

# 十六组第五周周报

**题目： 十六组第五周周报**

**小组班级： 2022211307**

**专业： 计算机学院（国家示范级软件学院）**

成员1学号： 2022212442

成员1姓名： 范兼玮

成员2学号： 2022211209

成员2姓名： 许恒栋

成员3学号： 2022211184

成员3姓名： 徐家辉

**2024 年 3 月 27 日**

目录

[项目进展概述： 3](#_Toc162448492)

[1. 数据准备： 3](#_Toc162448493)

[2. 系统设计与实现 4](#_Toc162448494)

[2.1系统设计： 4](#_Toc162448495)

[2.2系统运行示例： 4](#_Toc162448496)

[3. 核心算法 6](#_Toc162448497)

[3.1排序： 6](#_Toc162448498)

[3.1.1排序说明： 6](#_Toc162448499)

[3.1.2：与本系统相关的创新部分: 6](#_Toc162448500)

[3.2搜索： 7](#_Toc162448501)

[3.2.1：搜索说明： 7](#_Toc162448502)

[3.1.2：与本系统相关的创新部分 7](#_Toc162448503)

[4.下周计划： 8](#_Toc162448504)

[4.1游学路线规划功能开发 8](#_Toc162448505)

[4.2系统内容的图形化 8](#_Toc162448506)

## 项目进展概述：

本周我们着手开发游学推荐系统，并取得了一些进展。这周任务是完成基本功能需求（续1）部分的内容，其主要目标是根据学生的兴趣和喜好，推荐适合的景点和学校作为游学目的地，并支持查询和排序功能。在本周的工作中，我们完成了第一部分要求的功能，具体进展如下：

# 数据准备：

我们收集了一部分景点和学校的信息作为系统的数据源。这些数据将作为系统推荐和查询功能的基础。其中数据格式为:景点名称(1-5分) 游学热度(1-5分) 类型(1代表休闲娱乐，2代表名胜古迹，3代表踏青公园，4代表博物馆，5代表高校)。储存景点的数据结构如下：

typedef struct {

    char name[100];

    float popularity; // 游学热度

    float review; // 评价

    int type; // 景点类型：1代表休闲娱乐，2代表名胜古迹，3代表踏青公园，4代表博物馆，5代表高校

 } Place;

其中景点数据在data.txt中存储了223个景点数据，部分数据示例如下：

杭州西湖 4.8 4.9 2

西安兵马俑 4.9 4.8 2

上海迪士尼乐园 5.0 4.6 1

南京中山陵 4.7 4.8 2

丽江古城 4.4 4.9 2

武汉黄鹤楼 4.8 4.7 2

敦煌莫高窟 4.3 4.9 2

厦门鼓浪屿 4.7 4.8 2

故宫博物院 4.9 4.9 4

北京大学 4.9 4.8 5

清华大学 4.8 4.9 5

复旦大学 4.7 4.8 5

浙江大学 4.6 4.7 5

中国科学技术大学 4.5 4.8 5

湖南大学 4.4 4.7 5

中山大学 4.7 4.8 5

上海博物馆 4.3 4.9 4

西安博物院 4.5 4.8 4

# 系统设计与实现

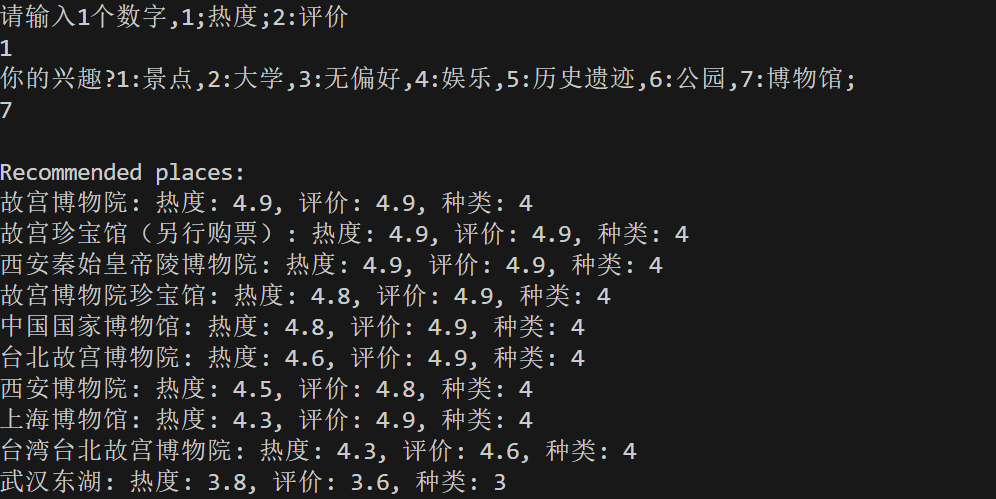
## 2.1系统设计：

实现了一个基于排序算法的推荐算法，能够根据游学景点和学校的热度、评价来进行排序，满足了核心算法的要求，尤其是在不进行完全排序的情况下快速找到前10个推荐项目的能力。

