# 实验9：文件

## 【实验目的及要求】

1．掌握C语言的文件操作

2. 完成实验报告

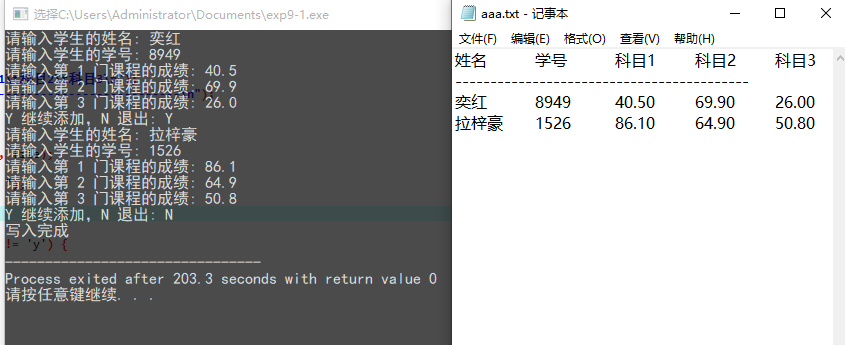
## 【实验内容】

1.P237 3， 5

2.编程，读入题3所建立的文本文件中所有学生的信息，然后按照平均成绩从高到低输出

请将代码附在下面：

# P237 3



1. #include <stdio.h>
2. #include <stdlib.h>
4. **struct** Student {
5. **char** name[50];
6. **char** id[11];
7. **float** scores[3];
8. };
10. **void** inputStudentInfo(**struct** Student \*student);
11. **void** writeStudentToFile(**const** **struct** Student \*student, **FILE** \*file);
13. **int** main() {
14. **struct** Student student;
15. **FILE** \*file = fopen("e:\\aaa.txt", "w");
17. **if** (file == NULL) {
18. printf("无法打开文件！");
19. **return** EXIT\_FAILURE;
20. }
22. fprintf(file, "姓名\t学号\t科目1\t科目2\t科目3\n");
23. fprintf(file, "------------------------------------------\n");
25. **while** (1) {
26. inputStudentInfo(&student);
27. writeStudentToFile(&student, file);
29. printf("Y 继续添加，N 退出: ");
30. **char** choice;
31. scanf(" %c", &choice);
33. **if** (choice != 'Y' && choice != 'y') {
34. **break**;
35. }
36. }
38. fclose(file);
39. printf("写入完成\n");
41. **return** EXIT\_SUCCESS;
42. }
44. **void** inputStudentInfo(**struct** Student \*student) {
45. printf("请输入学生的姓名: ");
46. scanf("%s", student->name);
48. printf("请输入学生的学号: ");
49. scanf("%s", student->id);
51. **for** (**int** i = 0; i < 3; i++) {
52. printf("请输入第 %d 门课程的成绩: ", i + 1);
53. scanf("%f", &student->scores[i]);
54. }
55. }
57. **void** writeStudentToFile(**const** **struct** Student \*student, **FILE** \*file) {
58. fprintf(file, "%s\t%s\t%.2f\t%.2f\t%.2f\n",
59. student->name, student->id,
60. student->scores[0], student->scores[1], student->scores[2]);
61. }

# P237 5

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <string.h>
5. #define MAX\_NAME\_LENGTH 50
6. #define MAX\_ID\_LENGTH 11
8. **typedef** **struct** {
9. **char** name[MAX\_NAME\_LENGTH];
10. **char** id[MAX\_ID\_LENGTH];
11. **float** scores[3];
12. } Student;
14. **void** readStudentsFromFile(Student \*\*students, **int** \*count, **const** **char** \*filename);
15. **void** writeStudentsToFile(**const** Student students[], **int** count, **const** **char** \*filename);
16. **int** findStudentById(**const** Student students[], **int** count, **const** **char** \*id);
18. **int** main() {
19. Student \*students = NULL;
20. **int** studentCount = 0;
21. **const** **char** \*filename = "e:\\aaa.txt";
23. readStudentsFromFile(&students, &studentCount, filename);
25. **while** (1) {
26. **char** idToDelete[MAX\_ID\_LENGTH];
27. printf("请输入要删除的学生学号: ");
28. scanf("%s", idToDelete);
30. **int** index = findStudentById(students, studentCount, idToDelete);
31. **if** (index != -1) {
32. **for** (**int** i = index; i < studentCount - 1; i++) {
33. students[i] = students[i + 1];
34. }
35. studentCount--;
36. writeStudentsToFile(students, studentCount, filename);
37. printf("学生信息已成功删除。\n");
38. } **else** {
39. printf("未找到学生信息！\n");
40. }
42. printf("Y 继续删除，N 退出: ");
43. **char** choice;
44. scanf(" %c", &choice);
46. **if** (choice != 'Y' && choice != 'y') {
47. **break**;
48. }
49. }

52. free(students);
54. **return** EXIT\_SUCCESS;
55. }
57. **void** readStudentsFromFile(Student \*\*students, **int** \*count, **const** **char** \*filename) {
58. **FILE** \*file = fopen(filename, "r");
59. **if** (file == NULL) {
60. perror("无法打开文件");
61. **return**;
62. }
64. **char** line[256];
65. **int** foundHeader = 0;
67. **while** (fgets(line, **sizeof**(line), file)) {
68. **if** (!foundHeader) {
69. foundHeader = 1;
70. **continue**;
71. }

