수업명

수요의 카페

프로젝트 제안서

제출일자: 12/22
제출자명: 황윤규
제출자학번: 214968

1. 프로젝트 목표

1) 배경 및 필요성 (14 pt)

고등학생 시절 게임이론을 게임으로 만들어 보여주는 게임을 해본 적이 있습니다. 특정 학문의 이론을 게임으로 접하게 되면서 재미도 있고 글로만 접하는 것이 아니라 더 이해하기 쉬웠던 경험이 있습니다. 이에 착안하여 제가 1학년 시절 경제학개론에서 어려움을 겪었던 기억이 있습니다. 그중에서 가장 핵심이었다고 생각이 드는 수요 공급 곡선과 관련된 내용 재미있게 만들면 좋겠다는 생각이 들어 게임으로 만들고자 합니다. 컨셉으로는 카페를 경영하는 게임으로 수요 공급 곡선에서 가격과 공급자가 원하는 가격인 균형지점이라는 곳이 있습니다. 이때 플레이어는 카페 메뉴를 균형지점에 해당하는 가격을 찾아내어 돈을 가장 많이 버는 것을 목표로 하는 게임입니다. 거기에 다른 추가적인 베블런 효과나 외부효과와 같은 것들을 추가하여 색다른 재미와 동시에 더 추가적인 효과를 기대할 수 있습니다.

2) 프로젝트 목표

수용 공급 곡선의 핵심 개념인 가격이 오르면 소비자의 수요가 줄어들고 가격을 내리면 수요가 증가한다는 개념을 게임을 통해 전하는 것을 목표로 합니다. 그리고 후에 엔딩에서 해당 개념을 설명해줌으로써 더 쉽고 재미있게 전하고자 합니다. 게임의 몰입도는 해치지 않되 게임을 진행함으로써 재미있고 유익한 시간이 되도록 하고 싶습니다.

3) 차별점

기존에 학습을 목표로 한 게임들은 대다수 그렇게 인지도를 가지지 못하였다고 알고 있습니다. 제가 느끼기에는 이런 게임들이 큰 반향을 일으키지 못한 까닭은 너무 학습에만 목표를 잡고 교육의 개념을 게임 내에 지속적으로 노출시켜 게임의 몰입감이 떨어졌기 때문이라고 생각합니다. 따라서 재미를 줄 수 있는 게임에 일종에 요소로 개념을 집어넣고 이를 핵심으로 두면 자연스럽게 해당 개념을 접하게되고 원리를 이해시킨 후에 엔딩에 이와 관련된 내용을 알려줌으로써 게임은 게임대로 몰입감을 더하고 마지막에 가르침도 얻을 수 있도록 하는 점이 차별점이라고 생각합니다.

2. 기능 계획

1) 기능 1 (플레이어 행동)

- 설명 플레이어의 행동 요소로 선택하여 행동하는 기능입니다.

(1) 세부 기능

- 플레이어의 행동은 낮과 밤으로 나뉘어지고 행동 요소가 달라집니다.

낮 : 1. 카페 메뉴 만들기 ,2. 디저트 만들기 2. 카페 메뉴 확인, 3. 정산 , 4. 미니게임, 5, 판매 시작

밤 : 1. 메뉴 가격 수정 , 2. 직원 고용 , 3. 카페 메뉴 확인, 4. 정산, 5. 신문 확인하기, 6. 잠자기

==========(낮)==========

1) 기능 2 (카페 메뉴 만들기)

- 설명 : 카페의 메뉴의 이름과 가격을 입력받아 메뉴를 제작합니다.

(1) 세부 기능

- 메뉴의 가격과 이름을 입력받게되면 수요 공급 곡선에 따라서 가격에 맞추에 판매량이 결정 됩니다.
- 후에 플레이어 행동 양식(낮) 5번 판매 시작 시에 정해진 판매량에 따라 판매됩니다.

1) 기능 3 (디저트 만들기)

- 설명 : 디저트의 이름을 입력받아 메뉴를 제작합니다.

(1) 세부 기능

- 디저트는 커피에 절반만큼 팔립니다.

- 하루에 한번 나오는 신문에서 나온 특정 단어를 조합하여 만들면 더 많이 팔립니다.

- 디저트는 한 종류만 제작이 가능합니다.

1) 기능 4 (카페 메뉴 확인)

- 설명 : 현재 존재하는 카페 메뉴를 모두 보여줍니다.

(1) 세부 기능

- 메뉴의 이름과 가격을 보여줍니다.

- 만일 메뉴가 이미 한번 판매를 한 전적이 있다면 판매량또한 보여줍니다.

- 이는 후에 플레이어 행동 양식(밤)에 1번 메뉴 가격 수정시에 용이하게 사용됩니다.

1) 기능 5 (정산)

- 설명 : 지금까지 판매한 메뉴들의 총 판매량을 모두 합산하여 보여줍니다.

(1) 세부 기능

- 현재까지 판매한 판매량을 종합하여 확인할 수 있도록 합니다.

1) 기능 6 (미니게임)

- 설명 : 미니게임을 실행합니다.

(1) 세부 기능

-> 미니게임을 통해서 그날 신문에 따라 정해지는 키워드를 맞추는 미니게임을 진행합니다,

-> 베블런 효과에 따른 미니게임을 진행합니다.

1) 기능 7 (판매 시작)

- 설명 : 현재 게시된 메뉴들로 판매를 시작합니다.

(1) 세부 기능

- 현재 게시된 메뉴들로 판매를 시작하되 판매량은 가격에 따라 정해진 양만 판매되어집니다.

- 판매량은 후에 플레이어 행동 양식(낮/밤) 카페 메뉴 확인을 통해 확인이 가능합니다.

==========(밤)==========

1) 기능 8 (메뉴 가격 수정)

- 설명 : 메뉴의 가격을 수정합니다.

(1) 세부 기능

- 메뉴의 가격을 다시 수정할 수 있습니다. 커피는 가격에 따른 판매량도 재조정됩니다.
- 메뉴의 수정은 메뉴별로 한 번만 가능합니다.

1) 기능 9 (직원 고용)

- 설명 : 돈을 일정 소모하여 직원을 고용합니다.

(1) 세부 기능

- 직원 고용 시에 카페 메뉴를 추가로 더 제작이 가능해집니다.
- 직원 고용 시 3명의 직원을 보여주며 고용 비용은 무작위로 나옵니다.

1) 기능 10 (신문 확인하기)

- 설명 : 하루에 한번 신문이 나오게 됩니다.

(1) 세부 기능

- 신문에는 그날 유행하는 단어가 나오게 됩니다.
- 랜덤하게 하나의 신문을 제공하여 줍니다.
- "오늘은 사과가 맛있습니다."와 같은 신문이 나오면 디저트에 사과타르트와 같은 메뉴를 제작하면 더 많이 팔리게 됩니다.

1) 기능 11 (낮과 밤)

- 설명 : 플레이어는 낮과 밤에 따라 행동양식이 달라집니다.

(1) 세부 기능

- 플레이어의 행동은 낮과 밤으로 나뉘어지고 행동 요소가 달라집니다.
- 낮에는 플레이어 행동 양식(낮) 5번 판매 시작을 통해 판매가 끝나면 밤으로 이동합니다.
- 밤에는 플레이어 행동 양식(밤) 5번 잠자기를 통해 낮으로 이동합니다.

1) 기능 12 (시간)

- 설명 : 플레이어가 행동할 수 있는 시간은 7일으로 제한되어 있습니다.

(1) 세부 기능

- 낮과 밤이 1일이며 밤이 지나면 1일이 지나가게 됩니다.
- 7일이 모두 지나게 되면 엔딩이 나오게 됩니다. 수요 공급 곡선에 관련된 내용을 소개해 줍니다.
- 제작자가 정해 놓은 가격을 달성시에 최종 엔딩이 나오게 됩니다.

1) 기능 13 (고객 피드백 시스템)

- 설명 : 판매 이후에 고객들이 메뉴에 대해 피드백을 남깁니다.

(1) 세부 기능

- 긍정 혹은 부정 피드백을 남깁니다.
- 긍정 피드백의 경우에는 균형 가격에 가까울수록 많아집니다. 반대로 멀수록 부정 피드백이 증가합니다.

1) 기능 14 (균형 가격 설정)

- 설명 : 새로운 메뉴를 만들수록 더 비싼 메뉴를 제작하여 균형 가격이 각기 다릅니다.

(1) 세부 기능

- 균형 가격의 경우에는 랜덤으로 제공되어집니다.

1) 기능 15 (베블런 효과)

- 설명 : 특별한 효과로 특정 메뉴를 가격을 높게 설정했을 때 판매량이 증가합니다.

(1) 세부 기능

- 미니게임에서 특정 이름의 메뉴의 힌트를 제공하며 해당 이름의 메뉴의 경우에는 가격을 높게 설정할수록 판매량이 증가합니다.

1) 기능 16 (외부 효과)

- 설명 : 하루에 한 번 신문이 나오며 신문을 통해 단어를 유추하거나 미니게임에서 그 단어 맞추기 게임이 진행됩니다.

(1) 세부 기능

- 신문에는 사과가 제철과 같은 단어가 나오면 사과가 들어간 디저트류는 더 많이

팔립니다.

3. 진척 사항

1) 기능 구현

(1) 기능 1 플레이어 행동

설명

- 설명 플레이어의 행동 요소로 선택하여 행동하는 기능입니다.

(1) 세부 기능

- 플레이어의 행동은 낮과 밤으로 나뉘어지고 행동 요소가 달라집니다.