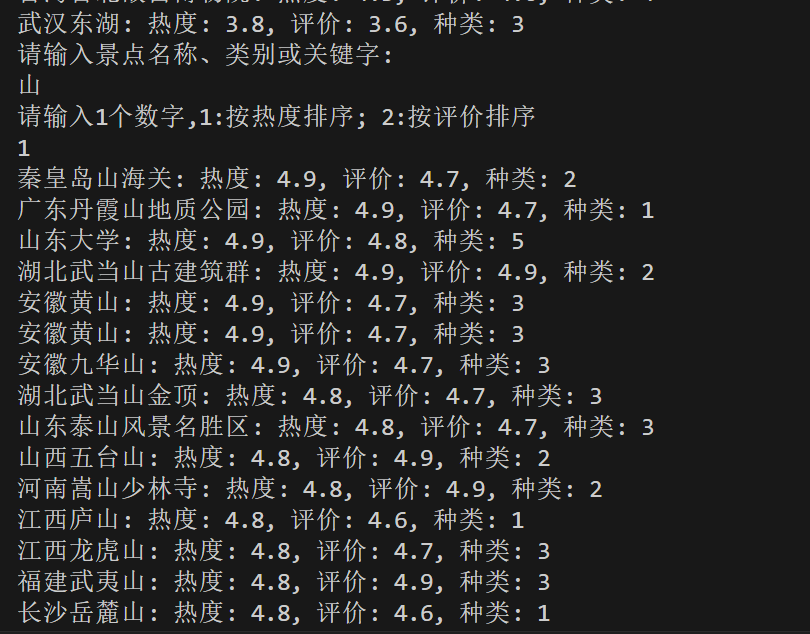
开发了一个查询功能，允许学生通过输入景点和学校的名称、类别或关键字来查找感兴趣的项目。查询结果支持按热度和评价进行排序，结合了查找算法和排序算法。

## 2.2系统运行示例：

由于暂时没有实现图形化，只在main函数里面调用主要函数来展示功能。

排序示例：用户可以选择按照热度，评价或者自己的兴趣来实现排序。

搜索示例：用户可以任意输入景点名称与关键字。



且当用户输入”休闲娱乐”，名胜古迹”，”踏青公园”，”博物馆”，”高校”时，该系统会自动筛选出对应景点类型的地方。

图片包含 文本

描述已自动生成

# 核心算法

使用快速选择（Quick Select）算法和插入排序（Insertion Sort）算法来实现推荐列表的快速生成和排序，优化了性能，特别是针对用户关心的前10个推荐项。

实现了一个高效的分区（Partition）逻辑，以支持基于个人兴趣的过滤，进一步个性化推荐结果。

## 3.1排序：

### 3.1.1排序说明：

本系统是基于排序算法的推荐机制，该推荐系统采用了快速选择（Quick Select）算法和插入排序（Insertion Sort）算法，高效地处理游学景点和学校的数据，以热度和评价为基准进行排序。这种方法的优势在于：

快速选择算法的运用，使得系统能够在没有对所有项目进行完全排序的情况下，迅速定位到热度或评价排在前10的游学项目。这一点对用户来说非常关键，因为用户往往最关心排名靠前的选项。快速选择算法通过分区操作，将数据分为两部分，一部分是小于某个选定值的元素，另一部分是大于或等于这个值的元素，这样重复进行直到找到所需的前N个元素，大幅提高了查找效率。

插入排序算法则用于对这前10个元素进行精确排序，保证了推荐结果的准确性和实用性。插入排序通过比较和移动操作，逐步将元素移至其应在的位置，尤其适合于小规模数据集的排序，因此它被用来对快速选择算法找到的前10个推荐项目进行细致排序。

### 3.1.2：与本系统相关的创新部分:

int partition(Place arr[], int low, int high, int sortKey, int interestType)

void quickSelect(Place arr[], int low, int high, int k, int sortKey, int interestType)

其中相较于传统的快排算法，本系统中的快排函数增加了两个参数sortKey与interestType。他们在main函数中的输入如下：

    int sortKey;

    printf("请输入1个数字,1;热度;2:评价\n");

    scanf("%d", &sortKey);

    // 用户选择个人兴趣

    int interestType;

    printf("你的兴趣?1:景点,2:大学,3:无偏好,4:娱乐,5:历史遗迹,6:公园,7:博物馆;\n");

    scanf("%d", &interestType);

可以看到，本系统中的快排函数增添了sortKey可以指定用户热度与评价来进行排序的参数与interestType可以先筛选出指定符合用户个人兴趣类型的景点再进行排序。

