子程序，说明参数赋值和调用方式，说明与本程序相直接关联的数据结构（数据库、数据文卷）。

## 存储分配

根据需要，说明本程序的存储分配。

## 注释设计

说明准备在本程序中安排的注释，如：

1. 加在模块首部的注释；
2. 加在各分枝点处的注释；
3. 对各变量的功能、范围、缺省条件等所加的注释；
4. 对使用的逻辑所加的注释等等。

## 限制条件

说明本程序运行中所受到的限制条件。

## 测试计划

说明对本程序进行单体测试的计划，包括对测试的技术要求、输入数据、预期结果、进度安排、人员职责、设备条件驱动程序及桩模块等的规定。

## 尚未解决的问题

说明在本程序的设计中尚未解决而设计者认为在软件完成之前应解决的问题。

# 排行榜设计说明

从本章开始，逐个地给出各个层次中的每个程序的设计考虑。以下给出的提纲是针对一般情况的。对于一个具体的模块，尤其是层次比较低的模块或子程序，其很多条目的内容往往与它所隶属的上一层 模块的对应条目的内容相同，在这种情况下，只要简单地说明这一点即可。

## 程序描述

给出对该程序的简要描述，主要说明安排设计本程序的目的意义，并且，还要说明本程序的特点（如 是常驻内存还是非常驻？是否子程序？是可重人的还是不可重人的？有无覆盖要求？是顺序处理还是并发处理等）。

## 功能

说明该程序应具有的功能，可采用IPO图（即输入一处理一输出图）的形式。

以IPO图形式表现输入输出：

图表待修改

|  |  |  |  |
| --- | --- | --- | --- |
| 功能名称 | 输入input | 处理过程process | 输出output |
| 用户登录 | 用户点击小程序并且授权登陆。 | 系统服务器检测到登录情况并且允许登录操作。 | 进入小程序初始页面。 |
| 设置 | 点击 | 读取信号并切换页面 | 进入设置页面 |
| 设定专注时长 | 想要专注多长时间 | 经过汇编语言处理 | 生成一个番茄钟 |
| 开始（倒）番茄钟计时 | 发出专注番茄钟开始的信号 | 读取信号并判断 | 开始蒸包子（专注），番茄钟开始计时 |
| 蒸包子（专注）中 | 无 | 判断是否在专注，是否在不玩手机 | 蒸包子中，随着时间长度的变化，包子的形态也会随着改变 |
| 查看排行榜 | 点击排行榜的按钮 | 按照数据库中储存的用户专注时长来进行排序 | 排行榜上各个用户的排名 |
| 预设任务番茄钟 | 输入预设事项和相关专注时长 | 添加相关的事项和时长到数据库中存储 | 预设事项形成 |
| 个性化商店 | 点击商店 | 加载数据库存储的商品信息 | 各个商品可在界面上展示出来 |
| 查看总览计时 | 点击统计总览 | 读取数据库中用户的专注信息 | 显示用户的蒸笼（也就是专注信息） |
| 查看时间历程（也叫历史时间轴） | 点击时间历程 | 加载日历，并从数据库中提取专注信息记录 | 界面上半部分是日历，日历下面是今日的专注记录 |
| 警告用户 | 用户在专注时期退出小程序界面 | 系统检测到界面切换出去的操作 | 弹出提示框，警告用户不专注了，请尽快回来 |
| 退出小程序 | 点击退出 | 系统检测到退出操作并允许执行 | 退出到微信界面 |

