知识竞赛小程序

程序设计基础实验：项目四

郑凯琳 205220025

实验内容概述

EasyX图形库结合算法，实现一个知识竞赛小程序

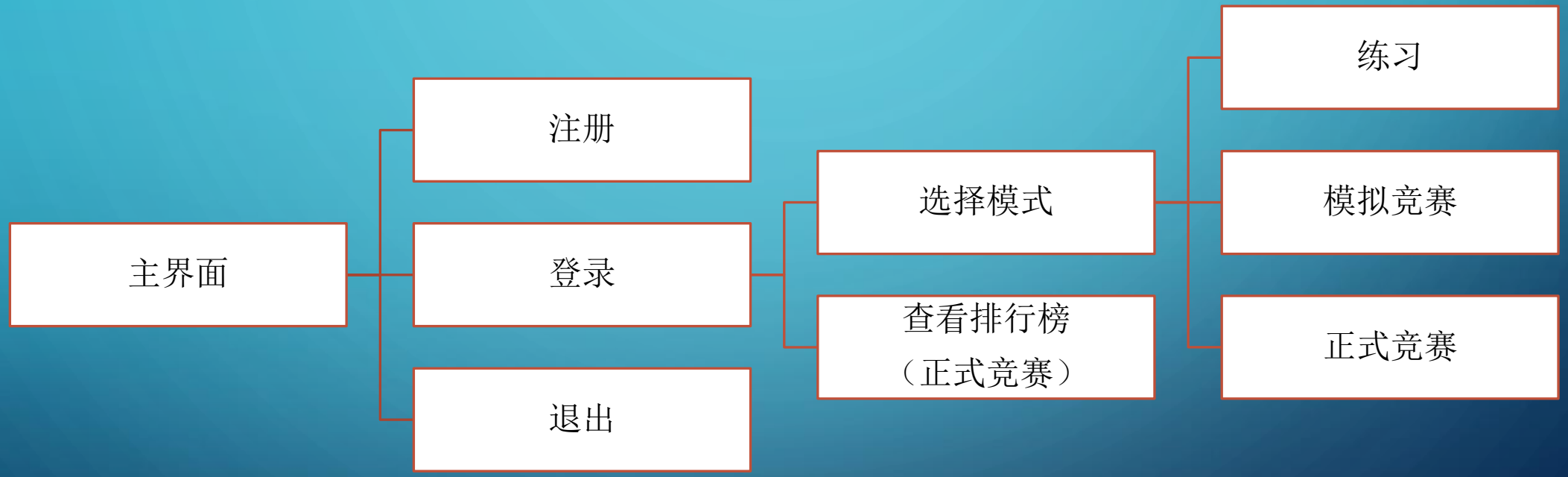
- 用户注册（具有唯一性的用户名、密码），用户登录，退出程序
- 选择模式：练习，模拟竞赛，正式竞赛
  - 练习模式
  - 模拟竞赛模式
  - 正式竞赛模式
- 查看正式竞赛的排行榜

程序：框架设计 & 主要功能设计

一、流程设计

二、功能实现

流程设计



（1）用户注册、登录和退出程序

- 使用EasyX创建一个窗口界面：
  - 用户名和密码的输入框
  - 注册、登录和退出按钮
- 当用户点击按钮时，
  - 注册：获取输入框中的用户名和密码，将其保存文件中。
    - 用户名和密码的唯一性
    - 密码的合法性
  - 登录：
    - 验证输入框中的用户名和密码是否与文件中的匹配。
    - 如果匹配成功，进入用户登录后的功能界面。
  - 退出：关闭程序或返回到登陆界面。

```
// 用户结构体
struct User
{
    string username;    // 用户名
    string password;    // 密码
};

// 检查用户名否已存在（唯一性）
bool isUsernameExisted(const vector<User>& users, const string& username)
{
    for (const User& user : users)
    {
        if (user.username == username)
            return true;
    }
    return false;
}
// 检查密码否已存在（唯一性）
bool isPasswordExisted(const vector<User>& users, const string& password)
{
    for (const User& user : users)
    {
        if (user.password == password)
            return true;
    }
    return false;
}
```

（2）用户登录后的功能

- 使用EasyX创建一个窗口界面：
  - 选择模式
  - 查看正式竞赛的排行榜
- 当用户点击按钮时，
  - 选择模式：进入相应的模式
    - 练习
    - 模拟竞赛
    - 正式竞赛
  - 查看正式竞赛的排行榜