낮 : 1. 카페 메뉴 만들기 ,2. 디저트 만들기 2. 카페 메뉴 확인, 3. 정산 , 4. 미니게임, 5, 판매 시작

밤 : 1. 메뉴 가격 수정 , 2. 직원 고용 , 3. 카페 메뉴 확인, 4. 정산, 5. 신문 확인하기, 6. 잠자기

적용된 배운내용

1) 클래스

클래스 내의 맴버에 대한 접근 지정자를 통하여 접근이 가능하게 합니다.

2) switch-case문

switch-case문을 이용하여 플레이어가 행동양식을 선택했을 때에 들어가는 선택지를 코드 내에서 알아보기 쉽도록 하였습니다.

3) if문

낮과 밤을 구분하기 위해서 time변수를 활용하였습니다.

4) has a

다른 클래스를 상속하여 해당 클래스의 함수를 사용합니다.

코드 스크린샷

```
// 전체적인 행동 요소를 나타내는 함수 입니다.
void Run::run(){
    srand(static_cast<unsigned int>(time(nullptr)));
    int r = rand() % 10; // 인덱스는 0~9 범위
    // keyword를 받아옴

    string key_word = event.ReturnKeyword(r);
    // 디저트의 가격을 초기화 시킵니다.
    menu.ResetDessertPrice();

    int menu_select;

    int time = 0;

    int number;

    while(day < 7){
        menu_select = 0;
        if(time == 0){
            cout << "낮" << endl;
        }
        else{
            cout << "밤" << endl;
        }

        cout << day << "일차" << endl;

        if(time == 0){
            // 낮에 할 행동을 보여주고 입력을 통해 확인합니다.
            while(menu_select < 5){
                cout << "==================행동(낮)====================" << endl;
                cout << "1. 카페 메뉴 만들기" << endl;
                cout << "2. 디저트 만들기" << endl;
                cout << "3. 카페 메뉴 확인" << endl;
                cout << "4. 신문 확인" << endl;
                cout << "5. 미니 게임" << endl;
                cout << "6. 판매 시작" << endl;
                cout << ">> ";
                cin >> menu_select;
```

```
switch(menu_select){
    case 1:
        // 직원 고용에 따라 만들 수 있는 커피 개수를 제한합니다.
        if(make_coffee_count > menu.ReturnCount()){
            // 참조한 Menu클래스의 함수 make_menu함수를 불러옵니다.
            menu.MakeMenu();
        }
        else{
            cout << "만들 수 있는 커피의 개수를 넘었습니다." << endl;
        }
        break;
    case 2:
        // 참조한 Menu클래스의 함수 make_dessert함수를 불러옵니다.
        menu.MakeDessert(key_word);
        break;
    case 3:
        // 참조한 Menu클래스의 show_menu 함수를 불러옵니다.
        menu.ShowMenu();
        break;
    case 4:
        // 신문을 보여줍니다.
        event.ShowNewspaper(r);
        break;
    case 5:
        number = 0;
        //미니게임을 실행합니다.
        try{
            cout << "====================| 미니게임 |====================" << endl;
            cout << "1. 특수 미니게임" << endl;
            cout << "2. 신문 미니게임" << endl;
            cout << ">>";
            cin.exceptions(ios::failbit | ios::badbit);
            cin >> number;

            if(number == 1){
                event.MiniGame(100);
            }
            else if(number == 2){
                event.MiniGame(r);
            }
            else{
                throw number;
            }
        }
        catch(const ios_base::failure& e){
            cout << "다시 입력해주세요." << endl;
            // 입력 스트림을 초기화
            cin.clear(); // 상태 플래그를 초기화
            cin.ignore(numeric_limits<streamsize>::max(), '\n'); // 입력 버퍼 비우기
        }
```

```
                    break;
                case 6:
                    // 시간을 밤으로 만듭니다.
                    //anim.anim_text("판매중");
                    if(menu.ReturnMenuCheck()){
                        menu.SellMenu();
                        time = 1;
                    }
                    else{
                        cout << "아직 메뉴가 하나도 없습니다." << endl;
                    }
                    break;
                default:
                    cout << "잘못된 입력입니다." << endl;
                    break;
            }
        }
    }
    else{
        //밤에 진행할 활동을 선택합니다.
        //밤이 왔다면 다시 고용할 수 있도록 함수를 조정합니다.
        check_employee = true;

        // 밤이 오면 직원들의 일급을 하루마다 더해줍니다.
        for(int i = 0; i < staff_count; i++){
            total_staff += employee[i].employee_price;
        }

        while(menu_select < 6){
            cout << "==================행동(밤)====================" << endl;
            cout << "1. 메뉴 가격 수정" << endl;
            cout << "2. 직원 고용" << endl;
            cout << "3. 카페 메뉴 확인" << endl;
            cout << "4. 정산" << endl;
            cout << "5. 미니 게임" << endl;
            cout << "6. 잠자기" << endl;
            cout << ">> ";
            cin >> menu_select;

            switch(menu_select){
                case 1:
                    menu.ModifyMenu();
                    break;
                case 2:
                    // 직원 고용이 한번 이루어 졌다면 그날은 접근하지 못하도록 합니다.
                    // 직원 수가 3명이 넘어가면 더이상 고용이 불가능합니다.
                    if(check_employee && staff_count < 3){
                        Hire(employee[staff_count]);
                    }
```

```
                    // 직원 고용이 한번 이루어 졌다면 그날은 접근하지 못하도록 합니다.
                    // 직원 수가 3명이 넘어가면 더이상 고용이 불가능합니다.
                    if(check_employee && staff_count < 3){
                        Hire(employee[staff_count]);
                    }
                    // 직원 고용을 진행하면 고용 여부와 상관없이 그날 고용은 끝이 납니다.
                    check_employee = false;

                    break;
                case 3:
                    menu.ShowMenu();
                    break;
                case 4:
                    menu.Total(total_staff);
                    break;
                case 5:
                    event.MiniGame(r);
                    break;
                case 6:
                    // 시간을 낮으로 만듭니다.
                    //anim.anim_text("자는중");
                    time = 0;
                    day++;
                    break;
                default:
                    cout << "잘못된 입력입니다." << endl;
                    break;
            }
        }
    }
    cout << endl;
}

ending();
}
```

(2) 기능 2 카페 메뉴 만들기

설명

- 설명 : 카페의 메뉴의 이름과 가격을 입력받아 메뉴를 제작합니다.

(1) 세부 기능

- 메뉴의 가격과 이름을 입력받게되면 수요 공급 곡선에 따라서 가격에 맞추에 판매량이 결정 됩니다.

- 후에 플레이어 행동 양식(낮) 5번 판매 시작 시에 정해진 판매량에 따라 판매됩니다.

적용된 배운내용

1) while문

while문을 사용하여 메뉴의 이름이 겹치게 된다면 while문의 반복조건을 체크하여 반복문의 처음으로 돌아가도록 하였습니다.

2) for문

for문을 활용하여 지금까지 존재한 모든 메뉴들과 입력받은 메뉴의 이름을 비교하도록 만들었습니다.

3) if문

if문을 활용하여 만일 메뉴의 이름을 체크합니다.

4) vector

menu를 입력받아 중복되지 않은 메뉴라면 vector에 저장합니다.