## 性能

说明对该程序的全部性能要求，包括对精度、灵活性和时间特性的要求。

### 精度

在进行向数据库提取所要信息时，只要输入相关的关键字，就能查找出相对应的信息；向数据库输入信息时，要确保输入的信息准确的并且数据类型要符合定义的数据类型。

### 灵活性

a．操作方式上的变化：暂无

b．运行环境的变化：暂无

c．同其他软件的接口的变化：暂无

d．精度和有效时限的变化：精度在必须调整的条件下，有效时限则依据现实的测试情况允许稍大范围的变化。

e．计划的变化或改进： 工作时间安排会存在必然的浮动，这部分要协同分工和组员一同来进行商定，前期的计划可以稍微有些变动，后期的安排尽量按照计划执行。

### 时间性

a启动该系统后，系统就能运行。

b.更新处理时间：由系统运行状态来决定

c.数据的转换和传送时间：能够在15秒钟内完成

## 输入项

给出对每一个输入项的特性，包括名称、标识、数据的类型和格式、数据值的有效范围、输入的方式。数量和频度、输入媒体、输入数据的来源和安全保密条件等等。

插入表格

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 格式 | 有效范围 | 输入方式 |
| 所有用户的专注时间相关信息 |  |  |  |  | 数据库获取 |

## 输出项

给出对每一个输出项的特性，包括名称、标识、数据的类型和格式，数据值的有效范围，输出的形式、数量和频度，输出媒体、对输出图形及符号的说明、安全保密条件等等。

插入表格

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 格式 | 有效范围 | 输出方式 |
| 用户专注排行榜 |  |  |  |  | 文字与图片 |

## 算法

var that = this;

var rankArray = new Array()

wx.cloud.callFunction({

    name: "mysqlOpreate",

    data: {

        sql: "select user\_id,user\_name,user\_avatarUrl from user"

    },

    success: function (res) {

        console.log(res)

        //将res中的信息写至rankList中

        that.userListInfo = res.result

        console.log(that.userListInfo)

        wx.cloud.callFunction({

            name: "mysqlOpreate",

            data: {

                sql: "select user\_id,steamer\_time\_count,steamer\_livetree\_count,steamer\_dietree\_count from steamer "

            },

            success: function (res) {

                for (app.globalData.i = 0; app.globalData.i < that.userListInfo.length; app.globalData.i++) {

                    var rankUser = { //用户对象

                        "username": "",

                        "userid": 0,

                        "useravatarUrl": "",

                        "sumtime": 0,

                        "completed": 0,

                        "failed": 0

                    }

                    rankUser.username = that.userListInfo[app.globalData.i].user\_name //将获取的信息填入对象内SS

                    rankUser.userid = that.userListInfo[app.globalData.i].user\_id

                    rankUser.useravatarUrl = that.userListInfo[app.globalData.i].user\_avatarUrl

                    console.log(rankUser)

                    for (var j = 0; j < res.result.length; j++) {

                        if (res.result[j].user\_id == rankUser.userid) {

                            that.setData({

                                tar: res.result[j]

                            })

                            break;

                        }

                    }

                    rankUser.sumtime = that.data.tar.steamer\_time\_count

                    rankUser.completed = that.data.tar.steamer\_livetree\_count

                    rankUser.failed = that.data.tar.steamer\_dietree\_count

                    rankArray[app.globalData.i] = rankUser

                    rankArray.sort(function (a, b) {

                        return b.sumtime - a.sumtime

                    })

                    that.setData({

                        rankingList: rankArray

                    })

                }

            },

            fail: console.error

        })

    },

    fail: console.error

})

