**设计报告**

1. **系统总体设计**
2. **系统总体功能描述**

本 “团体游景点门票预订系统” 模拟网上旅游门票预订平台 (如携程等) 的操作流程。用户可以以旅游团或管理员两种身份登录系统。进入系统后，可以登录或注册账户 (其中，为保障系统安全，管理员账户注册必须在已有管理员账户下进行，系统第一次运行时会创建默认管理员账户)。旅游团登录后可以根据景点名称、景点所在地区、剩余门票量是否充足、是否有折扣等查询系统中的景点信息，还可以对查询到的景点信息根据预订量及价格等进行排序。查询完毕后，可以直接输入景点ID查看景点详情，并可直接预订门票，对于门票量不足以及不适合当前年龄段的景点会有提示，确认预订前系统会根据当前季度以及旅客身份等计算总价，且预订后可以查询预订过的订单，并在出行前24小时内可以取消订单。旅游团可以对自己的信息（密码，旅游团人数、年龄段和联系电话）进行修改。管理员登录有可以查看系统中的所有景点的统计信息（不同季度的售票量、总售票量、总利润等），并根据这些统计信息对景点信息进行排序，还可以查看景点游客的年龄段分布，查看系统中所有订单、添加景点以及修改个人信息。

模块图

1. **接口定义规范**

源码文件: 本系统共有5 个源码文件(.cpp文件)，分别对应旅游团、管理员两个功能板块，以及数据库操作、输入验证相关函数，另有一个源码文件存放main函数。每个文件的文件名均为代表其功能的英文单词。除main函数所在文件外，每个源码文件对应一个头文件(.h文件)，用于存放该源码文件导出给其他源码文件调用的函数的声明，另有三个头文件存放数据结构体(Model.h)、控制台颜色码(Color.h)、以及每个模块都要使用的基本头文件(stdafx.h, 其中引用了stdio.h等基本头文件), sqlite3.h及pcre.h均为第三方库头文件，其他头文件均为IDE自动生成。

函数名：所有函数名均为表示其功能的英文单词的组合，除第一个字母外，每一个单词的第一个字母均为大写，以便分隔单词。具体功能模块(第一页模块图中的模块)对应的函数名首字母均大写

结构体变量：结构体名、结构体成员变量名均为表示对象的英文单词组合，单词间通过首字母大写分隔，结构体名的首字母大写，结构体成员变量名首字母小写。

1. **系统的数据结构描述**
2. 基本类型及指针类型

整数型变量（int），字符型（char）, 单精度浮点数 (float)

字符型指针(char\*): 引用字符串

旅游团信息结构体指针: 用于在订单结构体中引用发起这笔订单的旅游团

景点信息指针：用于在订单结构体中引用其包含的景点信息数组

景点人数限制信息指针: 用于在景点结构体中引用其包含的人数限制信息数组

其他结构体指针：用于在模块间传递结构体数据，数据需要为空时(如数据库查询失败)可直接使用NULL

1. 枚举类型

程序只包含一种枚举类型，代表旅游团中旅客的身份，其定义位于Model.h

typedef enum {

NORMAL = 0,

STUDENT = 1,

SOLDIER = 2,

} TouristIdantity;

1. 结构体类型

所有结构体变量定义位于Model.h 中，这些结构体存储的信息均存储在数据库中

1. 旅游团信息结构体: 记录旅游团信息，存储在数据库中的TouristGroup表中

typedef struct {

char id[11];

char password[20];

int peopleCount;

int age;

char phone[12];

} TouristGroup;

1. 系统管理员信息结构体：记录系统管理员信息，存储于数据库中的 SystemAdmin表

typedef struct {

char id[11];

char password[20];

char email[30];

char phone[12];

} SystemAdmin;

1. 景点人数限制信息结构体：记录景点人数限制信息，每个景点可以有多条，存储在数据库中的FeatureSpotLimitation表中

typedef struct {

int id;

char featureSpotId[7];

int isHot;

char timeSpanStart[6];

char timeSpanEnd[6];

int peopleCount;

} FeatureSpotLimitation;

1. 景点信息结构体：记录系统中的景点信息，存储在数据库中的FeatureSpot表中，其中 limitations和limitionCount成员在数据库中没有对应的字段，这两个成员会在查询时自动从FeatureSpotLimitation表中查询并填入，添加景点时会自动将limitations包含的人数限制信息写入FeatureSpotLimitation表。其中包含7个统计字段，有新订单及有订单取消时它们的值会被更新，添加景点时总利润的值为维护费用的负数。

typedef struct {

char id[7];

char name[20];

char discription[200];

int coldSeasonPrice;

int hotSeasonPrice;

int timeRequired;

float soldierDiscount;

float studentDiscount;

int level;