코드 스크린샷

```
// 카페 메뉴와 관련된 클래스입니다.
class Menu{
    // 커피 메뉴에 관련된 내용입니다.
    vector<string> menu_name;
    vector<int> menu_price;
    vector<int> menu_sell;
    // 수요 공급 곡선을 어떤걸 사용할 지 저장해 놓습니다.
    vector<int> supply_demand_num;

    // 메뉴가 만들어졌는지를 체크합니다.
    bool check_menu_make = false;

    // 디저트에 관련된 내용을 저장합니다.
    string dessert_name;
    int dessert_price;
    int dessert_sell;
    int dessert_total_sell;

    // 메뉴 확인시에 디저트가 만들어지지 않았다면 보여주지 않습니다.
    bool check_dessert_make = false;

    // 정산시에 사용되는 디저트와 커피 판매액을 모두 저장해 놓습니다.
    int total_sell;

    // 베블런 효과에서 사용되는 변수입니다.
    string veblen;
public:
    Menu(){
        total_sell = 0;
        dessert_total_sell = 0;
        veblen = "veblen";
    }
    Employee employ;

    // 메뉴를 만드는데 관여합니다.
    void MakeMenu();

    // 디저트를 만드는데 관여하고 key값을 받아 만일 그날 key단어가 포함되어있는지 체크합니다.
    void MakeDessert(string key);

    // 메뉴를 보여줍니다. 여기서 check_dessert_make, check_menu_make를 이용하여
    // 메뉴가 만들어졌는지 체크합니다.
    void ShowMenu();

    // 메뉴를 수정합니다.
    void ModifyMenu();

    // 날짜를 보낼때 사용되며 총 판매량을 저장합니다.
    void SellMenu();

    // 총 만들어져 있는 메뉴의 개수를 반환합니다.
    int ReturnCount();
```

```
    // 디저트를 만드는데 관여하고 key값을 받아 만일 그날 key단어가 포함되어있는지 체크합니다.
    void MakeDessert(string key);

    // 메뉴를 보여줍니다. 여기서 check_dessert_make, check_menu_make를 이용하여
    // 메뉴가 만들어졌는지 체크합니다.
    void ShowMenu();

    // 메뉴를 수정합니다.
    void ModifyMenu();

    // 날짜를 보낼때 사용되며 총 판매량을 저장합니다.
    void SellMenu();

    // 총 만들어져 있는 메뉴의 개수를 반환합니다.
    int ReturnCount();

    // 메뉴가 만들어졌는지 여부를 반환합니다.
    bool ReturnMenuCheck();

    //수요 공급 곡선을 선택하는 함수입니다.
    void SupplyDemand(int i, bool& check);

    // 정산에 사용됩니다.
    void Total(int total_staff);

    // 디저트에 사용되며 그날의 key단어가 포함되어져 있는지 여부를 파악합니다.
    bool containsWord(const string& targetWord, const string& Word);

    // 날짜가 지나가면 디저트의 가격을 다시 5000원으로 조정합니다.
    void ResetDessertPrice();

};
```

```cpp
void Menu::MakeMenu(){
    // 이름과 가격을 입력받을 변수입니다.
    string name;
    int price;

    // 해당 변수를 통해 메뉴가 정상적으로 입력되었는지 확인합니다.
    bool menu_check = true;

    // 제작할 메뉴를 입력받습니다.
    while(menu_check){
        menu_check = false;

        cout << "======================| 주의 사항 |=========================" << endl;
        cout << "메뉴의의 이름은 영어로 작성하여 주시길 바랍니다." << endl;
        cout << "왜 한글이 안되는 것일까요...." << endl;
        cout << "============================================================" << endl;
        cout << endl;

        cout << "메뉴명을 입력해주세요 : " << endl;
        cin.ignore(); // 이전 입력이 있다면 버퍼를 비웁니다.
        // 메뉴의 이름을 입력받습니다.
        getline(cin, name);

        //메뉴의 가격을 입력받습니다.
        cout << "메뉴 가격을 입력해주세요 : " << endl;
        cin >> price;

        // 지금까지 만들어진 메뉴의 크기만큼 for문을 돕니다.
        for(int i = 0; i < menu_name.size(); i++){
            // 이미 존재하는 메뉴라면 menu_check를 true로 두어 다시 실행합니다.
            if(menu_name[i] == name){
                cout << "이미 존재하는 메뉴입니다." << endl;
                menu_check = true;
            }
        }

        // 메뉴의 중복여부를 판단합니다.
        if(!menu_check){
            // 랜덤으로 수요공급곡선 번호를 지정하여 줍니다.
            int num = rand()%3 + 1;
            // 정해진 수요 공급 곡선의 번호를 저장합니다.
            supply_demand_num.push_back(num);

            // 메뉴의 이름과 가격을 집어넣어 줍니다.
            menu_name.push_back(name);
            menu_price.push_back(price);

            // 판매 시작 전에는 판매량이 없다는 것을 알리기위해 -1로 지정합니다.
            menu_sell.push_back(-1);

            // 메뉴가 만들어졌음을 나타냅니다.
            check_menu_make= true;
        }
    }
}
```

```
// 카페 메뉴와 관련된 클래스입니다.
class Menu{
    // 커피 메뉴에 관련된 내용입니다.
    vector<string> menu_name;
    vector<int> menu_price;
    vector<int> menu_sell;
    // 수요 공급 곡선을 어떤걸 사용할 지 저장해 놓습니다.
    vector<int> supply_demand_num;

    // 메뉴가 만들어졌는지를 체크합니다.
    bool check_menu_make = false;

    // 디저트에 관련된 내용을 저장합니다.
    string dessert_name;
    int dessert_price;
    int dessert_sell;
    int dessert_total_sell;

    // 메뉴 확인시에 디저트가 만들어지지 않았다면 보여주지 않습니다.
    bool check_dessert_make = false;

    // 정산시에 사용되는 디저트와 커피 판매액을 모두 저장해 놓습니다.
    int total_sell;

    // 베블런 효과에서 사용되는 변수입니다.
    string veblen;
public:
    Menu(){
        total_sell = 0;
        dessert_total_sell = 0;
        veblen = "veblen";
    }
    Employee employ;

    // 메뉴를 만드는데 관여합니다.
    void MakeMenu();

    // 디저트를 만드는데 관여하고 key값을 받아 만일 그날 key단어가 포함되어있는지 체크합니다.
    void MakeDessert(string key);

    // 메뉴를 보여줍니다. 여기서 check_dessert_make, check_menu_make를 이용하여
    // 메뉴가 만들어졌는지 체크합니다.
    void ShowMenu();

    // 메뉴를 수정합니다.
    void ModifyMenu();

    // 날짜를 보낼때 사용되며 총 판매량을 저장합니다.
    void SellMenu();

    // 총 만들어져 있는 메뉴의 개수를 반환합니다.
    int ReturnCount();
```

(3) 기능 3 디저트 만들기

설명

- 설명 : 디저트의 이름을 입력받아 메뉴를 제작합니다.

(1) 세부 기능

- 디저트는 커피에 절반만큼 팔립니다.

- 하루에 한번 나오는 신문에서 나온 특정 단어를 조합하여 만들면 더 많이 팔립니다.

- 디저트는 한 종류만 제작이 가능합니다.

적용된 배운내용

1) 클래스

디저트도 메뉴에 포함되어 클래스에 속하여 있습니다.

2) 함수

함수를 통해 키워드가 포함되어있는지를 리턴하여 확인합니다.

3)if문

if문을 통해 가격을 결정합니다.

코드 스크린샷

```
// 디저트를 만듭니다.
void Menu::MakeDessert(string key){
    int price;
    string name;

    cout << "======================| 주의 사항 |=========================" << endl;
    cout << "디저트의 이름은 영어로 작성하여 주시길 바랍니다." << endl;
    cout << "왜 한글이 안되는 것일까요...." << endl;
    cout << "============================================================" << endl;
    cout << endl;

    // 제작할 메뉴를 입력받습니다.
    cout << "디저트를 입력해주세요 : " << endl;
    cin.ignore(); // 이전 입력이 있다면 버퍼를 비웁니다.
    getline(cin, name);
    dessert_name = name;

    cout << "기본 디저트 가격은 5000원입니다. " << endl;

    // 그날의 키워드가 포함되어져 있는 지 체크합니다.
    if(containsWord(key, dessert_name)){
        cout << "디저트가 더 비싼 가격에 팔립니다!!" << endl;
        dessert_price = 7000;
    }
    else{
        dessert_price = 5000;
    }

    // 디저트 확인 시에 디저트의 여부를 판별
    check_dessert_make = true;
}
```

(4) 기능 4 카페 메뉴 확인

설명

- 설명 : 현재 존재하는 카페 메뉴를 모두 보여줍니다.

(1) 세부 기능

- 메뉴의 이름과 가격을 보여줍니다.
- 만일 메뉴가 이미 한번 판매를 한 전적이 있다면 판매량또한 보여줍니다.
- 이는

적용된 배운내용

1) if문

if문을 통해 메뉴의 제작여부를 판별합니다.

2) for문

for문을 통해 현재 vector내에 메뉴들을 모두 출력합니다.

코드 스크린샷

```cpp
// 메뉴를 보여줍니다.
void Menu::ShowMenu(){
    // 메뉴가 만들어졌는지 체크하여 그 여부에 따라 메뉴를 보여줍니다.
    if(check_menu_make){
        cout << "==============(coffee)================" << endl;

        // 메뉴의 크기만큼 for문을 작동시킵니다.
        for(int i = 0; i < menu_name.size(); i++){
            cout << (i+1) << "번 메뉴" << endl;
            cout << menu_name[i] << endl;
            cout << menu_price[i] << endl;

            //만약 아직 판매가 진행되어진 메뉴는 판매량을 보여줍니다.
            // 판매가 진행되지 않은 메뉴는 판매량이 -1임으로 이를 통해 판단합니다.
            if(menu_sell[i] != -1){
                cout << "판매량" << endl;
                cout << menu_sell[i] << endl;
            }
            cout << endl;
        }
    }
    else{
        cout << "아직 만들어진 메뉴가 없습니다." << endl;
    }

    // 디저트 제작여부에 따라서 메뉴를 보여줍니다.
    if(check_dessert_make){
        cout << "==============(dessert)================" << endl;
        cout << "오늘의 디저트" << endl;
        cout << dessert_name << endl;
        cout << dessert_price << endl;

        // 디저트도 판매 여부를 판단하여 판매가 되었다면 판매량을 보여줍니다.
        if(dessert_sell != 0){
            cout << "판매량" << endl;
            cout << dessert_sell << endl;
        }
        cout << endl;
    }
    else{
        cout << "아직 만들어진 디저트가 없습니다." << endl;
    }

}
```

(5) 기능 5 정산

설명

(1) **설명**
- 설명 : 지금까지 판매한 메뉴들의 총 판매량을 모두 합산하여 보여줍니다.

(1) 세부 기능

- 현재까지 판매한 판매량을 종합하여 확인할 수 있도록 합니다.

적용된 배운내용

1) 입출력 함수

입력과 출력을 합니다.

2) 클래스

클래스 내의 맴버에 대한 접근 지정자를 통하여 접근이 가능하게 합니다.

코드 스크린샷

```
//정산하여 총 가격을 보여줍니다.
void Menu::Total(int total_staff){
    cout << "====================| 정산 |======================" << endl;
    // 커피의 총 판매량을 모두 계산하여 보여줍니다.
    cout << "===| 커피 |===" << endl;
    cout << total_sell << endl;
    cout << "\n";

    // 디저트의 총 판매량을 모두 계산하여 보여줍니다.
    cout << "===| 디저트 |===" << endl;
    cout << dessert_total_sell << endl;
    cout << "\n";

    // 직원 월급의 총 판매량을 모두 계산하여 보여줍니다.
    cout << "===| 직원 월급 |===" << endl;
    cout << total_staff << endl;
    cout << "\n";

    // 커피 + 디저트 - 직원월급을 하여 보여줍니다.
    cout << "===| 정산 |===" << endl;
    cout << total_sell + dessert_total_sell - total_staff << endl;
}
```

(6) 기능 6 미니 게임

설명

-> 미니게임을 통해서 그날 신문에 따라 정해지는 키워드를 맞추는 미니게임을 진행합니다.

