C++프로그래밍및실습

레스토랑 청소 우선 순위 프로그램

진척 보고서 #3

제출일자:2024-12-15
제출자명:장유은
제출자학번:234199

1. 프로젝트 목표 (16 pt)

1) 배경 및 필요성 (14 pt)

레스토랑에서 손님이 퇴점하면 해당 자리를 청소해야 함. 하지만 레스토랑이 넓고 손님이 많아 퇴점한 자리인지 육안으로 확인이 어려움. 손님이 퇴점한 뒤 청소해야 할 자리를 알려주는 프로그램이 필요함.

2) 프로젝트 목표

현재 테이블에 손님이 있는지, 퇴점했는지 확인하고 청소해야 할 테이블을 알려주는 프로그램을 만드는 것을 목표로 함.

3) 차별점

퇴점한 테이블이 여러 개일 경우, 손님들의 선호도가 높은 좌석의 테이블을 먼저 청소하도록 우선순위를 알려줌.

2. 기능 계획

1) 기능 1 테이블 클래스, 입/퇴점 함수

- 설명 : 테이블 클래스를 만들어 번호, 점유 여부, 선호도를 입력받는다. 입점한 테이블을 객체배열로 저장하는 함수를 만들고 퇴점한 테이블도 객체배열로 저장하는 함수를 만든다.

2) 기능 2 입력받고 알려주기

- 설명 : 입점을 입력할지, 퇴점을 입력할지, 청소순위를 알고 싶은지 터미널에서

입력받고 알려주는 기능

3) 기능 3 청소 우선순위

- 설명 : 퇴점한 테이블 중 선호도가 높은 테이블을 분석하고 청소의 우선순위를 알려준다.

3. 진척사항

1) 기능 구현

(1) Table 객체

- 설명 : 테이블 번호, 손님 선호도, 점유 여부, 청소 여부 저장하는 멤버 변수를 가지며 상태를 반환하는 멤버함수도 포함하는 클래스

- 적용된 배운 내용 : 클래스, 함수, 조건문

- 코드 스크린샷

```
7   class Table{    // 테이블 클래스
8   public:
9       int tableNumber; // 테이블 번호
10      int preference; // 손님 선호도(높을수록 우선순위 높음)
11      bool occupy;    // 테이블 점유 여부
12      bool clean; // 테이블 청소 여부
13
14      Table(){    // 기본 생성자
15          tableNumber=0;
16          preference=0;
17          occupy=false;
18          clean=false;
19      }
20      Table(int n,int p){
21          tableNumber=n;
22          preference=p;
23          occupy=false;
24          clean=true;
25      }
26
27      //테이블 상태 반환(O,X,_)
28      char getStatus() const{
29          if(occupy) return 'O';  // 점유된 테이블
30          else if (clean) return '_'; // 청소된 테이블
31          else return 'X';    // 퇴점한 테이블
32      }
33  };
```

(2) Table 초기화 함수

- 설명 : 함수를 호출하면 2차원 객체배열에 테이블 번호와 선호도를 초기화시킴

- 적용된 배운 내용 : 객체 배열, 2차원 배열, 함수, 반복문

- 코드 스크린샷

```
4   const int numRows=3;     // 행 개수
5   const int numCols=3;     // 열 개수
```

```
35  // 테이블 초기화
36  void initialTables(Table tables[numRows][numCols]){
37      int tableNumber=1;  // 테이블 번호 시작
38      int preference=9;   // 선호도 시작
39      for (int i=0;i<numRows;++i)
40          for(int j=0;j<numCols;++j)
41              tables[i][j]=Table(tableNumber++,preference--);
42  }
```

(3) Table 상태 출력 함수

- 입출력 : 3X3 배열에 테이블의 상태를 출력해서 보여줌

- 적용된 배운 내용 : 객체 배열, 2차원 배열, 함수, 반복문, 조건문

- 코드 스크린샷

```
44  // 테이블 상태 출력(3X3 배열)
45  void printTables(Table tables[numRows][numCols]){
46      for(int i=0;i<numRows;i++){
47          cout<<"---|---|---"<<endl; // 구분선
48          for(int j=0;j<numCols;j++){
49              cout<<tables[i][j].getStatus(); // Table 객체 상태 출력
50              if(j==numCols-1)
51                  break;
52              cout<<"  |";
53          }
54          cout<<endl;
55      }
56      cout<<"---|---|---"<<endl; //마지막 구분선
57  }
```