## 3.2搜索：

### 3.2.1：搜索说明：

高效灵活的查询功能:系统的查询功能允许学生通过输入景点和学校的名称、类别或关键字进行搜索，这一功能的实现基于高效的查找算法，支持对查询结果按热度和评价进行排序。查询与排序的结合，不仅使得用户能够快速找到符合自己兴趣和偏好的游学项目，而且还能够得知这些项目的受欢迎程度和公众评价，帮助用户做出更加信息充分的决策。

关键字匹配：系统通过匹配用户输入的关键字（如景点名称、类别等）来过滤数据，快速返回相关的查询结果。这种灵活的查询方式大大增强了用户体验，使用户能够轻松地找到自己感兴趣的信息。

动态排序：在查询结果返回后，用户还可以选择不同的排序标准（例如按热度或评价排序），这一功能的实现依赖于动态的排序算法。系统会根据用户的选择，重新排序查询结果，确保用户总是能看到最符合自己当前需求的信息排列。

### 3.1.2：与本系统相关的创新部分

搜索函数：void searchAndSortPlaces(Place places[], int n)

根据用户输入的类型或关键字，函数遍历整个places数组，筛选出匹配的景点。匹配逻辑包括直接的字符串匹配和类型匹配。所有匹配的景点索引被存储在filteredIndexes数组中。

处理无匹配情况：如果没有找到任何匹配的景点，函数会输出提示信息并提前终止执行。

排序方式选择：函数提示用户选择排序方式，即按热度或评价排序，然后根据用户的选择进行排序操作。

其中，搜索函数也在内部做出了一个符合本系统要求的，与传统搜索函数不同的功能：当输入”休闲娱乐”，名胜古迹”，”踏青公园”，”博物馆”，”高校”时，该系统会自动筛选出对应景点类型的地方。

实现说明：

当识别到用户输入以上部分内容时，系统进入以下部分：

  recommendPlaces(places, n, sortKey, type);

     // 注意: 这个函数现在需要接收和处理interestType参数

        int topN = filteredCount ;

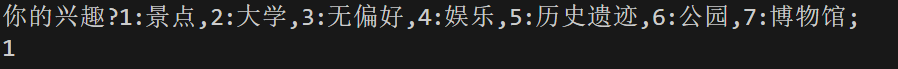
        for (int i = 0; i < topN; ++i) {

            printf("%s: 热度: %.1f, 评价: %.1f, 种类: %d\n",

                   places[i].name, places[i].popularity, places[i].review, places[i].type);

        }

}

这部分内容与第一部分排序时根据用户搜索时输入兴趣时使用的函数与代码完全相同，提高了代码的复用率。（该功能如下：）

通过这些设计和实现，游学推荐系统不仅提供了高效准确的推荐列表，而且还提供了强大灵活的查询功能，大大提升了用户的使用体验和满意度。未来的开发中，我们将继续优化这些算法和功能，进一步提升系统性能和用户体验。

# 4.下周计划：

接下来的一周，我们的团队将专注于进一步丰富和完善游学推荐系统，主要工作内容包括两大部分：一是开发和完善游学路线规划的相关功能；二是争取实现系统内容的图形化，提升用户体验。以下是详细的工作计划：

## 4.1游学路线规划功能开发

1.单个目标景点的最优游学线路规划：我们计划实现一个功能，使得用户在输入一个目标景点或场所信息后，系统能够基于用户当前位置规划到达该目标的最优路线。核心算法将基于最短路径算法，如迪杰斯特拉算法或A\*算法。

2.多个目标景点的最优游学线路规划：我们还将开发一个更为复杂的功能，允许用户输入多个希望访问的景点或场所。系统将为用户规划一条包含所有这些景点的最优游学线路，既满足从当前位置出发，又能在参观完所有景点后返回到出发点。这一功能的核心算法将探索解决途经多点的最短/次优路径算法，如旅行商问题(TSP)求解算法。

## 4.2系统内容的图形化

界面设计与实现：为了提升用户体验，我们将争取实现系统操作界面和展示内容的图形化。这包括游学路线的图形展示，使用户能够直观地看到从当前位置到目标景点的推荐路线，以及包含多个景点的整体游学路线图。

交互优化：通过图形化界面，我们计划提供一个更为直观和用户友好的操作体验。用户将能够通过点击和拖拽等简单操作，输入景点信息、查看路线规划结果，并获得关于景点的更多图形化信息，如景点图片、路线标识等。

功能集成测试：图形化内容的实现将伴随着功能的集成测试，确保新开发的路线规划功能与图形化界面无缝结合，同时保持系统的稳定性和可靠性。

下周，我们的目标不仅是完成上述功能的开发和测试，还包括确保这些功能的用户体验达到最佳。我们相信，通过实现这些计划，能够显著提升游学推荐系统的整体质量和用户满意度。

## 备注：

本周进展虽然有限，但我们已经完成了系统的一部分功能，并确定了下一步的工作计划。我们将继续努力，争取在下周取得更多的进展，完成系统的开发任务。

感谢团队成员的辛勤工作和配合。