（3）练习模式

- 准备题库数据：党史知识竞赛题库.txt
  - 预处理数据文件文件。
  - 创建一个包含多个题目和正确答案的数据结构，存储每个题目的信息。
  - 每个题目包括题目内容、选项和正确答案，可以使用结构体或类来表示。
- 创建EasyX窗口界面：
  - 题目内容 & 选项 —— 供用户选择答案。
  - 使用文本框或标签来显示题目内容，使用按钮或单选框来表示选项供用户选择。

- 显示题目和选项：
  - 在窗口界面中，根据题库数据随机选择一个题目，将题目内容显示在文本框或标签中。
  - 将题目的选项显示为按钮或单选框的形式，供用户选择答案。
- 检查答案：
  - 当用户选择了一个选项后，获取用户的选择，并与题目的正确答案进行比较。
  - 如果用户选择的答案与正确答案一致，显示答案正确的提示信息。
  - 如果用户选择的答案与正确答案不一致，显示答案错误的提示信息，并显示正确答案。
- 检查答案：
  - 在完成一道题目后，弹出一个对话框询问用户是否继续练习。
  - 根据用户的选择，决定是否继续显示下一道题目。
  - 如果用户选择继续，回到步骤3，显示下一道题目。
  - 如果用户选择退出，结束练习模式。

```cpp
// 结构体表示题目
struct Question {
    string content;          // 题目内容
    vector<string> options;  // 选项
    int correctAnswer;       // 正确答案的索引
};

// 显示题目和选项
void displayQuestion(const Question& question) {
    // 在窗口界面中显示题目内容和选项供用户选择
    // 使用EasyX的相关函数进行界面绘制和显示
}

// 检查答案
bool checkAnswer(const Question& question, int userAnswer) {
    return (userAnswer == question.correctAnswer);
}
```

```cpp
// 创建窗口界面并初始化EasyX图形环境

// 随机选择一道题目
int questionIndex = rand() % questionBank.size();
const Question& currentQuestion = questionBank[questionIndex];

// 显示题目和选项
displayQuestion(currentQuestion);

// 等待用户选择答案
int userAnswer = -1;
// 获取用户的选择

// 检查答案并显示结果
if (checkAnswer(currentQuestion, userAnswer)) {
    // 显示答案正确的提示信息
}
else {
    // 显示答案错误的提示信息，并显示正确答案
}

// 弹出对话框询问用户是否继续练习
// 根据用户的选择，决定是否继续显示下一道题目

// 关闭EasyX图形环境，释放资源
closegraph();
```

（4）模拟竞赛模式

- 准备题库数据：党史知识竞赛题库.txt
  - 预处理数据文件文件。
  - 创建一个包含多个题目和正确答案的数据结构，存储每个题目的信息。
  - 每个题目包括题目内容、选项和正确答案，可以使用结构体或类来表示。
- 创建EasyX窗口界面：
  - 题目内容 & 选项 —— 供用户选择答案。
  - 使用文本框或标签来显示题目内容，使用按钮或单选框来表示选项供用户选择。

- 随机抽取题目组成竞赛试卷：
  - 在管理员设定的时间限制内，从题库中随机抽取一定数量的题目，组成竞赛试卷。
  - 可以使用随机数生成器来选择题目，确保每个题目的抽取概率相同。
- 显示竞赛试卷中的题目和选项：
  - 在窗口界面中，按照竞赛试卷的顺序显示题目和选项供用户选择答案。
  - 可以使用循环遍历竞赛试卷中的题目，并在界面中逐个显示题目内容和选项。
- 用户完成所有题目后计算评分：
  - 用户完成所有题目的答题过程中，记录用户的选择和每个题目的正确答案。
  - 在竞赛结束后，根据用户的答题情况计算评分。
  - 可以根据正确答题数量、错误答题数量或其他评分标准来计算用户的得分。
- 显示错误的题目及其正确答案：
  - 在竞赛结束后，根据用户的答题情况，显示错误的题目及其正确答案。
  - 可以使用对话框、文本框或标签来显示错误的题目和正确答案。

```cpp
// 随机抽取题目组成竞赛试卷
vector<Question> generateExamPaper(int numQuestions) {
    vector<Question> examPaper;

    // 使用随机数生成器从题库中随机抽取题目，组成竞赛试卷
    // 确保每个题目的抽取概率相同

    return examPaper;
}

// 显示竞赛试卷中的题目和选项
void displayExamPaper(const vector<Question>& examPaper) {
    // 在窗口界面中按照竞赛试卷的顺序显示题目和选项供用户选择
    // 使用EasyX的相关函数进行界面绘制和显示
}

// 用户完成所有题目后计算评分
int calculateScore(const vector<Question>& examPaper, const vector<int>& userAnswers) {
    int numCorrect = 0;

    // 根据用户的答题情况计算评分
    // 可以根据正确答题数量、错误答题数量或其他评分标准来计算得分

    return numCorrect;
}

// 显示错误的题目及其正确答案
void displayIncorrectAnswers(const vector<Question>& examPaper, const vector<int>& userAnswers) {
    // 在竞赛结束后，根据用户的答题情况，显示错误的题目及其正确答案
    // 可以使用对话框、文本框或标签来显示错误的题目和正确答案
}
```

```cpp
// 创建窗口界面

// 管理员设定的时间限制
int timeLimit = 30;  // 假设为30秒

// 生成竞赛试卷
vector<Question> examPaper = generateExamPaper(5);  // 假设竞赛试卷包含5道题目

// 显示竞赛试卷中的题目和选项
displayExamPaper(examPaper);

// 等待用户完成所有题目
// 记录用户的选择和每个题目的正确答案

// 计算评分
int score = calculateScore(examPaper, userAnswers);

// 显示错误的题目及其正确答案
displayIncorrectAnswers(examPaper, userAnswers);

// 关闭EasyX图形环境，释放资源
closegraph();
```