(4) 번호를 행과 열로 변환하는 함수

- 입출력 : 테이블 번호를 입력받으면 행, 열로 변환해줌

- 적용된 배운 내용 : 함수

- 코드 스크린샷

```
59  // 번호를 행과 열로 변환
60  void change(int tableNumber,int &row, int &col){
61      row = (tableNumber - 1) / numCols;
62      col = (tableNumber - 1) % numCols;
63  }
```

(5) Table 입점 함수

- 입출력 : 배열과 번호를 입력받고 조건문 확인 후 멘트 출력

- 설명 : 테이블 번호를 입력받으면 행,열로 변환한 후 점유/청소 여부를 확인한다. 입점시 상태도 수정한다.

- 적용된 배운 내용 : 객체 배열, 2차원 배열, 함수, 조건문

- 코드 스크린샷

```
65  // 테이블 입점
66  void entryTable(Table tables[numRows][numCols],int tableNumber){
67      int row, col;
68      change(tableNumber, row, col);
69
70      if(row<0||row>=numRows||col<0||col>=numCols){
71          cout<<"잘못된 테이블 번호입니다."<<endl;
72          return;
73      }
74
75      if(tables[row][col].occupy)
76          cout<<"테이블"<<tableNumber<<"은 이미 입점 중입니다. 다른 테이블을 선택하세요."<<endl;
77      else if(!tables[row][col].clean){
78          cout<<"테이블"<<tableNumber<<"은 청소되지 않았습니다. 먼저 청소를 완료하세요."<<endl;
79      }
80      else{
81          tables[row][col].occupy=true;
82          tables[row][col].clean=false;   // 입점 시 청소 상태 해제
83          cout<<"테이블"<<tableNumber<<" 입점했습니다."<<endl;
84      }
85  }
```

(6) Table 퇴점 함수

- 입출력 : 배열과 번호를 입력받고 조건문 확인 후 멘트 출력

- 설명 : 테이블 번호를 입력받으면 행,열로 변환한 후 점유 여부를 확인한다. 퇴점시 상태도 수정한다.

- 적용된 배운 내용 : 객체 배열, 2차원 배열, 함수, 조건문

- 코드 스크린샷

```
87  // 테이블 퇴점
88  void leaveTable(Table tables[numRows][numCols], int tableNumber){
89      int row, col;
90      change(tableNumber, row, col);
91
92      if(row<0||row>=numRows||col<0||col>=numCols){
93          cout<<"잘못된 테이블 번호입니다."<<endl;
94          return;
95      }
96
97      if(!tables[row][col].occupy)
98          cout<<"테이블"<<tableNumber<<"은 이미 퇴점했습니다."<<endl;
99      else{
100         tables[row][col].occupy=false;
101         cout<<"테이블"<<tableNumber<<" 퇴점했습니다."<<endl;
102     }
103 }
```

(7) Table 청소 함수

- 입출력 : 배열과 번호를 입력받고 조건문 확인 후 멘트 출력

- 설명 : 테이블 번호를 입력받으면 행,열로 변환한 후 점유/청소 여부를 확인한다. 청소 시 상태도 수정한다.

- 적용된 배운 내용 : 객체 배열, 2차원 배열, 함수, 조건문

- 코드 스크린샷

```
105 // 테이블 청소
106 void cleanTable(Table tables[numRows][numCols], int tableNumber){
107     int row, col;
108     change(tableNumber, row, col);
109
110     if(row<0||row>=numRows||col<0||col>=numCols){
111         cout<<"잘못된 테이블 번호입니다."<<endl;
112         return;
113     }
114
115     if(tables[row][col].occupy)
116         cout<<"테이블"<<tableNumber<<"은 손님이 입점해 있는 테이블 입니다."<<endl;
117     else if(tables[row][col].clean)
118         cout<<"테이블"<<tableNumber<<"은 이미 청소한 테이블 입니다."<<endl;
119     else{
120         tables[row][col].clean=true;
121         cout<<"테이블"<<tableNumber<<" 청소했습니다."<<endl;
122     }
123 }
```

(8) 청소 우선순위 함수

- 입출력 : 배열을 입력받고 선호도가 높은 순으로 청소해야 할 테이블을 알려줌

- 설명 : 임시 배열에 퇴점하고 청소되지 않은 테이블을 추가하고 버블 정렬을 이용해 선호도에 따라 정렬한다.