-> 베블런 효과에 따른 미니게임을 고려 중입니다.

적용된 배운내용

1) while 문

정답이거나 정해진 횟수 전까지 반복합니다.

2) 클래스

클래스 내의 맴버에 대한 접근 지정자를 통하여 접근이 가능하게 합니다.

3) map

각 신문의 이름을 key로 두어 값(value)를 나타냅니다.

index를 통해 값을 찾아냅니다.

4) try-catch 문

잘못된 값이 입력됐을 시에 무한 루프가 돌지 않도록 try-catch문을 통해 방지합니다.

```
// 미니게임을 실행합니다.
void Event::MiniGame(int index) {
    // map에서 index에 해당하는 키 찾기
    auto it = newspaperKeywords.begin();
    advance(it, index);

    string keyword = it->second;  // 해당 키워드
    string guessed(keyword.size(), '_');

    // 사용할 수 있는 횟수이며 10번으로 제한합니다.
    int attempts_left = 10;

    // 추측한 단어를 저장합니다.
    string guessed_letters = "";

    // 맞추거나 횟수를 모두 소진할 때까지 반복하여 실행합니다.
    while (attempts_left > 0 && guessed != keyword) {
        // 현재 상태 출력
        cout << "\n단어: ";
        for (char c : guessed) {
            cout << c << ' ';
        }
        cout << "\n남은 시도 횟수: " << attempts_left << "\n";
        cout << "추측한 글자: " << guessed_letters << "\n";
        cout << "알파벳 한 글자를 입력하세요: ";

        // 추측할 단어를 입력받습니다.
        char guess;
        cin >> guess;
        guess = tolower(guess);

        // 잘못된 입력이거나 이미 추측한 단어일 경우에 다시 진행합니다.
        if (!isalpha(guess) || guessed_letters.find(guess) != std::string::npos) {
            cout << "잘못된 입력입니다. 한 글자 알파벳만 입력하거나, 이미 추측한 글자를 제외해주세요.\n";
            continue;
        }
```

```
        // 추측한 단어는 담아줍니다.
        guessed_letters += guess;

        // 추측한 알파벳이 포함되어져 있는 곳을 찾습니다.
        bool found = false;
        for (size_t i = 0; i < keyword.size(); ++i) {
            if (keyword[i] == guess) {
                guessed[i] = guess;
                found = true;
            }
        }

        // 포함 여부를 보여줍니다.
        if (found) {
            cout << "좋아요! '" << guess << "'는 단어에 포함되어 있습니다.\n";
        } else {
            cout << "안타깝습니다. '" << guess << "'는 단어에 없습니다.\n";
        }

        // 횟수를 소진합니다.
        attempts_left--;
    }

    // 성공 여부를 보여줍니다.
    if (guessed == keyword) {
        cout << "\n축하합니다! 단어를 맞추셨습니다: " << keyword << "\n";
    } else {
        cout << "\n게임 오버! 다시 도전해주세요!!: " << "\n";
    }
}
```

```
case 5:
    number = 0;
    //미니게임을 실행합니다.
    try{
        cout << "====================| 미니게임 |====================" << endl;
        cout << "1. 특수 미니게임" << endl;
        cout << "2. 신문 미니게임" << endl;
        cout << ">>";
        cin.exceptions(ios::failbit | ios::badbit);
        cin >> number;

        if(number == 1){
            event.MiniGame(100);
        }
        else if(number == 2){
            event.MiniGame(r);
        }
        else{
            throw number;
        }
    }
    catch(const ios_base::failure& e){
        cout << "다시 입력해주세요." << endl;
        // 입력 스트림을 초기화
        cin.clear(); // 상태 플래그를 초기화
        cin.ignore(numeric_limits<streamsize>::max(), '\n'); // 입력 버퍼 비우기
    }

    break;
```

(7) 기능 7 (판매 시작)

- 설명 : 현재 게시된 메뉴들로 판매를 시작합니다.

(1) 세부 기능

- 현재 게시된 메뉴들로 판매를 시작하되 판매량은 가격에 따라 정해진 양만 판매되어집니다.
- 판매량은 푸에 플레이어 행동 양식 (낮/밤) 카페 메뉴 확인을 통해 확인이 가능합니다.
- 메뉴 제작시에 랜덤으로 수요 공급 곡선을 정합니다,

적용된 배운내용

1) for 문

for문을 활용하여 만들어진 메뉴들의 수요 공급 곡선을 정하고 이를 판매량으로 결정합니다.

2) 함수

수요 공급 곡선을 정해주는 함수를 호출하여 수요 공급곡선에 맞추어 판매량을 경정하여 줍니다.

3) min 함수

min 함수를 활용하여 수요자의 수요와 공급자의 공급중에 적은 값을 골라 가격이

너무 싸면 공급이 줄어들고, 비싸면 수요가 줄어 판매가 줄어듭니다.

4) 난수 생성

1~3 사이의 난수로 3가지 수요공급곡선중에 하나를 선택합니다.

코드 스크린샷

```
void Menu::SellMenu(){
    // 디저트의 판매량은 커피의 절반만큼 팔립니다.
    // 다만 커피의 판매량은 가격에 따라 바뀔 수 있음으로 0으로 초기화하고 다시 받아줍니다.
    dessert_sell = 0;

    // 메뉴를 전부 돌면서 정해진
    for(int i = 0; i < menu_price.size(); i++){
        SupplyDemand(i);
    }

    // 커피의 모든 판매량의 절반만큼 판매되어 저장합니다.
    dessert_total_sell += 5000 * dessert_sell;
}
```

1) 기능 8 (메뉴 가격 수정)

- 설명 : 메뉴의 가격을 수정합니다.

(1) 세부 기능

- 메뉴의 가격을 다시 수정할 수 있습니다. 커피는 가격에 따른 판매량도 재조정됩니다.

- 메뉴의 수정은 메뉴별로 한 번만 가능합니다.

적용된 배운내용

1) string

수정할 이름을 받습니다.

2) while문

수정사항을 bool 변수인 menu_check를 통해 수정이 올바르게 종료될 경우에 반복을 종료합니다.

3) for문

입력받은 이름이 메뉴에 존재한다면 가격을 입력받고 그 가격으로 수정하여 줍니다.

4) if문

입력받은 이름이 메뉴에 존재한다면 수정을 진행합니다.

```cpp
// 균형지점을 지정해주는 함수 입니다.
void Menu::SupplyDemand(int i, bool& check){
    int price, demand, supply;

    // 3개의 균형지점 중에 랜덤으로 결정하여 줍니다.
    if(supply_demand_num[i] == 1){
        price = menu_price[i] / 100;
        demand = 100 - price;
        supply = 3 * price;
    }
    else if(supply_demand_num[i] == 2){
        price = menu_price[i] / 100;
        demand = 150 - price;
        supply = 1.5 * price;
    }
    else{
        price = menu_price[i] / 100;
        demand = 80 - price;
        supply = price;
    }

    // 수요와 공급 중에서 작은 값을 판매량으로 결정합니다.
    int insert = min(demand, supply);

    // 몇개가 판매되는지 판매량을 넣습니다.
    // veblen이 포함 시에 판매량이 늘어납니다.
    // check를 통해 veblen은 한번만 적용 가능하도록 합니다.
    if(containsWord(veblen, menu_name[i]) && check){
        menu_sell[i] = insert + 30;
        check = false;
    }
    else{
        menu_sell[i] = insert;
    }

    // 디저트의 판매량은 커피의 절반만큼 팔립니다.
    if(check_dessert_make){
        dessert_sell += menu_sell[i] / 2;
    }

    // 판매된 총 가격을 입력합니다. (정산에서 이용됩니다.)
    total_sell += menu_price[i] * menu_sell[i];
}
```

1) 기능 9 (직원 고용)

- 설명 : 돈을 일정 소모하여 직원을 고용합니다.

(1) 세부 기능

- 직원 고용 시에 카페 메뉴를 추가로 더 제작이 가능해집니다.
- 직원 고용 시 3명의 직원을 보여주며 고용 비용은 무작위로 나옵니다.

적용된 배운내용

1) class

class를 사용하여 같은 객체를 여러 가지 사용하기 위해서 활용하였습니다.

2) friend

접근 지정자와 관계없이 멤버 변수를 공유하여 함수에서도 사용할 수 있도록 하였습니다.

3) 참조자 매개변수

참조자 매개변수를 활용하여 함수가 Employee 객체를 반환하게 하였습니다.