## 流程逻辑

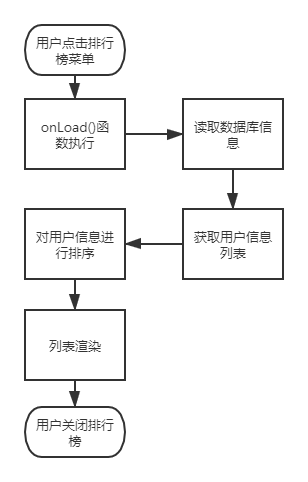


图 3 排行榜模块流程图

## 接口

用图的形式说明本程序所隶属的上一层模块及隶属于本程序的下一层模块、子程序，说明参数赋值和调用方式，说明与本程序相直接关联的数据结构（数据库、数据文卷）。

## 存储分配

根据需要，说明本程序的存储分配。

## 注释设计

说明准备在本程序中安排的注释，如：

1. 加在模块首部的注释；
2. 加在各分枝点处的注释；
3. 对各变量的功能、范围、缺省条件等所加的注释；
4. 对使用的逻辑所加的注释等等。

## 限制条件

说明本程序运行中所受到的限制条件。

## 测试计划

说明对本程序进行单体测试的计划，包括对测试的技术要求、输入数据、预期结果、进度安排、人员职责、设备条件驱动程序及桩模块等的规定。

## 尚未解决的问题

说明在本程序的设计中尚未解决而设计者认为在软件完成之前应解决的问题。

# 商店设计说明

从本章开始，逐个地给出各个层次中的每个程序的设计考虑。以下给出的提纲是针对一般情况的。对于一个具体的模块，尤其是层次比较低的模块或子程序，其很多条目的内容往往与它所隶属的上一层 模块的对应条目的内容相同，在这种情况下，只要简单地说明这一点即可。

## 程序描述

给出对该程序的简要描述，主要说明安排设计本程序的目的意义，并且，还要说明本程序的特点（如 是常驻内存还是非常驻？是否子程序？是可重人的还是不可重人的？有无覆盖要求？是顺序处理还是并发处理等）。

## 功能

说明该程序应具有的功能，可采用IPO图（即输入一处理一输出图）的形式。

以IPO图形式表现输入输出：

图表待修改

|  |  |  |  |
| --- | --- | --- | --- |
| 功能名称 | 输入input | 处理过程process | 输出output |
| 用户登录 | 用户点击小程序并且授权登陆。 | 系统服务器检测到登录情况并且允许登录操作。 | 进入小程序初始页面。 |
| 设置 | 点击 | 读取信号并切换页面 | 进入设置页面 |
| 设定专注时长 | 想要专注多长时间 | 经过汇编语言处理 | 生成一个番茄钟 |
| 开始（倒）番茄钟计时 | 发出专注番茄钟开始的信号 | 读取信号并判断 | 开始蒸包子（专注），番茄钟开始计时 |
| 蒸包子（专注）中 | 无 | 判断是否在专注，是否在不玩手机 | 蒸包子中，随着时间长度的变化，包子的形态也会随着改变 |
| 查看排行榜 | 点击排行榜的按钮 | 按照数据库中储存的用户专注时长来进行排序 | 排行榜上各个用户的排名 |
| 预设任务番茄钟 | 输入预设事项和相关专注时长 | 添加相关的事项和时长到数据库中存储 | 预设事项形成 |
| 个性化商店 | 点击商店 | 加载数据库存储的商品信息 | 各个商品可在界面上展示出来 |
| 查看总览计时 | 点击统计总览 | 读取数据库中用户的专注信息 | 显示用户的蒸笼（也就是专注信息） |
| 查看时间历程（也叫历史时间轴） | 点击时间历程 | 加载日历，并从数据库中提取专注信息记录 | 界面上半部分是日历，日历下面是今日的专注记录 |
| 警告用户 | 用户在专注时期退出小程序界面 | 系统检测到界面切换出去的操作 | 弹出提示框，警告用户不专注了，请尽快回来 |
| 退出小程序 | 点击退出 | 系统检测到退出操作并允许执行 | 退出到微信界面 |