- 적용된 배운 내용 : 객체 배열, 2차원 배열, 함수, 조건문, 반복문

- 코드 스크린샷

```
125  //청소 우선순위(선호도 높은 순으로 출력)
126  void priorityTable(Table tables[numRows][numCols]){
127      // 청소해야 할 테이블 임시 저장할 배열
128      Table tempTables[numRows*numCols];
129      int count=0;
130
131      //퇴점하고 청소되지 않은 테이블만 배열에 추가
132      for(int i=0;i<numRows;i++){
133          for(int j=0;j<numCols;j++){
134              if(!tables[i][j].occupy&&!tables[i][j].clean)
135                  tempTables[count++]=tables[i][j];
136          }
137      }
138
139      if(count==0)
140          cout<<"청소할 테이블이 없습니다."<<endl;
141      else{
142          // 선호도에 따라 정렬 (버블 정렬 사용)
143          for (int i = 0; i < count - 1; ++i) {
144              for (int j = 0; j < count - i - 1; ++j) {
145                  if (tempTables[j].preference < tempTables[j + 1].preference) {
146                      Table temp = tempTables[j];
147                      tempTables[j] = tempTables[j + 1];
148                      tempTables[j + 1] = temp;
149                  }
150              }
151          }
152
153          cout<<"청소 우선순위 : ";
154          for(int i=0;i<count;++i){
155              cout<<"테이블"<<tempTables[i].tableNumber<<" (선호도"<<tempTables[i].preference<<")"
156              if(i<count-1)
157                  cout<<"->";
158          }
159          cout<<endl;
160      }
161  }
```

(9) main 함수

- 설명 : 작동을 원하는 번호를 입력받고 해당 테이블 번호 입력받고 해당 함수 호출, 멘트 출력

- 적용된 배운 내용 : 2차원 배열, switch문, 객체 배열, 함수

- 코드 스크린샷

```
163  int main(){
164      Table tables[numRows][numCols]; // 테이블 입력할 객체 배열
165      initialTables(tables);  // 테이블 초기화
166
167      while(true){
168          printTables(tables);    // 현재 테이블 상태 출력
169
170          int choice;
171          cout<<endl;
172          cout<<"0. 프로그램 종료"<<endl;
173          cout<<"1. 입점 테이블 입력"<<endl;
174          cout<<"2. 퇴점 테이블 입력"<<endl;
175          cout<<"3. 청소한 테이블 입력"<<endl;
176          cout<<"4. 테이블 청소 우선순위"<<endl;
177          cout<<"원하는 번호를 입력하세요:";
178          cin>>choice;
179
180          int tableNumber;    // 입력한 테이블 번호 저장할 변수
181          switch(choice){
182          case 1:
183              cout<<"입점 테이블 번호를 입력하세요:";
184              cin>>tableNumber;
185              entryTable(tables,tableNumber); // 입점 처리 함수
186              break;
187          case 2:
188              cout<<"퇴점 테이블 번호를 입력하세요:";
189              cin>>tableNumber;
190              leaveTable(tables,tableNumber);//퇴점 처리 함수
191              break;
192          case 3:
193              cout<<"청소한 테이블 번호를 입력하세요:";
194              cin>>tableNumber;
195              cleanTable(tables,tableNumber); // 청소 처리 함수
196              break;
197          case 4:
198              priorityTable(tables); // 청소우선순위 함수
199              break;
```