4) 난수 생성

1~3 사이의 난수로 3가지 수요공급곡선중에 하나를 선택합니다.

```
//직원 고용 토대 제작 완료
void Hire(Employee& hire){
    srand((unsigned)time(NULL));

    string yn;
    cout << "=======================| 주의 사항 |=========================" << endl;
    cout << "직원 고용을 진행시 고용 여부와 상관없이 그날 고용은 끝이납니다" << endl;
    cout << "주의하여 고용을 진행하여 주시길 바랍니다." << endl;
    cout << "============================================================" << endl;

    cout << "직원 고용" << endl;

    //직원을 돌아가면서 총 3번을 보여줍니다.
    for(int i = 0; i < 3; i++){
        // 가격과 고용비용, 이름을 랜덤으로 뽑습니다.
        int employee_money = rand()%10000 + 1000;
        int menu_count_plus = rand()%3 + 1;
        int k = rand()%9;

        // 저장된 직원의 이름을 랜덤으로 보여줍니다.
        cout << "직원이름 : "  << hire.rand_name[k] << endl;

        // 랜덤으로 고용비용을 소개하여 줍니다.
        cout << "고용비용 : " << employee_money << "원" << endl;

        // 메뉴 추가의 개수도 제공하여 줍니다. (추후 확률 조정 예정)
        cout << "메뉴 추가 개수 : " << menu_count_plus << "개" << endl;

        // 한번만 고용이 가능합니다.
        cout << "해당 직원을 고용하시겠습니까?(Y/N) : ";
        cin >> yn;
```

```
        if(yn == "Y" || yn == "y" || yn == "Yes" || yn == "yes"){
            cout << "해당직원을 고용하였습니다." << endl;
            // 고용 시에 만들 수 있는 메뉴 개수 추가
            // 최대 메뉴 개수를 제한하여 추가합니다.
            if(make_coffee_count + menu_count_plus < MAX_MENU){
                make_coffee_count += menu_count_plus;
            }
            else{
                // 메뉴의 최대개수로 제한합니다.
                cout << "최대 메뉴 개수에 도달하였습니다.(10개)" << endl;
                make_coffee_count = MAX_MENU;
            }

            // Employee class에 고용한 이름과 고용 비용 vector에 입력
            hire.employee_name = hire.rand_name[k];
            hire.employee_price = employee_money;
            staff_count++;

            // 현재 가용 메뉴 개수를 출력하여 보여줍니다.
            cout << "현재 가용 가능한 메뉴의 개수는 " << make_coffee_count << "개 입니다." << endl;
            cout << "현재 고용한 직원 수는 " << "(" << staff_count << "/3)" << "명 입니다." << endl;
            break;
        }
        else if(yn == "N" || yn == "n" || yn == "No" || yn == "no"){
            cout << "다음 직원을 소개합니다." << endl;
            cout << endl;
        }
        else{
            cout << "다시 한번 입력해주세요." << endl;
        }
    }
}
```

1) 기능 10 (신문 확인하기)

- 설명 : 하루에 한번 신문이 나오게 됩니다.

(1) 세부 기능

- 신문에는 그날 유행하는 단어가 나오게 됩니다.
- 랜덤하게 하나의 신문을 제공하여 줍니다.
- "오늘은 사과가 맛있습니다."와 같은 신문이 나오면 디저트에 사과타르트와 같은 메뉴를 제작하면 더 많이 팔리게 됩니다.

적용된 배운내용

1) 파일 입출력

index를 통해 신문과 값을 가져와서 해당 파일이 존재하면 열어 해당 파일의 내용을 보여줌

2) map

각 신문의 이름을 key로 두어 값(value)를 나타냅니다.

index를 통해 값을 찾아냅니다.

3) auto it

auto it을 통해 랜덤으로 정해진 신문의 키워드를 찾아 저장해 놓습니다.

코드 스크린샷

```cpp
// 각종 이벤트와 관련된 클래스 입니다.
class Event {
public:
    // 신문 파일과 키워드 저장합니다.
    map<string, string> newspaperKeywords = {
        {"newspaper1.txt", "nobel"},
        {"newspaper2.txt", "newjeans"},
        {"newspaper3.txt", "ott"},
        {"newspaper4.txt", "chuncheon"},
        {"newspaper5.txt", "electric"},
        {"newspaper6.txt", "apple"},
        {"newspaper7.txt", "ai"},
        {"newspaper8.txt", "martial law"},
        {"newspaper9.txt", "bitcoin"},
        {"newspaper10.txt", "subway"}
    };

    // 신문을 보여주는데 사용되어집니다. index의 경우에는 하루에 랜덤으로 선택되어집니다.
    void ShowNewspaper(int index);

    // 미니게임을 실행합니다.
    void MiniGame(int index);

    // 그날의 keyword를 반환합니다.
    string ReturnKeyword(int index);
};
```

```cpp
// 신문을 보여주는 함수입니다.
void Event::ShowNewspaper(int index) {
    // map에서 index에 해당하는 키 찾기
    auto it = newspaperKeywords.begin();
    advance(it, index);

    string news = it->first;  // 선택된 파일 이름
    string keyword = it->second;  // 해당 키워드

    // 선택되어지는 신문을 확인합니다.
    //cout << "선택된 신문 파일: " << news << endl;
    //cout << "핵심 키워드: " << keyword << endl;

    // 파일 읽기
    ifstream is{news};
    if (!is) {
        cerr << "파일 오픈에 실패하였습니다: " << news << endl;
        return;
    }

    cout << "==================| 신문 내용 |====================" << endl;
    char c;
    while (is.get(c)) {
        cout << c;
    }
    cout << endl;
}
```

1) 기능 11 (낮과 밤)

- 설명 : 플레이어는 낮과 밤에 따라 행동양식이 달라집니다.

(1) 세부 기능

- 플레이어의 행동은 낮과 밤으로 나뉘어지고 행동 요소가 달라집니다.
- 낮에는 플레이어 행동 양식(낮) 5번 판매 시작을 통해 판매가 끝나면 밤으로 이동합니다.
- 밤에는 플레이어 행동 양식(밤) 5번 잠자기를 통해 낮으로 이동합니다.

적용된 배운내용

1) case문

밤에 case6를 실행하면 하루를 더해 날짜를 표시합니다.

2) while문

시간 기한인 7일이 되기 전까지 반복합니다.

코드 스크린샷

```
// 전체적인 행동 요소를 나타내는 함수 입니다.
void Run::run(){
    srand(static_cast<unsigned int>(time(nullptr)));
    int r = rand() % 10; // 인덱스는 0~9 범위
    // keyword를 받아옴

    string key_word = event.ReturnKeyword(r);
    // 디저트의 가격을 초기화 시킵니다.
    menu.ResetDessertPrice();

    int menu_select;

    int time = 0;

    while(day < 7){
case 6:
    // 시간을 낮으로 만듭니다.
    //anim.anim_text("자는중");
    time = 0;
    day++;
    break;
```

1) 기능 12 (시간)

- 설명 : 플레이어가 행동할 수 있는 시간은 7일으로 제한되어 있습니다.

(1) 세부 기능

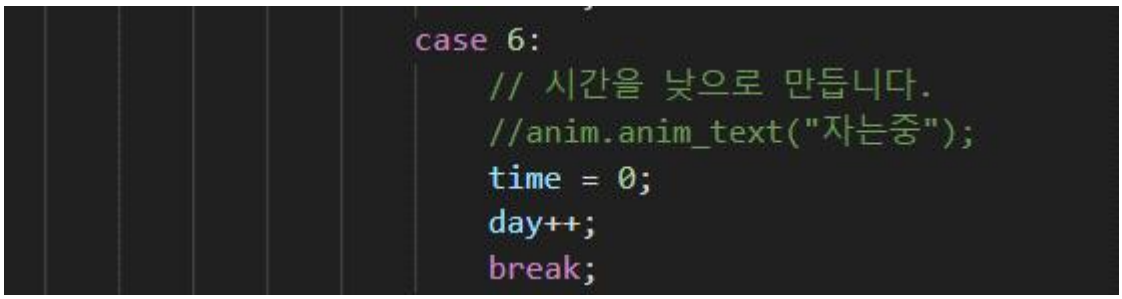
- 낮과 밤이 1일이며 밤이 지나면 1일이 지나가게 됩니다.
- 7일이 모두 지나게 되면 엔딩이 나오게 됩니다. 수요 공급 곡선에 관련된 내용을 소개해 줍니다.
- 제작자가 정해 놓은 가격을 달성시에 최종 엔딩이 나오게 됩니다.

적용된 배운내용

1) while문

시간 기한인 7일이 되기 전까지 반복합니다.

코드 스크린샷



```
case 6:
    // 시간을 낮으로 만듭니다.
    //anim.anim_text("자는중");
    time = 0;
    day++;
    break;
```

~~1) 기능 13 (고객 피드백 시스템) 삭제된 기능~~

~~- 설명 : 판매 이후에 고객들이 메뉴에 대해 피드백을 남깁니다.~~

~~(1) 세부 기능~~

~~- 긍정 혹은 부정 피드백을 남깁니다.~~

~~- 긍정 피드백의 경우에는 균형 가격에 가까울수록 많아집니다. 반대로 멀수록 부정 피드백이 증가합니다.~~

1) 기능 14 (균형 가격 설정)

- 설명 : 새로운 메뉴를 만들수록 더 비싼 메뉴를 제작하여 균형 가격이 각기 다릅니다.

(1) 세부 기능

- 균형 가격의 경우에는 랜덤으로 제공되어집니다.

적용된 배운내용

1) 함수

함수에 index값을 받아와 그에 해당하는 인덱스에 따라 균형함수가 정해집니다.

2) if문

if문을 통해 veblen효과의 적용 여부를 받습니다.

코드 스크린샷

```
void Run::run(){
    srand(static_cast<unsigned int>(time(nullptr)));
    int r = rand() % 10; // 인덱스는 0~9 범위
    // keyword를 받아옴

    string key_word = event.ReturnKeyword(r);
    // 디저트의 가격을 초기화 시킵니다.
    menu.ResetDessertPrice();

    int menu_select;

    int time = 0;

    int number;

    while(day < 7){
        menu_select = 0;
        if(time == 0){
            cout << "낮" << endl;
        }
        else{
            cout << "밤" << endl;
        }
```

1) 기능 15 (베블런 효과)

- 설명 : 특별한 효과로 특정 메뉴를 가격을 높게 설정했을 때 판매량이 증가합니다.

