**郑州轻工业大学**

**《操作系统》课程****实验报告**

实验名称：短作业优先调度算法

姓 名： 张海水

学 号： 522107260429

地 点： 四号楼401

指导老师： 李志刚

专业班级： z移动软件21-04

实验成绩：

1. 实验目的：

目的：了解并掌握作业调度的功能，熟悉并掌握各种作业调度算法。

任务：模拟实现先来先服务或者短作业优先调度算法。

二、实验内容：

1、实验内容

模拟实现SJF调度。

设置作业体：作业名，作业的到达时间，服务时间，作业状态（W——等待,R——运行,F——完成），作业间的链接指针；

作业初始化：由用户输入作业名、服务时间、到达时间进行初始化，同时，初始化作业的状态为W。

显示函数：在作业调度前、调度中和调度后进行显示。

排序函数：对等待状态的作业按照调度算法排序（不同的调度算法排序方式不同），注意考虑到达时间。

调度函数：每次从等待队列队首调度已到达的适合的作业执行，状态变化。当服务结束时，状态变为F。

删除函数：撤销状态为F的作业。

2、实验要求

（1）测试数据可以随即输入或从文件中读入；

（2）必须要考虑到作业的到达时间；

（3）最终能够计算每一个作业的周转时间、带权周转时间。

1. 实验代码

显示部分(html):

1. <template id="myapp">
2. <div>
3. <table id="zuoyeQue">
4. <thead>
5. <tr>
6. <th style="text-align: center;" colspan="3">原始作业</th>
7. </tr>
8. </thead>
9. <tbody>
10. <tr>
11. <th>作业名称</th>
12. <th>作业进入时间</th>
13. <th>作业运行时间</th>
14. </tr>
15. <tr v-for="(item, index) in zuoye" :key="index">
16. <td><input type="text" v-model="item.name"></td>
17. <td><input type="text" v-model="item.enterTime"></td>
18. <td><input type="text" v-model="item.processTime"></td>
19. </tr>
20. </tbody>
21. <tr>
22. <td colspan="3"><button @click="addZuoye">添加作业(添加后数据自动同步)</button></td>
23. </tr>
24. </table>
25. <br>
26. <div id="handle">
27. <p>记录的运行时间:{{totalProcessTime}}</p>
28. <button onclick="javascript:location.reload()">
29. 运行完毕后请重置
30. </button>
31. <button @click="sortZuoye">
32. 单步运行
33. </button>
34. <button @click="sortZuoyeAll">
35. 直接运行至结果
36. </button>
37. <br>
38. <br>
39. </div>
40. <br>
41. <div id="guocheng" v-if="showGuocheng">
42. <table id="sortedZuoyeQue">
43. <thead>
44. <tr>
45. <th style="text-align: center;" colspan="3">待进行处理的作业(通过进入时间排序)</th>
46. </tr>
47. </thead>
48. <tbody>
49. <tr>
50. <th>作业名称</th>
51. <th>作业进入时间</th>
52. <th>作业运行时间</th>
53. </tr>
54. <tr v-for="(item, index) in sortedZuoye" :key="index">
55. <td>{{item.name}}</td>
56. <td>{{item.enterTime}}</td>
57. <td>{{item.processTime}}</td>
58. </tr>
59. <tr v-if="sortedZuoye.length==0">
60. <th>
61. <h5>此时无进程在队列!</h5>
62. </th>
63. </tr>
64. </tbody>
65. </table>
66. <br><br>
67. <table id="preparedQue">
68. <thead>
69. <tr>
70. <th style="text-align: center;" colspan="3">已到达的作业(通过处理时间排序)</th>
71. </tr>
72. </thead>
73. <tbody>
74. <tr>
75. <th>作业名称</th>
76. <th>作业进入时间</th>
77. <th>作业运行时间</th>
78. </tr>
79. <tr v-for="(item, index) in preparedZuoye" :key="index">
80. <td>{{item.name}}</td>
81. <td>{{item.enterTime}}</td>
82. <td>{{item.processTime}}</td>
83. </tr>
84. <tr v-if="preparedZuoye.length==0">
85. <th>
86. <h5>目前队列为空!</h5>
87. </th>
88. </tr>
89. </tbody>
90. </table>
91. <br>
92. <table id="processsingQue">
93. <thead>
94. <tr>
95. <th style="text-align: center;" colspan="4">正在进行的作业</th>
96. </tr>
97. </thead>
98. <tbody>
99. <tr>
100. <th>作业名称</th>
101. <th>作业进入时间</th>
102. <th>作业运行时间</th>
103. <th>作业开始时间</th>
104. </tr>
105. <tr v-if="processingZuoye!=null">
106. <td>{{processingZuoye.name}}</td>
107. <td>{{processingZuoye.enterTime}}</td>
108. <td>{{processingZuoye.processTime}}</td>
109. <td>{{processingZuoye.startTime}}</td>
110. </tr>
111. </tbody>
112. </table>
113. </div>
114. <br>
115. <table id="finishedQue">
116. <thead>
117. <tr>
118. <th style="text-align: center;" colspan="7">已完成的作业</th>
119. </tr>
120. </thead>
121. <tbody>
122. <tr>
123. <th>作业名称</th>
124. <th>作业进入时间</th>
125. <th>作业运行时间</th>
126. <th>作业开始时间</th>
127. <th>作业完成时间</th>
128. <th>作业周转时间</th>
129. <th>作业带权周转时间</th>
130. </tr>
131. <tr v-for="(item, index) in finishedZuoye" :key="index">
132. <td>{{item.name}}</td>
133. <td>{{item.enterTime}}</td>
134. <td>{{item.processTime}}</td>
135. <td>{{item.startTime}}</td>
136. <td>{{item.finishedTime}}</td>
137. <td>{{item.turnaroundTime}}</td>
138. <td>{{item.weightedTurnaroundTime}}</td>
139. </tr>
140. <tr v-if="finishedZuoye.length==0">
141. <th>
142. <h5>目前队列为空!</h5>
143. </th>
144. </tr>
145. *<!-- <template v-if="finishedZuoye.lenth<1">*
147. </template> -->
148. </tbody>
149. </table>
150. </div>
151. </template>