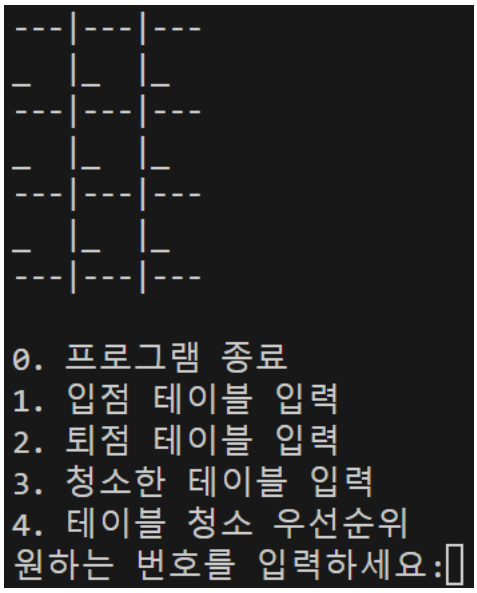
```
200          case 0:
201              cout<<"프로그램을 종료합니다"<<endl;
202              return 0;
203          default:
204              cout<<"잘못된 입력입니다."<<endl;
205              break;
206          }
207      }
208      return 0;
209  }
```

2) 테스트 결과

(1) 테이블 초기화, 상태 출력

- 설명 : 시작했을 때 3X3 배열과 초기 상태로 출력되는지 확인

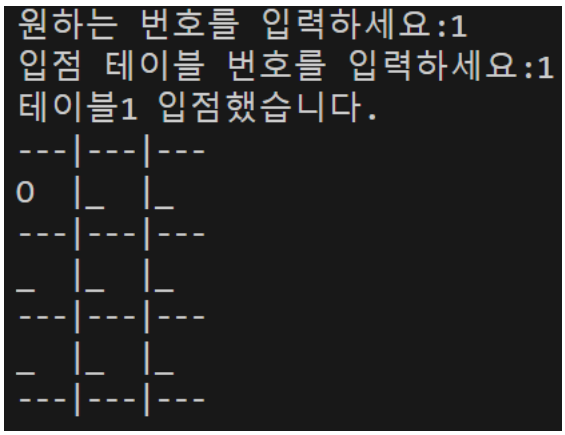
- 테스트 결과 스크린샷

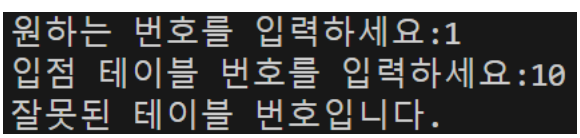


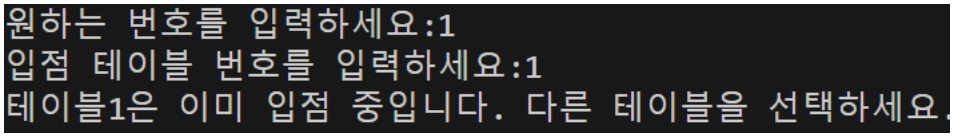
(2) 입점 함수에서 테이블 번호 조건에 따른 출력

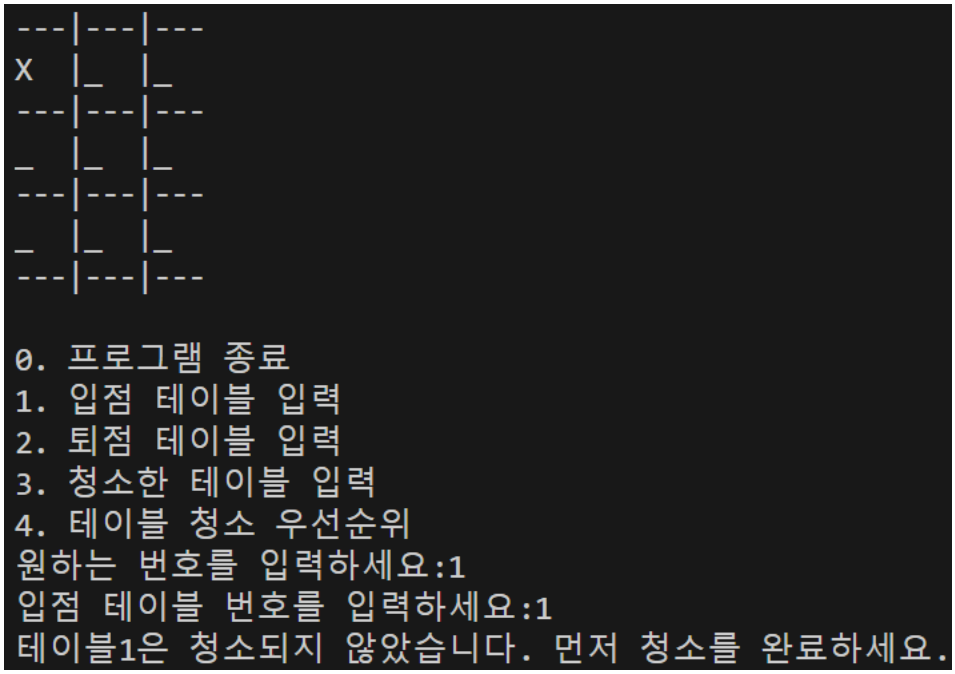
- 설명 : 범위를 벗어난 번호인지, 입점 중인 번호인지, 청소되지 않은 번호인지 확인, 입점했다면 상태 변경해 출력되는지 확인