FeatureSpotLimitation \*limitations;

int limitationCount;

char district[20];

float maintenanceFee;

int reservationCount; // 统计字段开始

int visitCount;

int totalTicketSold;

int coldSeasonTickets;

int hotSeasonTickets;

int totalTicketCount;

float totalProfit; // 统计字段结束

int isSuitableForEldersAndChildren;

} FeatureSpot;

1. 订单信息结构体：记录旅游团创建的订单信息，存储于Reservation表中。其中featureSpots、featureSpotCount 没有对应的数据库字段, 这些字段会在查询时自动从数据库中的ReservationFeatureSpot表（该表只有两个字段，无需为其创建结构体，其定义详见下一部分）中查询，添加订单时将订单与景点的关系存入ReservationFeatureSpot表中。touristGroup成员对应数据库中的touristGroupID字段，数据库中的字段只包含发起订单的旅游团ID，查询时程序会自动通过数据库中的旅游团ID查询旅游团信息并写入touristGroup成员

typedef struct {

char id[30];

char orderDate[18];

FeatureSpot \*featureSpots;

int featureSpotCount;

int isHotSeason;

TouristGroup \*touristGroup;

char time[18];

TouristIdantity idantity;

float totalPrice;

} Reservation;

1. 数据存储方式

程序中的所有数据均存储于sqlite数据库。Sqlite是一种轻量级的数据库系统，能够非常方便地对数据进行操作，同时比传统的数据库系统如MySQL、SQL Server更加简洁，更适合小型程序使用。本程序所涉及到的数据增删改查以及排序均为通过构造SQL语句实现，无需为其另外编写代码。本系统所用数据库文件名为Database.db, 第一次运行时会自动创建数据库文件，并写入默认管理员信息。

系统数据库中使用的表的创建语句(SQL)如下:

旅游团信息：

CREATE TABLE TouristGroup (

id CHAR(10) PRIMARY KEY NOT NULL,

password CHAR(20),

peopleCount INTEGER ,

ages INTEGER,

phone CHAR(11) UNIQUE);

管理员信息：

CREATE TABLE SystemAdmin (

id CHAR(10) PRIMARY KEY NOT NULL,

password CHAR(20),

phone CHAR(11) UNIQUE,

email CHAR(30) UNIQUE);

景点信息：

CREATE TABLE FeatureSpot (

id CHAR(6) PRIMARY KEY NOT NULL, name CHAR(20),

discription TEXT,

coldSeasonPrice INTEGER,

hotSeasonPrice INTEGER,

timeRequired INTEGER,

soldierDiscount REAL,

studentDiscount REAL,

level INTEGER,

district CHAR(20),

maintenanceFee REAL,

reservationCount INTEGER,

visitCount INTEGER,

totalTicketSold INTEGER,

coldSeasonTickets INTEGER,

hotSeasonTickets INTEGER,

totalTicketCount INTEGER,

totalProfit REAL,

isSuitableForEldersAndChildren INTEGER );

景点人数限制信息：

CREATE TABLE FeatureSpotLimitation (

id INTEGER PRIMARY KEY AUTOINCREMENT,

featureSpotId CHAR(6),

isHot INTEGER,

timeSpanStart CHAR(5),

timeSpanEnd CHAR(5),

peopleCount INTEGER);

订单信息：

CREATE TABLE Reservation (

id CHAR(30) PRIMARY KEY NOT NULL,

orderDate CHAR(18),

isHotSeason INTEGER,

touristGroupId CHAR(10),

time CHAR(18),

idantity INTEGER,

totalPrice REAL);

订单与景点间的关系：

CREATE TABLE ReservationFeatureSpot (

reservationId CHAR(30),

featureSpotId CHAR(10),

PRIMARY KEY(reservationId, featureSpotId));

1. **系统使用的第三方库**

本系统使用了两个第三方库，以便实现某些功能

1. Sqlite3

sqlite数据库的c语言实现，在本系统中用于数据库相关功能

1. Pcre

轻量级c语言正则表达式库，在本系统中用于输入验证

本系统使用的所有第三方库均为开源库，均为官网下载的编译好的库文件，其源代码没有经过任何修改，均通过动态链接的方式与主程序链接 (即在程序运行目录下要有pcre.dll与sqlite3.dll两个文件)。

1. **系统开发环境及运行环境描述**

开发环境: IDE为微软Visual Studio 2017, 编译器为Visual Studio自带的 cl 12.0 编译器, 软件环境为Windows 10。