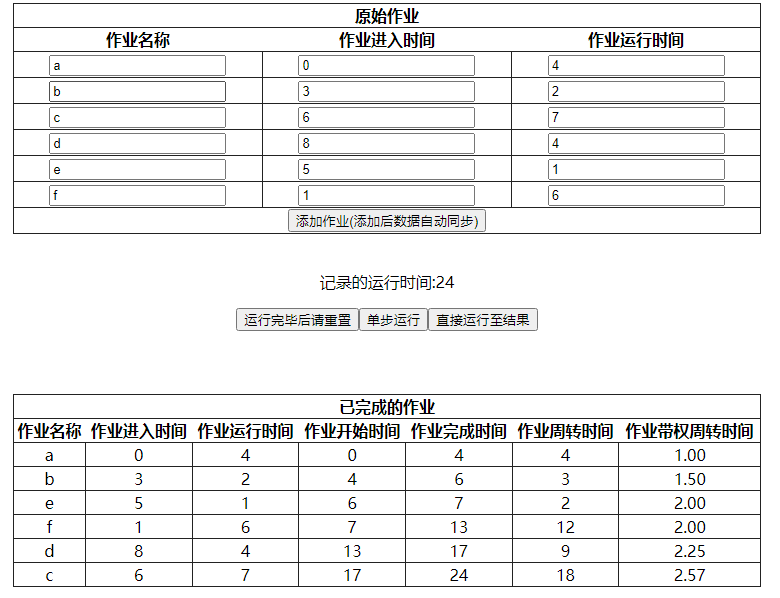
计算部分(vuejs):