- 테스트 결과 스크린샷





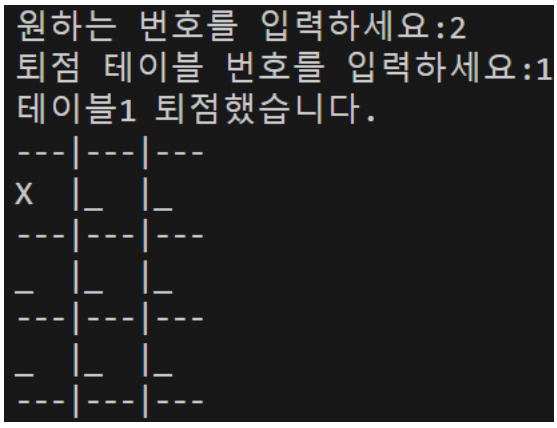


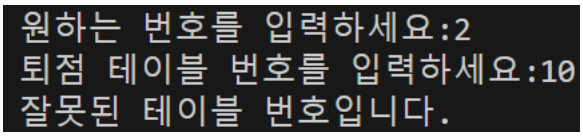


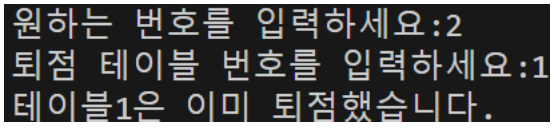
(3) 퇴점 함수에서 테이블 번호 조건에 따른 출력

- 설명 : 범위를 벗어난 번호인지, 이미 퇴점한 번호인지 확인, 퇴점했다면 상태 변경해 출력되는지 확인

- 테스트 결과 스크린샷



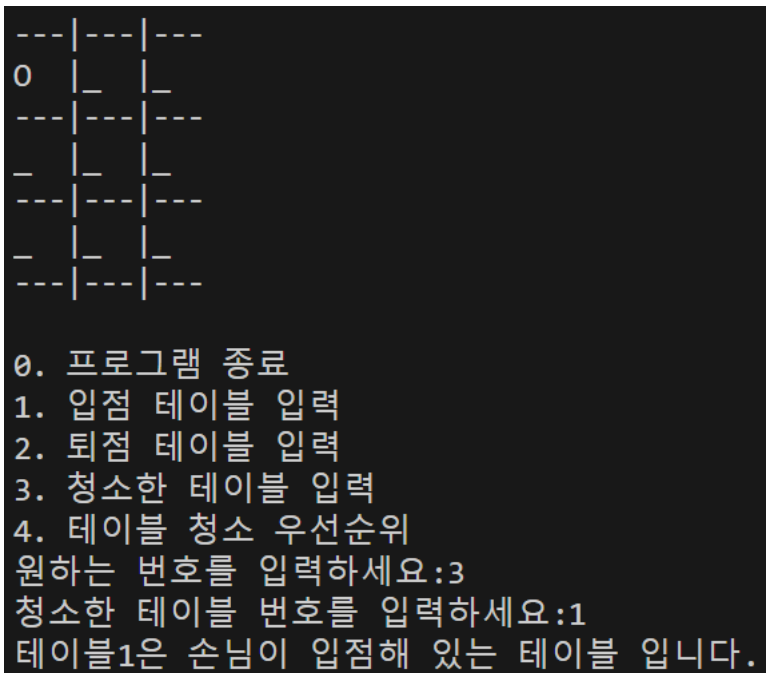


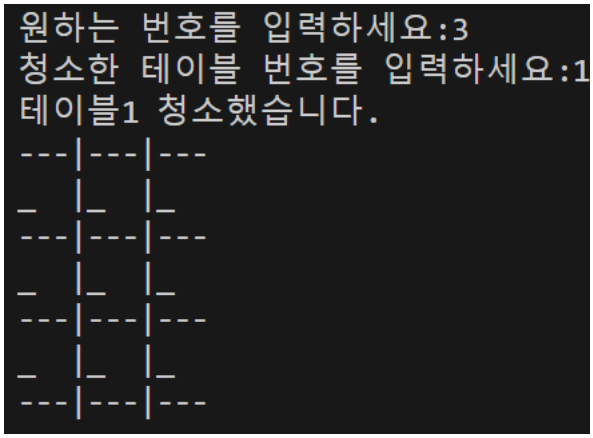


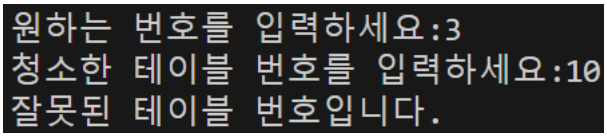
(4) 청소 함수에서 테이블 번호 조건에 따른 출력

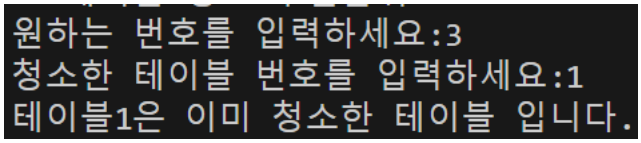
- 설명 : 범위를 벗어난 번호인지, 입점한 번호인지, 이미 청소한 번호인지 확인, 청소했다면 상태 변경해 출력되는지 확인

- 테스트 결과 스크린샷









(4) 테이블 청소 우선순위

- 설명 : 선호도가 높은 순으로 청소해야 할 테이블의 우선순위 출력

- 테스트 결과 스크린샷

```
---|---|---
X  |X  |X
---|---|---
_  |_  |_
---|---|---
_  |_  |_
---|---|---

0. 프로그램 종료
1. 입점 테이블 입력
2. 퇴점 테이블 입력