运行环境: 软件环境为windows 10操作系统，硬件环境为 Intel Xeon E3-1505 CPU, 16GB运行内存，1TB SSD。

1. **模块详细设计**
2. **游客团及管理员登录模块**
3. 功能概述: 此模块实现管理员及游客团登录账号的过程, 游客团或管理员输入账号和密码登录系统，若账号不存在或密码错误则给出提示，并反复要求重新输入，直到输入正确为止
4. 处理流程: 系统从键盘读取输入的ID号和密码，然后根据ID号从数据库相应表中检索用户信息，如检索不到，则输出错误信息，否则将检索到的用户信息中的密码与用户输入的密码进行比较，如密码错误，输出错误信息，否则直接进入主菜单
5. 模块的输入和输出：输入只有字符串类型，包含用户名和密码，输出内容均为提示信息。

流程图：

输入ID及密码

从数据库中查询ID

存在？

验证密码

密码正确？

进入相应主菜单

否

否

是

是

1. 接口描述：
2. 旅游团登录模块: void TouristGroupLogin(); (位于TouristGroup.cpp中, 函数声明位于TouristGroup.h中)

无参数及返回值，包含了旅游团登录功能。

1. 管理员登录模块: void Login(); (位于 Admin.cpp中)

无参数及返回值，包含了旅游团登录功能。

1. 边界情况及处理办法

边界情况: 输入的字符串长度不在6-10个字符之间

解决方法: 在Validation.cpp 中定义了函数inputStringWithLengthLimit，用于确保用户输入的字符串长度正确，该函数内部使用gets\_s函数(gets函数的安全版本，可以限制读取大小，防止越界)读入一个长为100的字符串，然后验证其真实长度，如不符合要求，则要求用户重新输入。

1. **游客团/管理员注册模块**
2. 功能概述：此模块实现旅游团及管理员的注册功能，新用户输入相关信息(旅游团ID，密码、人数、年龄段、联系电话或管理员ID、密码、邮箱、电话) 即可注册账号，当用户输入ID、邮箱或联系电话时，如果输入的信息已被其他旅游团或管理员使用，则提示用户重新输入，由于输入密码时使用”\*”代表字符，为防止用户输入错误，系统将在用户输入密码后再次要求用户输入密码并验证。其中管理员注册必须在已登录已有管理员账号的情况下进行。
3. 处理流程：先从键盘读取旅游团ID，然后再数据库中查询该ID，如果已经存在，则给出提示并要求重新输入，否则再读取密码，密码要求用户输入两次，如两次密码输入不一致则给出提示并要求重新输入。然后再读取旅游团人数及年龄段，读取联系电话时查询是否已被其他用户占用。输入完所有信息并验证正确后，将新的旅游团信息存入数据库，如果成功存入则显示绿色“注册成功”字样，继续按回车即可进入主菜单。如添加记录时发生错误，则显示红色错误信息。流程图：

输入ID

存在？

是

输入两次密码

相同？

否

输入其他信息

否

是

有效？

否

是

存入数据库

立即以新旅游团身份登录系统

1. 输入及输出：输入的ID和联系电话为字符串类型，年龄段、人数为整数类型。输出均为提示文字。
2. 接口描述：a. 旅游团注册：void TouristGroupRegister(); (位于TouristGroup.cpp中, 函数声明位于TouristGroup.h中) 无参数及返回值
3. 管理员注册： void AddAdmin(); 位于Admin.cpp, 无参数及返回值
4. 边界情况及处理办法

边界情况：ID不符合要求

解决方法：ID为字母和数字的组合，长度在6-10个字符之间，可以使用 ^[A-Za-z0-9]{6,10}$ 这个正则表达式进行验证。

边界情况：ID或联系电话已被占用，如直接写入数据库将出错

解决办法：在用户输入ID或联系电话后立即在数据库中查询，如已存在，则给出警告并要求重新输入

边界情况：用户输入的手机号码格式有误

解决办法：使用正则表达式 ^[0-9]{11}$ 进行验证

边界情况：用户输入的整数不正确(如包含字母等其他字符)

解决方法：读取整数时，先读入字符串，再使用正则表达式 ^[0-9]\*$ 验证，验证通过后将字符串通过atoi函数转换为整数。

1. **旅游团景点查询模块**
2. 功能概述：用户选择查询方式，并输入必要查询信息，然后选择排序方式，然后输出景点查询结果，然后询问用户是否查询景点详情，如用户选是，则要求用户输入景点ID，然后查询景点详情。系统有热门景点推荐，即只显示预订量大于10的景点。
3. 处理流程：

景点查询

按名称

按是否有票

按区域

按有无折扣

输入关键词

输入区域

构建查询条件语句(SQL中WHERE部分)

选择排序方式

根据价格排序

根据预订量排序

构建查询排序语句(SQL中ORDER BY部分)

查询数据库

输出查询结果

景点详情查询

热门推荐

否

景点详情查询

输入景点ID

有效?