1. const App = {
2. template: '#myapp',
3. data() {
4. return {
5. zuoye: [
6. {
7. name: 'a',
8. enterTime: 0,
9. processTime: 4,
10. },
11. {
12. name: 'b',
13. enterTime: 3,
14. processTime: 2,
15. },
16. {
17. name: 'c',
18. enterTime: 6,
19. processTime: 7,
20. },
21. {
22. name: 'd',
23. enterTime: 8,
24. processTime: 4,
25. },
26. {
27. name: 'e',
28. enterTime: 5,
29. processTime: 1,
30. },
31. {
32. name: 'f',
33. enterTime: 1,
34. processTime: 6,
35. },
36. ],
37. sortedZuoye: [
38. ],
39. finishedZuoye: [
40. ],
41. preparedZuoye: [
42. ],
43. processingZuoye: null,
44. totalProcessTime: 0,
45. processingTimes: 0,
46. showGuocheng: false
47. }
48. },
49. methods: {
50. *//添加一跳作业记录*
51. addZuoye() {
52. this.zuoye.push({
53. name: 'default',
54. enterTime: 0,
55. processTime: 0,
56. })
57. },
58. sortZuoyeAll() {
59. *//循环单步执行*
60. this.sortZuoye()
61. while (this.sortedZuoye.length != 0 || this.preparedZuoye.length != 0 || this.processingZuoye != null) {
62. this.sortZuoye()
63. }
64. this.showGuocheng = false
65. },
66. sortZuoye() {
67. this.showGuocheng = true
68. *//如果运行次数为0*
69. if (this.processingTimes == 0) {
70. *//按照zuoye的进入时间排序*
71. *//将zuoye深度拷贝一份,并赋值给sortedZuoye*
72. this.sortedZuoye = JSON.parse(JSON.stringify(this.zuoye));
73. console.log(this.sortedZuoye);
74. this.sortedZuoye.sort((a, b) => {
75. return a.enterTime - b.enterTime
76. })
77. *//加入第一条作业到准备队列*
78. this.preparedZuoye.push(this.sortedZuoye.shift())
79. } else {
80. if (this.processingZuoye != null) {
81. this.processing()
82. } else {
83. this.showProcessingZuoye()
84. }
85. }
86. this.processingTimes++
87. },
88. showProcessingZuoye() {
89. console.log("进入showProcessingZuoye");
90. *//添加数据的进行时刻*
91. this.preparedZuoye[0].startTime = this.totalProcessTime
92. *//将准备中的数据第一条添加到正在进行作业中*
93. this.processingZuoye = this.preparedZuoye.shift()
94. },
95. *//主进程*
96. processing() {
97. console.log("进入processing");
98. *//将完成数据的最后一条的运行时间加上*
99. this.totalProcessTime += this.processingZuoye.processTime
100. *//添加数据的完成时刻*
101. this.processingZuoye.finishedTime = this.totalProcessTime
102. *//周转时间=作业完成时刻-作业到达时刻；*
103. this.processingZuoye.turnaroundTime = this.processingZuoye.finishedTime - this.processingZuoye.enterTime
104. *//带权周转时间=周转时间/运行时间；*
105. this.processingZuoye.weightedTurnaroundTime = (this.processingZuoye.turnaroundTime / this.processingZuoye.processTime).toFixed(2)
106. *//将数据添加到已完成数据列表中*
107. this.finishedZuoye.push(this.processingZuoye)
108. this.processingZuoye = null
109. *//创建将小于准备时间的队列*
110. let filteredArr = this.sortedZuoye.filter(item => {
111. return item.enterTime <= this.totalProcessTime
112. })
113. *//将小于准备时间的队列加入到准备中的队列*
114. filteredArr.map(ele => {
115. this.preparedZuoye.push(ele)
116. })
117. *//将filteredArr的队列从sortedZuoye中删除*
118. this.sortedZuoye = this.sortedZuoye.filter(item => {
119. return item.enterTime > this.totalProcessTime
120. })
121. *//将preparedZuoye按照processTime排序*
122. this.preparedZuoye.sort((a, b) => {
123. return a.processTime - b.processTime
124. })
125. }
126. },
127. }
128. Vue.createApp(App).mount('#app')
129. 实验结果

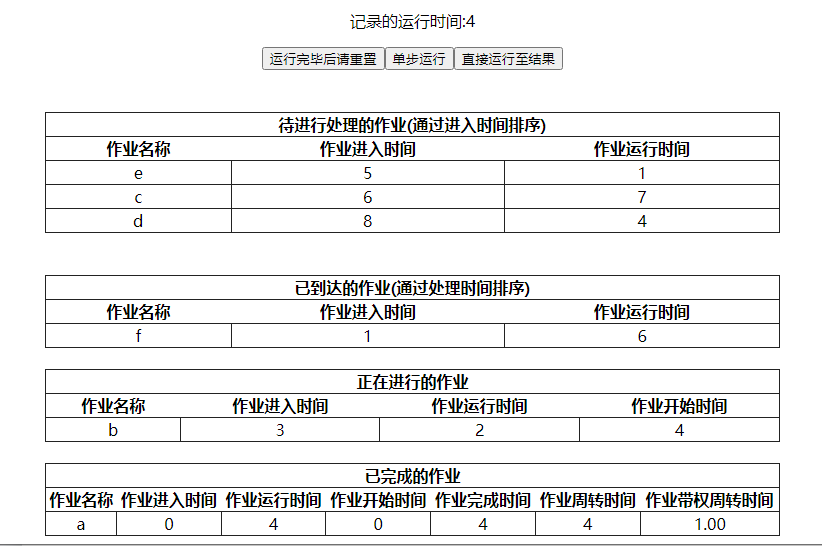
直接点击sjf算法实现.html即可进入页面

运行结果截图:

1.直接运行至结果



1. 单步运行



五、实验总结（含心得体会）

答:这次的sjf实验课对于我们理解进程的调度非常的有用,一开始完全无从下手也不知道如何进行调度,在一步步的尝试和搜索中,慢慢了解到具体的过程是什么样的,然后开始思考如何运用代码将这种过程展示出来,在这中间也是经历了很多的错误与bug,有时候代码的顺序不对就没法达到想要的效果.