(1) 세부 기능

- 미니게임에서 특정 이름의 메뉴의 힌트를 제공하며 해당 이름의 메뉴의 경우에는

가격을 높게 설정할수록 판매량이 증가합니다.

적용된 배운내용

1) class

class를 사용하여 같은 객체를 여러 개 사용하기 위해서 활용하였습니다.

2) for문

for 문에서 메뉴의 숫자만큼 돌고, veblen 효과를 적용 시킬 단어를 보냅니다.

3) 참조자 매개변수

참조자 매개변수로 활용하여 함수에 veblen 효과가 한번만 적용되도록 (예를 들면 베블런 효과가 적용되는 단어가 메뉴 2개에 들어있을 경우) check_veblen함수를 보내 확인합니다.

코드 스크린샷

```
void Menu::SellMenu(){
    bool check_veblen = true;

    // 디저트의 판매량은 커피의 절반만큼 팔립니다.
    // 다만 커피의 판매량은 가격에 따라 바뀔 수 있음으로 0으로 초기화하고 다시 받아줍니다.
    dessert_sell = 0;

    // 메뉴를 전부 돌면서 정해진
    for(int i = 0; i < menu_price.size(); i++){
        SupplyDemand(i, check_veblen);
    }

    // 커피의 모든 판매량의 절반만큼 판매되어 저장합니다.
    if(check_dessert_make){
        dessert_total_sell += 5000 * dessert_sell;
    }
}
```

1) 기능 16 (외부 효과)

~~- 설명 : 하루에 한 번 신문이 나오며 그 신문에 단어에 대한 힌트가 제공됩니다.~~

- 설명 : 하루에 한 번 신문이 나오며 신문을 통해 단어를 유추하거나 미니 게임에서 그 단어 맞추기 게임이 진행됩니다.

(1) 세부 기능

- 신문에는 사과가 제철과 같은 단어가 나오면 사과가 들어간 디저트류는 더 많이 팔립니다.

적용된 배운내용

1) getline()

메뉴 이름 작성시에 띄어쓰기도 입력을 받기 위해 사용하였습니다.

2) if문

함수를 통해 해당 키워드가 포함되어져 있는지를 체크합니다.

코드 스크린샷

```
// 균형지점을 지정해주는 함수 입니다.
void Menu::SupplyDemand(int i, bool& check){
    int price, demand, supply;

    // 3개의 균형지점 중에 랜덤으로 결정하여 줍니다.
    if(supply_demand_num[i] == 1){
        price = menu_price[i] / 100;
        demand = 100 - price;
        supply = 3 * price;
    }
    else if(supply_demand_num[i] == 2){
        price = menu_price[i] / 100;
        demand = 150 - price;
        supply = 1.5 * price;
    }
    else{
        price = menu_price[i] / 100;
        demand = 80 - price;
        supply = price;
    }

    // 수요와 공급 중에서 작은 값을 판매량으로 결정합니다.
    int insert = min(demand, supply);

    // 몇개가 판매되는지 판매량을 넣습니다.
    // veblen이 포함 시에 판매량이 늘어납니다.
    // check를 통해 veblen은 한번만 적용 가능하도록 합니다.
    if(containsWord(veblen, menu_name[i]) && check){
        menu_sell[i] = insert + 30;
        check = false;
    }
    else{
        menu_sell[i] = insert;
    }

    // 디저트의 판매량은 커피의 절반만큼 팔립니다.
    if(check_dessert_make){
        dessert_sell += menu_sell[i] / 2;
    }

    // 판매된 총 가격을 입력합니다. (정산에서 이용됩니다.)
    total_sell += menu_price[i] * menu_sell[i];
}
```

```cpp
// 디저트를 만듭니다.
void Menu::MakeDessert(string key){
    int price;
    string name;

    cout << "======================| 주의 사항 |=========================" << endl;
    cout << "디저트의 이름은 영어로 작성하여 주시길 바랍니다." << endl;
    cout << "왜 한글이 안되는 것일까요...." << endl;
    cout << "============================================================" << endl;
    cout << endl;

    // 제작할 메뉴를 입력받습니다.
    cout << "디저트를 입력해주세요 : " << endl;
    cin.ignore(); // 이전 입력이 있다면 버퍼를 비웁니다.
    getline(cin, name);
    dessert_name = name;

    cout << "기본 디저트 가격은 5000원입니다. " << endl;

    // 그날의 키워드가 포함되어져 있는 지 체크합니다.
    if(containsWord(key, dessert_name)){
        cout << "디저트가 더 비싼 가격에 팔립니다!!" << endl;
        dessert_price = 7000;
    }
    else{
        dessert_price = 5000;
    }

    // 디저트 확인 시에 디저트의 여부를 판별
    check_dessert_make = true;
}
```

```cpp
// 특정 단어가 문자열에 포함되어 있는지 확인하는 함수
bool Menu::containsWord(const string& targetWord,const string& Word) {
    istringstream stream(Word);
    string word;

    while (stream >> word) {
        if (word == targetWord) {
            return true;
        }
    }
    // 만일 단어가 포함되어져 있지 않다면 false를 반환합니다.
    return false;
}
```

4. 테스트 결과

(1) 플레이어 행동

- 설명

플레이어의 행동 요소를 선택하여 행동합니다.

- 테스트 결과 스크린샷

낮

```
낮
1일차
====================행동(낮)=====================
1. 카페 메뉴 만들기
2. 디저트 만들기
3. 카페 메뉴 확인
4. 신문 확인
5. 미니 게임
6. 판매 시작
>>
```

1번 행동

```
>> 1
=======================| 주의 사항 |==========================
메뉴의의 이름은 영어로 작성하여 주시길 바랍니다.
왜 한글이 안되는 것일까요....
================================================================

메뉴명을 입력해주세요 :
coffee
메뉴 가격을 입력해주세요 :
1500
====================행동(낮)=====================
```

2번 행동

```
>> 2
=======================| 주의 사항 |==========================
디저트의 이름은 영어로 작성하여 주시길 바랍니다.
왜 한글이 안되는 것일까요....
================================================================

디저트를 입력해주세요 :
dessert
기본 디저트 가격은 5000원입니다.
```

2번 행동

3번 행동

```
>> 3
==============(coffee)================
1번 메뉴
coffee
1500

==============(dessert)================
오늘의 디저트
dessert
5000
```

4번 행동

```
 >> 4
 ==================| 신문 내용 |====================
 같은 기간 걸그룹 뉴진스를 둘러싼 내홍으로 증시에서도 화제가
 된 하이브가 종목 검색 순위 2위를 차지했다. 뉴진스가 소속사
 어도어와의 전속계약 해지를 선언한 이튿날인 지난달 29일
 어도어의 모회사인 하이브 주가가 전날보다 4.08% 하락하며
 20만원 밑으로 주가가 내려갔다. 뉴진스는 전날 기자회견을 통해
 어도어와의 전속계약을 이날부터 해지하겠다고 전격 발표했다.
 뉴진스는 하이브 산하 레이블 걸그룹이 자신들의 콘셉트를 모방하고,
 어도어와 하이브로부터 불합리한 처우를 받아왔다고 주장하고 있다.
 어도어는 "전속계약에서 중요한 내용이라고 할 수 있는 연예활동의
 섭외·교섭이나 지원, 대가의 수령, 정산·정산자료 제공 등을 충실히
 이행했다"며 "전속계약은 2029년 7월 31일까지 유효하다"고 답한 상황이다.
                    행동(낮)
```

5번 행동

```
======================| 미니게임 |======================
1. 특수 미니게임
2. 신문 미니게임
>>
```

6번 행동

```
밤
1일차
==================행동(밤)====================
1. 메뉴 가격 수정
2. 직원 고용
3. 카페 메뉴 확인
4. 정산
5. 미니 게임
6. 잠자기
>>
```

밤

1번 행동

```
>> 1
수정할 메뉴명를 입력해주세요 :
coffee
수정할 가격을 입력해주세요 :
2000
 수정을 다시 진행하시겠습니까? (y/n): n
```

2번 행동

```
>> 2
======================| 주의 사항 |=========================
직원 고용을 진행시 고용 여부와 상관없이 그날 고용은 끝이납니다
주의하여 고용을 진행하여 주시길 바랍니다.
============================================================
직원 고용
직원이름 : 박건우
고용비용 : 6194원
메뉴 추가 개수 : 3개
해당 직원을 고용하시겠습니까?(Y/N) : y
해당직원을 고용하였습니다.
현재 가용 가능한 메뉴의 개수는 4개 입니다.
현재 고용한 직원 수는 (1/3)명 입니다.
```

3번 행동

```
 >> 3
 ==============(coffee)================
 1번 메뉴
 coffee
 2000
 판매량
 15

 ==============(dessert)================
 오늘의 디저트
 dessert
 5000
 판매량
 7
```

4번 행동

```
>> 4
==================| 정산 |=====================
===| 커피 |===
22500

===| 디저트 |===
35000

===| 직원 월급 |===
0

===| 정산 |===
57500
```

5번 행동

```
====================| 미니게임 |====================
1. 특수 미니게임
2. 신문 미니게임
>>
```

6번 행동

```
>> 6

낮
2일차
==================행동(낮)====================
1. 카페 메뉴 만들기
2. 디저트 만들기
3. 카페 메뉴 확인
4. 신문 확인
5. 미니 게임
6. 판매 시작
```

(2) 카페 메뉴 만들기

- 설명 : 카페의 메뉴의 이름과 가격을 입력받아 메뉴를 제작합니다.