（5）正式竞赛模式

- 准备题库数据：党史知识竞赛题库.txt
  - 预处理数据文件文件。
  - 创建一个包含多个题目和正确答案的数据结构，存储每个题目的信息。
  - 每个题目包括题目内容、选项和正确答案，可以使用结构体或类来表示。
- 创建EasyX窗口界面：
  - 题目内容 & 选项 —— 供用户选择答案。
  - 使用文本框或标签来显示题目内容，使用按钮或单选框来表示选项供用户选择。

- 随机抽取题目组成竞赛试卷：
  - 在管理员设定的时间限制内，从题库中随机抽取一定数量的题目，组成竞赛试卷。
  - 可以使用随机数生成器来选择题目，确保每个题目的抽取概率相同。
- 显示竞赛试卷中的题目和选项：
  - 在窗口界面中，按照竞赛试卷的顺序显示题目和选项供用户选择答案。
  - 可以使用循环遍历竞赛试卷中的题目，并在界面中逐个显示题目内容和选项。
- 用户完成作答或者时间结束后计算评分：
  - 用户完成所有题目的答题过程中，记录用户的选择和每个题目的正确答案。
  - 在竞赛结束后，根据用户的答题情况计算评分。
  - 可以根据正确答题数量、错误答题数量或其他评分标准来计算用户的得分。
- 给出在所有结果中的排名：
  - 将所有参与竞赛的用户的评分进行排名，根据得分高低确定排名顺序。
  - 可以根据得分进行排序，并显示用户的排名。

```cpp
// 给出在所有结果中的排名
int calculateRank(const vector<int>& scores, int userScore) {
    // 对所有参与竞赛的用户的评分进行排名，根据得分高低确定排名顺序
    // 可以根据得分进行排序，并返回用户的排名

    return rank;
}
```

（6）查看排行榜（正式竞赛）

- 准备成绩和时间数据：
  - 创建一个数据结构来存储每次正式竞赛的成绩和时间信息。
  - 每次竞赛的数据包括用户ID、成绩和用时等信息，可以使用结构体或类来表示。
- 创建EasyX窗口界面：
  - 显示排行榜。
  - 使用表格、文本框或标签来显示排行榜中的结果。

```
// 结构体表示每次竞赛的数据
struct CompetitionData {
    string userID;    // 用户ID
    int score;        // 成绩
    int timeTaken;    // 用时（假设以秒为单位）
};

// 创建EasyX窗口界面

// 对成绩进行排序的比较函数
bool compareData(const CompetitionData& data1, const CompetitionData& data2) {
    // 根据规定的条件进行比较，例如按照答题正确数量、用时等进行排序

    return true;  // 根据实际情况返回比较结果
}
```

- 对成绩进行排序：
  - 根据规定的条件（例如答题正确数量、用时等），对竞赛成绩进行排序。
  - 使用sort进行排序。
- 显示排行榜结果：
  - 在窗口界面中按照规定的顺序显示排行榜中的结果。
  - 循环遍历排行榜数据，并在界面中逐个显示用户ID、成绩和用时等信息。

```
// 创建窗口界面

// 管理员保存的竞赛成绩和时间数据
vector<CompetitionData> competitionData = {
    { "User1", 10, 60 },
    { "User2", 15, 70 },
    // 添加更多数据...
};

// 对成绩进行排序
sort(competitionData.begin(), competitionData.end(), compareData);

// 显示排行榜结果
displayLeaderboard(competitionData);

// 关闭EasyX图形环境，释放资源
closegraph();
```

结束

谢谢聆听！