查询并显示景点详情

门票充足?

适合当前年龄段

门票预订

是

是

是

1. 输入及输出：输入查询方式及查询所需信息(按名称查询时，输入关键词；按地区查询时，输入地区名)，还需输入排序方式。输出包含景点列表及后面查询的景点详情。如用户选择预订门票，则还输出一个景点ID进入景点门票预订模块。
2. 接口描述：void QueryAndOrder();

位于TouristGroup.cpp文件，无返回值及参数。

1. 边界情况及处理方法

边界情况: 输入的查询方式或排序方式不在给定选项中

解决方法: 每次输入调用\_getch()函数(封装于Validation.h中的readKey()函数中)读取用户输入的字符，如不在选项范围内，则要求重新输入。

1. **旅游团门票预订模块**
2. 功能概述：用户输入要预订门票的景点ID，可以输入多个景点ID，输入后系统会检查当前景点门票是否充足、景点是否适合当前年龄段。预订前系统会要求用户输入当前旅游团身份信息(普通群众、军人、学生)和出行时间。然后系统会根据输入的景点、旅游团身份信息、当前季度、旅游团人数计算总价格，并显示出来，再询问用户是否预订，如选是，则添加订单数据到数据库，并更新数据库中对应景点信息中的统计字段。
3. 处理流程：

门票预订

输入景点ID

门票充足

适合当前年龄段

输入旅客身份

输入出行时间

计算并显示价格

确认预订？

返回

生成订单数据并写入数据库

更新景点统计信息

否

否

否

是

是

是

1. 输入及输出: 该模块需输入景点ID、旅客身份、出行时间，其中景点ID及出行时间为字符串型变量，旅客身份为枚举型变量，输入时以数字表示。输出包含一个单精度浮点型的总价格，以及各种提示信息。
2. 接口描述: void MakeReservation(char \*id);

位于TouristGroup.cpp中，包含一个字符型的id参数，用于景点查询模块给本模块传递景点ID，如在景点查询模块中用户选择了预订该景点门票，则该参数将传入景点ID，将跳过输入景点ID的步骤。如在景点查询模块中用户选择了预订其他景点，则传入参数为NULL，此时将要求用户输入景点ID，可输入多个ID。

1. 边界情况及解决方法

边界情况：用户输入的景点ID不存在

解决方法：输入后在数据库中查询该景点ID，如不存在则要求重新输入

边界情况：用户输入的旅客身份不符合要求

解决方法：每次读取用户输入的字符，如不在选项范围内，则要求重新输入。

边界情况：出行时间不符合日期格式（yyyy:mm:dd:hh:mm）

解决方法：使用以下正则表达式匹配用户输入字符串，如匹配失败则要求重新输入 ^(\d\d\d\d):(0[1-9]|1[0-2]):([0-3]\d):(20|21|22|23|[0-1]\d):[0-5]\d$

1. **旅游团订单管理模块**
2. 功能概述：查询旅游团订单，并可查询单笔订单的详情，可在出行前24小时前取消订单。
3. 处理流程：

从数据库中查询当前旅游团的所有订单

输出订单信息

查询详情?

输入订单ID

返回

存在?

属于当前用户?

查询并输出订单详细信息

可取消？

取消?

删除数据库中的订单数据

更景点的统计信息

否

否

是

是

是

是、

1. 输入输出：输入订单ID(字符型)，及是否取消订单(整型)，输出订单列表及订单详情。
2. 接口定义：void ReservationManagetment();位于TouristGroup.cpp，无返回值及参数
3. 边界情况及处理办法

边界情况：输入的订单ID不存在及不属于当前旅游团

解决方法：输入订单ID后先查询数据库，如订单信息不存在或不属于当前旅游团，则要求重新输入

1. **旅游团/管理员信息修改模块**
2. 功能概述：该模块实现旅游团/管理员信息查看与修改，可修改的信息包括游客团密码、人数、年龄段、联系电话或管理员密码、邮箱、手机号。进入该模块后按下要修改的信息对应的数字键即可修改该信息，其中密码要求输入两次。
3. 处理流程