## 性能

说明对该程序的全部性能要求，包括对精度、灵活性和时间特性的要求。

### 精度

在进行向数据库提取所要信息时，只要输入相关的关键字，就能查找出相对应的信息；向数据库输入信息时，要确保输入的信息准确的并且数据类型要符合定义的数据类型。

### 灵活性

a．操作方式上的变化：暂无

b．运行环境的变化：暂无

c．同其他软件的接口的变化：暂无

d．精度和有效时限的变化：精度在必须调整的条件下，有效时限则依据现实的测试情况允许稍大范围的变化。

e．计划的变化或改进： 工作时间安排会存在必然的浮动，这部分要协同分工和组员一同来进行商定，前期的计划可以稍微有些变动，后期的安排尽量按照计划执行。

### 时间性

a启动该系统后，系统就能运行。

b.更新处理时间：由系统运行状态来决定

c.数据的转换和传送时间：能够在15秒钟内完成

## 输入项

给出对每一个输入项的特性，包括名称、标识、数据的类型和格式、数据值的有效范围、输入的方式。数量和频度、输入媒体、输入数据的来源和安全保密条件等等。

插入表格

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 格式 | 有效范围 | 输入方式 |
| 当前登入的用户 |  |  |  |  | 数据库获取 |
| 皮肤信息表 |  |  |  |  | 数据库获取 |
| 声音信息表 |  |  |  |  | 数据库获取 |
| 购买动作 |  |  |  |  | 按钮 |

## 输出项

给出对每一个输出项的特性，包括名称、标识、数据的类型和格式，数据值的有效范围，输出的形式、数量和频度，输出媒体、对输出图形及符号的说明、安全保密条件等等。

插入表格

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 格式 | 有效范围 | 输出方式 |
| 皮肤信息 |  |  |  |  | 文字与图片 |
| 声音信息 |  |  |  |  | 文字与图片 |
| 购买信息反馈 |  |  |  |  | 跳出窗格文字提示 |

## 算法

详细说明本程序所选用的算法，具体的计算公式和计算步骤。

插入伪代码

**public** **class** Shoping{

变量：当前用户 currentLoginUser

变量：商品被购买状态 isbuy

变量：商品类 buns

容器：List of buns

/\*获取用户金币数\*/

User\_Money\_Get(){

从currentLoginUser中获取金币数信息

**return**

}

/\*商品信息获取\*/

Buns\_Info\_Get(){

**for** i=0 and buns to buns!=**null**

**do** buns=从数据库中获取第i个商品数据

在List of buns容器中添加buns元素

**return**

}

/\*购买商品\*/

Buns\_Buy(){

**int** cost = 从数据库tbl\_shop\_baozi中获取所购买buns的花费

**if** cost<User\_Money\_Get()

then {

将数据库tbl\_shop\_baozi中baozi\_isbuy修改为true

将数据库tbl\_user中的user\_money-=cost

}

**else** 返回购买失败信息

}

}

## 流程逻辑

用图表（例如流程图、判定表等）辅以必要的说明来表示本程序的逻辑流程。

插入图片

## 接口

用图的形式说明本程序所隶属的上一层模块及隶属于本程序的下一层模块、子程序，说明参数赋值和调用方式，说明与本程序相直接关联的数据结构（数据库、数据文卷）。

## 存储分配

根据需要，说明本程序的存储分配。

## 注释设计

说明准备在本程序中安排的注释，如：

1. 加在模块首部的注释；
2. 加在各分枝点处的注释；
3. 对各变量的功能、范围、缺省条件等所加的注释；
4. 对使用的逻辑所加的注释等等。

## 限制条件

说明本程序运行中所受到的限制条件。

## 测试计划

说明对本程序进行单体测试的计划，包括对测试的技术要求、输入数据、预期结果、进度安排、人员职责、设备条件驱动程序及桩模块等的规定。

## 尚未解决的问题

说明在本程序的设计中尚未解决而设计者认为在软件完成之前应解决的问题。

# 个人中心设计说明

从本章开始，逐个地给出各个层次中的每个程序的设计考虑。以下给出的提纲是针对一般情况的。对于一个具体的模块，尤其是层次比较低的模块或子程序，其很多条目的内容往往与它所隶属的上一层 模块的对应条目的内容相同，在这种情况下，只要简单地说明这一点即可。

## 程序描述

给出对该程序的简要描述，主要说明安排设计本程序的目的意义，并且，还要说明本程序的特点（如 是常驻内存还是非常驻？是否子程序？是可重人的还是不可重人的？有无覆盖要求？是顺序处理还是并发处理等）。