74. line[strcspn(line, "\n")] = '\0';
76. \*students = (Student \*)realloc(\*students, (\*count + 1) \* **sizeof**(Student));
77. **if** (\*students == NULL) {
78. perror("内存分配失败");
79. fclose(file);
80. **return**;
81. }
83. sscanf(line, "%[^,\t]\t%[^,\t]\t%f\t%f\t%f",
84. (\*students)[\*count].name, (\*students)[\*count].id,
85. &(\*students)[\*count].scores[0], &(\*students)[\*count].scores[1], &(\*students)[\*count].scores[2]);
86. (\*count)++;
87. }
89. fclose(file);
90. }
92. **void** writeStudentsToFile(**const** Student students[], **int** count, **const** **char** \*filename) {
93. **FILE** \*file = fopen(filename, "w");
94. **if** (file == NULL) {
95. perror("无法打开文件");
96. **return**;
97. }
99. fprintf(file, "姓名\t学号\t科目1\t科目2\t科目3\n");
101. **for** (**int** i = 0; i < count; i++) {
102. fprintf(file, "%s\t%s\t%.2f\t%.2f\t%.2f\n",
103. students[i].name, students[i].id,
104. students[i].scores[0], students[i].scores[1], students[i].scores[2]);
105. }
107. fclose(file);
108. }
110. **int** findStudentById(**const** Student students[], **int** count, **const** **char** \*id) {
111. **for** (**int** i = 0; i < count; i++) {
112. **if** (strcmp(students[i].id, id) == 0) {
113. **return** i;
114. }
115. }
116. **return** -1;
117. }

# 2.

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <string.h>
5. #define MAX\_NAME\_LENGTH 50
6. #define MAX\_ID\_LENGTH 11
8. **typedef** **struct** {
9. **char** name[MAX\_NAME\_LENGTH];
10. **char** id[MAX\_ID\_LENGTH];
11. **float** scores[3];
12. } Student;
14. **void** readStudentsFromFile(Student \*\*students, **int** \*count, **const** **char** \*filename);
15. **int** findStudentById(**const** Student students[], **int** count, **const** **char** \*id);
16. **float** calculateAverageScore(**const** Student \*student);
17. **void** bubbleSortByAverageScore(Student students[], **int** count);
19. **int** main() {
20. Student \*students = NULL;
21. **int** studentCount = 0;
22. **const** **char** \*filename = "e:\\aaa.txt";
24. readStudentsFromFile(&students, &studentCount, filename);
26. bubbleSortByAverageScore(students, studentCount);
28. printf("姓名\t学号\t科目1\t科目2\t科目3\t平均分\n");
29. **for** (**int** i = 0; i < studentCount; i++) {
30. **float** avgScore = calculateAverageScore(&students[i]);
31. printf("%s\t%s\t%.2f\t%.2f\t%.2f\t%.2f\n",
32. students[i].name, students[i].id,
33. students[i].scores[0], students[i].scores[1], students[i].scores[2], avgScore);
34. }
36. free(students);
38. **return** EXIT\_SUCCESS;
39. }
41. **void** readStudentsFromFile(Student \*\*students, **int** \*count, **const** **char** \*filename) {
42. **FILE** \*file = fopen(filename, "r");
43. **if** (file == NULL) {
44. perror("无法打开文件");
45. **return**;
46. }
48. **char** line[256];
49. **int** foundHeader = 0;
51. **while** (fgets(line, **sizeof**(line), file)) {
52. **if** (!foundHeader) {
53. foundHeader = 1;
54. fgets(line, **sizeof**(line), file);
55. **continue**;
56. }
58. line[strcspn(line, "\n")] = '\0';
60. \*students = (Student \*)realloc(\*students, (\*count + 1) \* **sizeof**(Student));
61. **if** (\*students == NULL) {
62. perror("内存分配失败");
63. fclose(file);
64. **return**;
65. }
67. sscanf(line, "%[^,\t]\t%[^,\t]\t%f\t%f\t%f",
68. (\*students)[\*count].name, (\*students)[\*count].id,
69. &(\*students)[\*count].scores[0], &(\*students)[\*count].scores[1], &(\*students)[\*count].scores[2]);
70. (\*count)++;
71. }
73. fclose(file);
74. }
76. **int** findStudentById(**const** Student students[], **int** count, **const** **char** \*id) {
77. **for** (**int** i = 0; i < count; i++) {
78. **if** (strcmp(students[i].id, id) == 0) {
79. **return** i;
80. }
81. }
82. **return** -1;
83. }
85. **float** calculateAverageScore(**const** Student \*student) {
86. **return** (student->scores[0] + student->scores[1] + student->scores[2]) / 3.0;
87. }
89. **void** bubbleSortByAverageScore(Student students[], **int** count) {
90. **for** (**int** i = 0; i < count - 1; i++) {
91. **for** (**int** j = 0; j < count - i - 1; j++) {
92. **float** avgScoreJ = calculateAverageScore(&students[j]);
93. **float** avgScoreJPlusOne = calculateAverageScore(&students[j + 1]);
94. **if** (avgScoreJ < avgScoreJPlusOne) {
95. Student temp = students[j];
96. students[j] = students[j + 1];
97. students[j + 1] = temp;
98. }
99. }
100. }
101. }