- 테스트 결과 스크린샷

```
>> 1
=======================| 주의 사항 |=========================
메뉴의의 이름은 영어로 작성하여 주시길 바랍니다.
왜 한글이 안되는 것일까요....
=============================================================

메뉴명을 입력해주세요 :
coffee
메뉴 가격을 입력해주세요 :
1500
====================행동(낮)====================
```

- 존재하는 메뉴일 경우

```
메뉴명을 입력해주세요 :
coffee
메뉴 가격을 입력해주세요 :
2000
이미 존재하는 메뉴입니다.
```

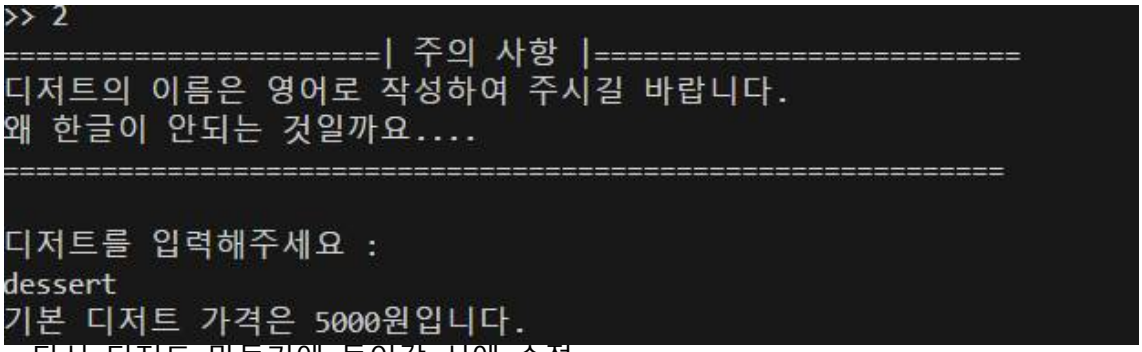
- 생산 가능한 메뉴의 개수를 넘었을 경우

```
메뉴명을 입력해주세요 :
goo
메뉴 가격을 입력해주세요 :
3000
====================행동(낮)====================
1. 카페 메뉴 만들기
2. 디저트 만들기
3. 카페 메뉴 확인
4. 신문 확인
5. 미니 게임
6. 판매 시작
>> 1
만들 수 있는 커피의 개수를 넘었습니다.
```

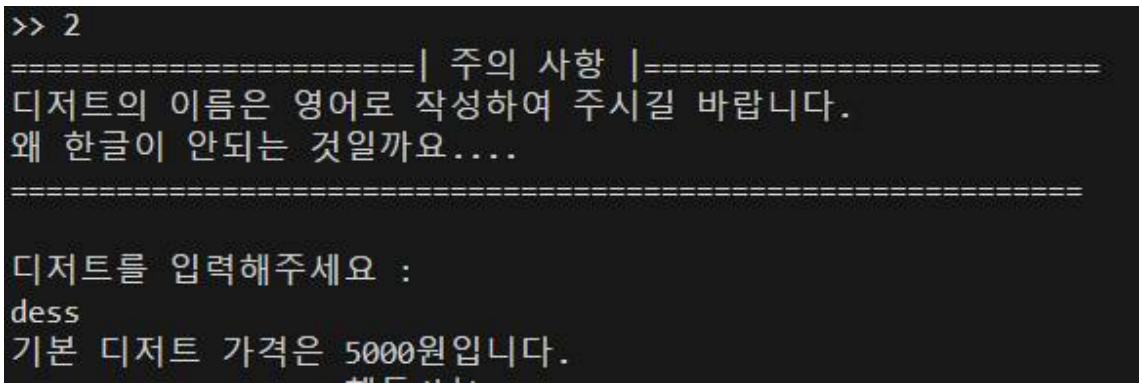
(3) 디저트 만들기

- 설명 : 디저트의 이름을 입력받아 메뉴를 제작합니다.

- 테스트 결과 스크린샷



- 다시 디저트 만들기에 들어갈 시에 수정

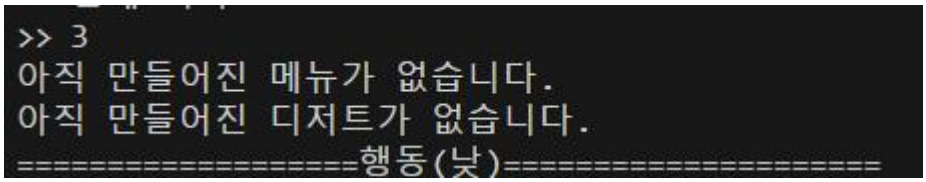


(4) 카페 메뉴 확인

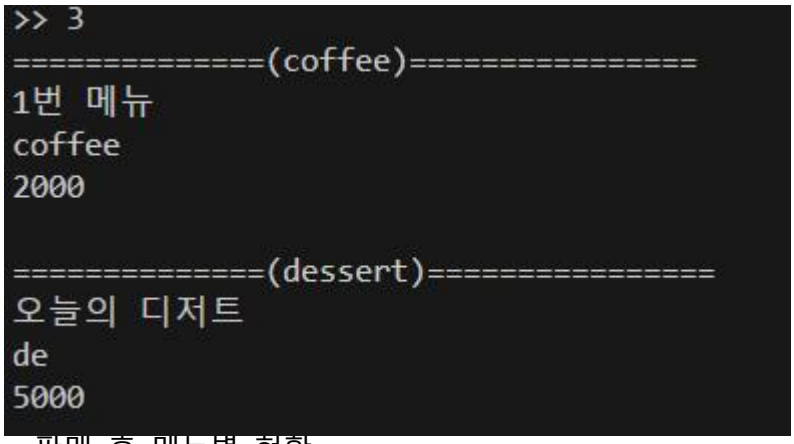
- 설명 : 현재 존재하는 카페 메뉴를 모두 보여줍니다.

- 테스트 결과 스크린샷

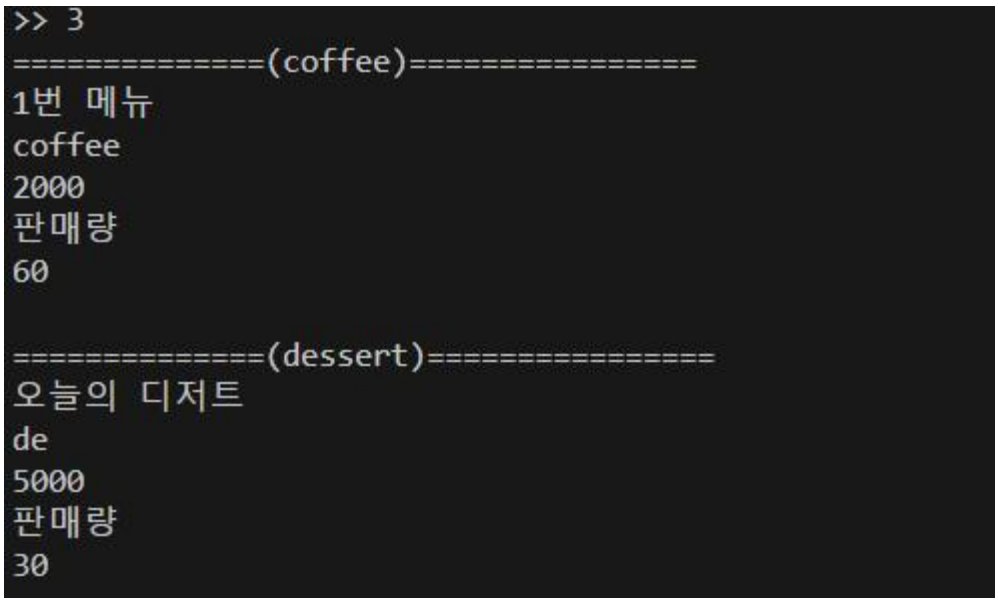
- 만들어진 메뉴가 없을 시에



- 판매전 메뉴별 현황



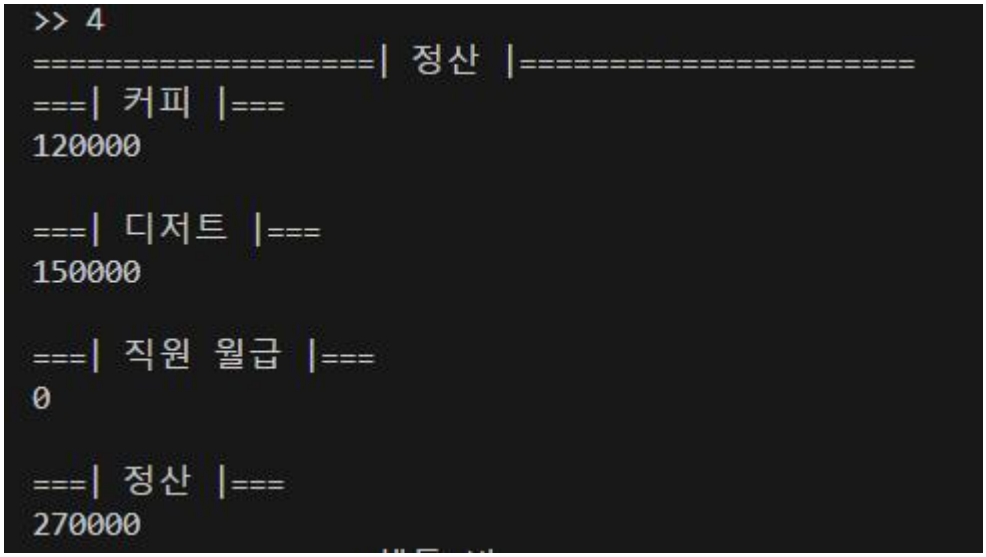
- 판매 후 메뉴별 현황



(5) 정산

- 설명 : 지금까지 판매한 메뉴들의 총 판매량을 모두 합산하여 보여줍니다.