## 功能

说明该程序应具有的功能，可采用IPO图（即输入一处理一输出图）的形式。

以IPO图形式表现输入输出：

图表待修改

|  |  |  |  |
| --- | --- | --- | --- |
| 功能名称 | 输入input | 处理过程process | 输出output |
| 用户登录 | 用户点击小程序并且授权登陆。 | 系统服务器检测到登录情况并且允许登录操作。 | 进入小程序初始页面。 |
| 设置 | 点击 | 读取信号并切换页面 | 进入设置页面 |
| 设定专注时长 | 想要专注多长时间 | 经过汇编语言处理 | 生成一个番茄钟 |
| 开始（倒）番茄钟计时 | 发出专注番茄钟开始的信号 | 读取信号并判断 | 开始蒸包子（专注），番茄钟开始计时 |
| 蒸包子（专注）中 | 无 | 判断是否在专注，是否在不玩手机 | 蒸包子中，随着时间长度的变化，包子的形态也会随着改变 |
| 查看排行榜 | 点击排行榜的按钮 | 按照数据库中储存的用户专注时长来进行排序 | 排行榜上各个用户的排名 |
| 预设任务番茄钟 | 输入预设事项和相关专注时长 | 添加相关的事项和时长到数据库中存储 | 预设事项形成 |
| 个性化商店 | 点击商店 | 加载数据库存储的商品信息 | 各个商品可在界面上展示出来 |
| 查看总览计时 | 点击统计总览 | 读取数据库中用户的专注信息 | 显示用户的蒸笼（也就是专注信息） |
| 查看时间历程（也叫历史时间轴） | 点击时间历程 | 加载日历，并从数据库中提取专注信息记录 | 界面上半部分是日历，日历下面是今日的专注记录 |
| 警告用户 | 用户在专注时期退出小程序界面 | 系统检测到界面切换出去的操作 | 弹出提示框，警告用户不专注了，请尽快回来 |
| 退出小程序 | 点击退出 | 系统检测到退出操作并允许执行 | 退出到微信界面 |

## 性能

说明对该程序的全部性能要求，包括对精度、灵活性和时间特性的要求。

### 精度

在进行向数据库提取所要信息时，只要输入相关的关键字，就能查找出相对应的信息；向数据库输入信息时，要确保输入的信息准确的并且数据类型要符合定义的数据类型。

### 灵活性

a．操作方式上的变化：暂无

b．运行环境的变化：暂无

c．同其他软件的接口的变化：暂无

d．精度和有效时限的变化：精度在必须调整的条件下，有效时限则依据现实的测试情况允许稍大范围的变化。

e．计划的变化或改进： 工作时间安排会存在必然的浮动，这部分要协同分工和组员一同来进行商定，前期的计划可以稍微有些变动，后期的安排尽量按照计划执行。

### 时间性

a启动该系统后，系统就能运行。

b.更新处理时间：由系统运行状态来决定

c.数据的转换和传送时间：能够在15秒钟内完成

## 输入项

给出对每一个输入项的特性，包括名称、标识、数据的类型和格式、数据值的有效范围、输入的方式。数量和频度、输入媒体、输入数据的来源和安全保密条件等等。

插入表格

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标记 | 类型 | 格式 | 有效范围 | 输入方式 |
| 查看个人中心 | 9.1 |  | 按钮 | 0和1 | 点击按钮 |

## 输出项

给出对每一个输出项的特性，包括名称、标识、数据的类型和格式，数据值的有效范围，输出的形式、数量和频度，输出媒体、对输出图形及符号的说明、安全保密条件等等。

插入表格

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 格式 | 有效范围 | 输出方式 |
| 用户头像 | 9.1 | / | 图片 | / | 图片 |
| 用户名称 | 9.1 | String | 字母+数字 | a/A-z/Z、1-9 | 文字 |
| 用户id | 9.1 | String | 字母+数字 | a/A-z/Z、1-9 | 文字 |
| 总时长 | 9.1 | time | 时间 | 0-0000000 | 时间 |
| 失败/成功 占比 | 9.1 | double | 数字（百分比） | 0-100% | 数字（百分比） |
| 总成功数 | 9.1 | int | 数字 | 0-0000000 | 数字 |
| 成就完成度 | 9.1 | int | 数字 | 0-0000000 | 数字 |