3. 청소한 테이블 입력
4. 테이블 청소 우선순위
원하는 번호를 입력하세요:4
청소 우선순위 : 테이블1 (선호도9)->테이블2 (선호도8)->테이블3 (선호도7
)
```

(5) 프로그램 종료

- 설명 : 0번 입력하면 프로그램 종료

- 테스트 결과 스크린샷

```
원하는 번호를 입력하세요:0
프로그램을 종료합니다
PS C:\CPP2409-P>
```

4. 계획 대비 변경 사항

1) 기능 1

- 이전 : 테이블 번호, 손님 선호도, 점유 여부 저장하는 테이블 클래스

- 이후 : 테이블 청소 여부를 확인하는 clean 멤버변수를 추가했다. 점유, 청소, 퇴점 여부를 반환하는 getStatus() 멤버함수도 추가했다. 입/퇴점 함수에도 청소 여부에 따른 출력을 수정했다.

- 사유 : 테이블의 청소여부를 구분하고 3X3 배열에 테이블 상태를 한눈에 보이게 출력하고 싶었다.

2) 기능 2

- 이전 : 입점, 퇴점, 청소순위 중 알고 싶은 것을 터미널에서 입력받음

- 이후 : 테이블 모양과 상태를 함께 출력하고 청소한 테이블도 입력받음

- 사유 : 테이블 상태를 한눈에 보이게 출력하고 청소 여부도 입력을 통해 변경하고 싶었다.

5. 프로젝트 일정

(진행한 작업과 진행 중인 작업 등을 표기)

업무	11/3	11/17	12/1	12/15	12/22
제안서 작성	완료				
기능1		완료			
기능2			완료		
기능3				완료	
기능 보완					------→