输出当前旅游团/管理员信息

选择要修改的信息

输入新的信息

有效？

更新数据库中当前旅游团/管理员的信息

是

否

1. 输入输出：输入包含一个字符（用户选择要修改的信息）以及一个字符串，代表用户修改的信息，输出内容为提示文字。
2. 接口定义：a. 旅游团信息修改：void ModifyInfo(); 位于TouristGroup.cpp文件，无返回值及参数。

b. 管理员信息修改：void ModifyPersonalInfo(); 位于Admin.cpp ，无返回值及参数

1. 边界情况及处理办法

边界情况：用户在应该输入数字的地方输入了其他字符

处理办法：通过正则表达式验证用户输入，验证整数的正则表达式为 ^[0-9]\*$

边界情况：用户输入的邮箱或手机号码不符合格式要求，或已被其他用户使用

处理办法：通过正则表达式 ^[0-9]{11}$ 验证手机号， ^[a-zA-Z0-9\_.-]+@[a-zA-Z0-9-]+(\\.[a-zA-Z0-9-]+)\*\\.[a-zA-Z0-9]{2,6}$ 验证邮箱，并在输入后查询数据库，确保手机号或邮箱没有被使用，否则要求重新输入

1. **管理员景点查询及统计模块**
2. 功能概述：该模块能够列出所有景点及其统计信息（预订量、访问量、利润、不同季度的售票量），并根据这些统计信息对景点信息进行排序。在列出景点后，管理员可以选择查看景点详情，这将列出所查询景点的所有信息，包含各项统计信息及年龄分布。
3. 处理流程:

选择排序方式

按预订量排序

按访问量排序

按利润排序

按不同季度售票量排序

构建SQL查询语句

查询并显示景点信息

查询详情？

返回

输入景点ID

存在？

查询并输出景点详情

统计景点游客年龄段分布

是

是

否

否

1. 输入输出：如查询景点详细信息，输入包含要查询的景点ID。输出包含排序过的景点列表，如查询了详细信息，则还有景点详细信息、景点旅客年龄段分布信息。
2. 接口定义：void ListFeatureSpot(); 位于Admin.cpp中，无返回值及参数。
3. 边界情况及处理办法

边界情况：查询景点详情时输入的景点ID无效。

解决方法：输入后查询数据库，如不存在则输出错误信息，并要求重新输入，否则直接输出查询到的信息

1. **管理员订单查询模块**
2. 功能概述：该模块能够列出所有订单信息。在列出订单后，管理员可以选择查看订单详情。
3. 处理流程：

输出所有旅游团订单

查询详情?

输入订单ID

存在？

查询并显示订单信息

是

否

1. 输入输出：如查询详细信息，输入包含要查询的订单ID。输出包含所有旅游团的订单列表，如查询了详细信息，则还有订单详细信息。
2. 接口定义：void ListReservations();位于Admin.cpp中，无返回值及参数。
3. 边界情况及处理办法

边界情况：查询订单详情时输入的订单ID无效。

解决方法：输入后查询数据库，如不存在则输出错误信息，并要求重新输入，否则直接输出查询到的信息

1. **管理员景点添加模块**
2. 功能概述：该模块允许管理员添加新的景点，添加景点时要输入景点的详细信息以及景点的人数限制信息。
3. 处理流程：

将新的景点信息存入数据库

输入景点信息

有效？

输入景点人数限制信息

显示输入的景点所有信息

发布？

否

是

是

1. 输入输出：输入包含景点的各项信息，以及最多十条的景点人数限制信息。输出为提示文字。
2. 接口定义：void AddFeatureSpot(); 位于Admin.cpp, 无任何返回值和参数
3. 边界情况及处理办法

边界情况：输入的数字格式不正确（如包含其他字符）

处理方法：整数用正则表达式 ^[0-9]\*$ 验证，浮点数用 ^[0-9]\*\.?[0-9]+$，如验证失败，要求重输，否则通过atoi及atof将输入的字符串转换为数字。

边界情况：输入的景点ID不符合要求

处理方法：使用正则表达式 ^[A-Za-z0-9]{6,10}$ 进行验证，验证失败要求重新输入。

边界情况：输入的景点ID或景点名称与其他景点冲突

处理办法：输入景点ID或名称后查询数据库，确保ID和名称没有被其他景点使用，否则要求重新输入。

边界情况：输入景点人数限制信息时，输入的时间格式不正确。

处理办法：使用正则表达式 ^(20|21|22|23|[0-1]\\d):[0-5]\\d$ 验证输入，如验证失败要求重输