## 算法

详细说明本程序所选用的算法，具体的计算公式和计算步骤。

插入伪代码

//个人中心设计伪代码

Public **class** personal{

变量：当前用户 currentLoginUser

//获取个人信息

CurrentLoginUser\_Info\_Get(){

从currentLoginUser中获取各个数据信息

**return**

}

}

## 流程逻辑

用图表（例如流程图、判定表等）辅以必要的说明来表示本程序的逻辑流程。

插入图片

## 接口

用图的形式说明本程序所隶属的上一层模块及隶属于本程序的下一层模块、子程序，说明参数赋值和调用方式，说明与本程序相直接关联的数据结构（数据库、数据文卷）。

## 存储分配

根据需要，说明本程序的存储分配。

## 注释设计

说明准备在本程序中安排的注释，如：

1. 加在模块首部的注释；
2. 加在各分枝点处的注释；
3. 对各变量的功能、范围、缺省条件等所加的注释；
4. 对使用的逻辑所加的注释等等。

## 限制条件

说明本程序运行中所受到的限制条件。

## 测试计划

说明对本程序进行单体测试的计划，包括对测试的技术要求、输入数据、预期结果、进度安排、人员职责、设备条件驱动程序及桩模块等的规定。

## 尚未解决的问题

说明在本程序的设计中尚未解决而设计者认为在软件完成之前应解决的问题。

# 管理员界面设计说明

从本章开始，逐个地给出各个层次中的每个程序的设计考虑。以下给出的提纲是针对一般情况的。对于一个具体的模块，尤其是层次比较低的模块或子程序，其很多条目的内容往往与它所隶属的上一层 模块的对应条目的内容相同，在这种情况下，只要简单地说明这一点即可。

## 程序描述

给出对该程序的简要描述，主要说明安排设计本程序的目的意义，并且，还要说明本程序的特点（如 是常驻内存还是非常驻？是否子程序？是可重人的还是不可重人的？有无覆盖要求？是顺序处理还是并发处理等）。

## 功能

说明该程序应具有的功能，可采用IPO图（即输入一处理一输出图）的形式。

以IPO图形式表现输入输出：

图表待修改

|  |  |  |  |
| --- | --- | --- | --- |
| 功能名称 | 输入input | 处理过程process | 输出output |
| 用户登录 | 用户点击小程序并且授权登陆。 | 系统服务器检测到登录情况并且允许登录操作。 | 进入小程序初始页面。 |
| 设置 | 点击 | 读取信号并切换页面 | 进入设置页面 |
| 设定专注时长 | 想要专注多长时间 | 经过汇编语言处理 | 生成一个番茄钟 |
| 开始（倒）番茄钟计时 | 发出专注番茄钟开始的信号 | 读取信号并判断 | 开始蒸包子（专注），番茄钟开始计时 |
| 蒸包子（专注）中 | 无 | 判断是否在专注，是否在不玩手机 | 蒸包子中，随着时间长度的变化，包子的形态也会随着改变 |
| 查看排行榜 | 点击排行榜的按钮 | 按照数据库中储存的用户专注时长来进行排序 | 排行榜上各个用户的排名 |
| 预设任务番茄钟 | 输入预设事项和相关专注时长 | 添加相关的事项和时长到数据库中存储 | 预设事项形成 |
| 个性化商店 | 点击商店 | 加载数据库存储的商品信息 | 各个商品可在界面上展示出来 |
| 查看总览计时 | 点击统计总览 | 读取数据库中用户的专注信息 | 显示用户的蒸笼（也就是专注信息） |
| 查看时间历程（也叫历史时间轴） | 点击时间历程 | 加载日历，并从数据库中提取专注信息记录 | 界面上半部分是日历，日历下面是今日的专注记录 |
| 警告用户 | 用户在专注时期退出小程序界面 | 系统检测到界面切换出去的操作 | 弹出提示框，警告用户不专注了，请尽快回来 |
| 退出小程序 | 点击退出 | 系统检测到退出操作并允许执行 | 退出到微信界面 |

## 性能

说明对该程序的全部性能要求，包括对精度、灵活性和时间特性的要求。

### 精度

在进行向数据库提取所要信息时，只要输入相关的关键字，就能查找出相对应的信息；向数据库输入信息时，要确保输入的信息准确的并且数据类型要符合定义的数据类型。

### 灵活性

a．操作方式上的变化：暂无

b．运行环境的变化：暂无

c．同其他软件的接口的变化：暂无

d．精度和有效时限的变化：精度在必须调整的条件下，有效时限则依据现实的测试情况允许稍大范围的变化。

e．计划的变化或改进： 工作时间安排会存在必然的浮动，这部分要协同分工和组员一同来进行商定，前期的计划可以稍微有些变动，后期的安排尽量按照计划执行。

### 时间性

a启动该系统后，系统就能运行。

b.更新处理时间：由系统运行状态来决定

c.数据的转换和传送时间：能够在15秒钟内完成

## 输入项

给出对每一个输入项的特性，包括名称、标识、数据的类型和格式、数据值的有效范围、输入的方式。数量和频度、输入媒体、输入数据的来源和安全保密条件等等。

插入表格

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 格式 | 有效范围 | 输入方式 |
| 管理员账号 |  |  |  |  | 数据库获取 |
| 管理员密码 |  |  |  |  | 数据库获取 |
| 输入的管理员账号 |  |  |  |  | 输入框 |
| 输入的管理员密码 |  |  |  |  | 输入框 |

## 输出项

给出对每一个输出项的特性，包括名称、标识、数据的类型和格式，数据值的有效范围，输出的形式、数量和频度，输出媒体、对输出图形及符号的说明、安全保密条件等等。

插入表格

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 类型 | 格式 | 有效范围 | 输出方式 |
| 功能模块列表 |  |  |  |  | 文字+图片 |
| 功能信息表 |  |  |  |  | 文字表格 |
| 可进行的操作 |  |  |  |  | 图标按钮 |

## 算法

详细说明本程序所选用的算法，具体的计算公式和计算步骤。

插入伪代码

**public** **class** Administrator{

变量：管理员对象 administrator

变量：管理员账号 administrator\_login\_id

变量：管理员密码 administrator\_login\_pwd

变量：功能类 func

容器：List of func

/\*管理员验证登录\*/

Administrator\_Login(administrator\_login\_id,administrator\_login\_pwd){

**if** 输入的id与pwd为空

then 返回提示信息

将数据库中tbl\_Administrator的administrator\_id匹配项的administrator\_pwd值取出

**if** 数据库得对应密码==administrator\_login\_pwd

then 成功登录

**else** 登录失败

**return**

}

/\*功能信息列表获取\*/

FuncList\_Info\_Get(){

**for** i=0 and func to func!=**null**

**do** func=从数据库中获取第i个选择的功能数据

在List of func容器中添加func元素

**return**

}

}

## 流程逻辑

用图表（例如流程图、判定表等）辅以必要的说明来表示本程序的逻辑流程。

插入图片

## 接口

用图的形式说明本程序所隶属的上一层模块及隶属于本程序的下一层模块、子程序，说明参数赋值和调用方式，说明与本程序相直接关联的数据结构（数据库、数据文卷）。

## 存储分配

根据需要，说明本程序的存储分配。

## 注释设计

说明准备在本程序中安排的注释，如：

1. 加在模块首部的注释；
2. 加在各分枝点处的注释；
3. 对各变量的功能、范围、缺省条件等所加的注释；
4. 对使用的逻辑所加的注释等等。

## 限制条件

说明本程序运行中所受到的限制条件。

## 测试计划

说明对本程序进行单体测试的计划，包括对测试的技术要求、输入数据、预期结果、进度安排、人员职责、设备条件驱动程序及桩模块等的规定。

## 尚未解决的问题

说明在本程序的设计中尚未解决而设计者认为在软件完成之前应解决的问题。