- 테스트 결과 스크린샷

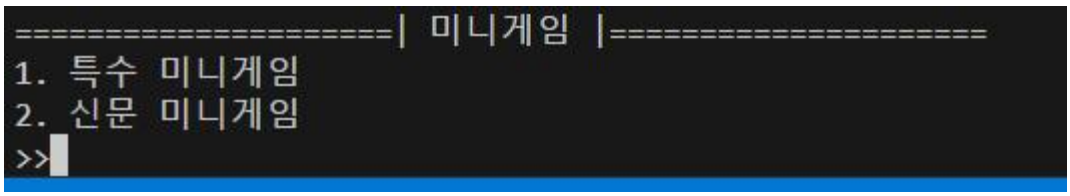


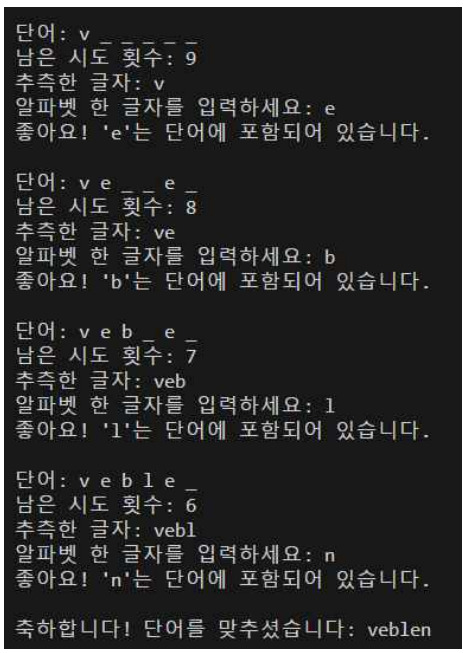
(6) 미니게임

-> 설명 :미니게임을 통해서 그날 신문에 따라 정해지는 키워드를 맞추는 미니게임을 진행합니다,

-> 베블런 효과에 따른 미니게임을 고려중입니다.

- 테스트 결과 스크린샷



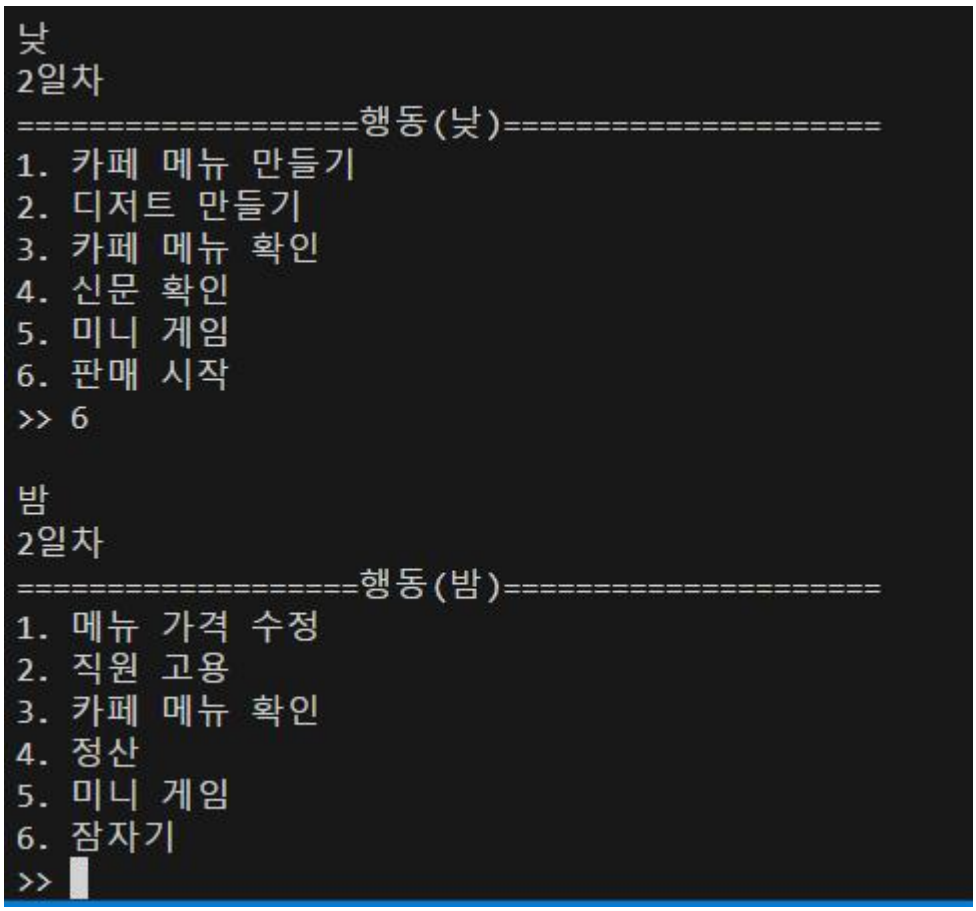


(7) 판매 시작

- 설명 : 현재 게시된 메뉴들로 판매를 시작합니다.

판매 시에 하루가 지나갑니다.

- 테스트 결과 스크린샷



(8) 메뉴 가격 수정

- 설명 : 메뉴의 가격을 수정합니다.

- 테스트 결과 스크린샷

```
>> 3
==============(coffee)================
1번 메뉴
coffee
2000
판매량
60
```

- 메뉴의 가격을 수정합니다.

```
>> 1
수정할 메뉴명를 입력해주세요 :
coffee
수정할 가격을 입력해주세요 :
3000
 수정을 다시 진행하시겠습니까? (y/n): n
==============(coffee)================
1번 메뉴
coffee
3000
판매량
60
```

- 메뉴가 없을 시

```
>> 1
수정할 메뉴명를 입력해주세요 :
de
메뉴 이름을 다시 한번 확인해주세요.
```

(9) 직원 고용

- 설명 : 직원을 고용합니다.

- 테스트 결과 스크린샷

- 고용

```
>> 2
======================| 주의 사항 |=========================
직원 고용을 진행시 고용 여부와 상관없이 그날 고용은 끝이납니다
주의하여 고용을 진행하여 주시길 바랍니다.
============================================================
직원 고용
직원이름 : 김칠우
고용비용 : 9972원
메뉴 추가 개수 : 2개
해당 직원을 고용하시겠습니까?(Y/N) : y
해당직원을 고용하였습니다.
현재 가용 가능한 메뉴의 개수는 3개 입니다.
현재 고용한 직원 수는 (1/3)명 입니다.
```

- 고용 끝

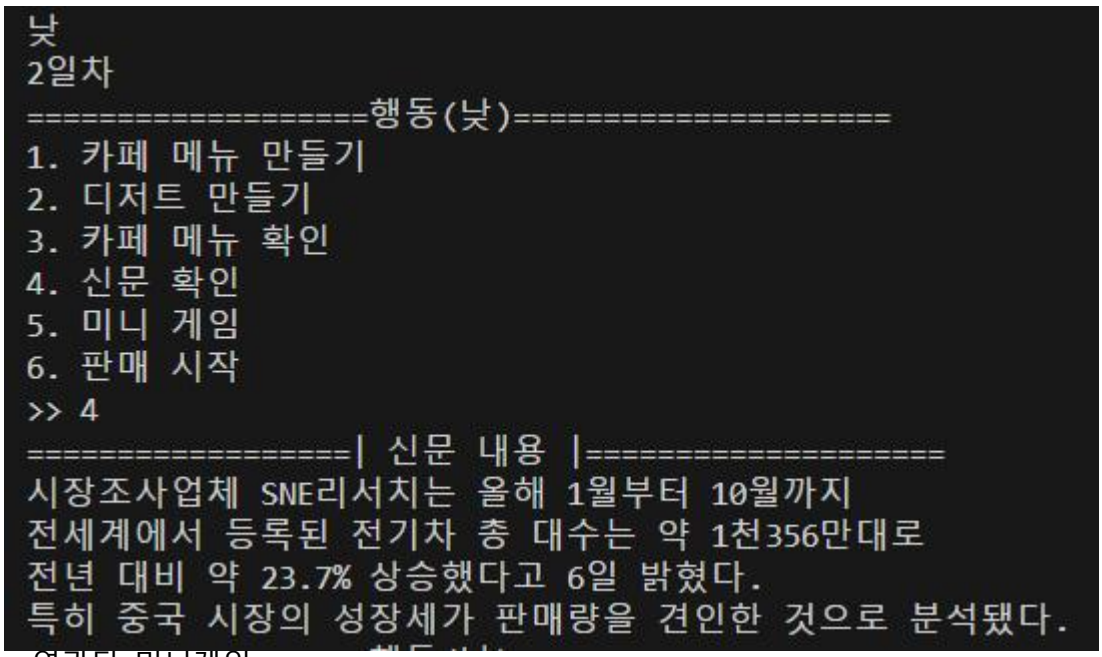
```
====================행동(밤)====================
1. 메뉴 가격 수정
2. 직원 고용
3. 카페 메뉴 확인
4. 정산
5. 미니 게임
6. 잠자기
>> 2
오늘의 직원 고용은 마무리 되었습니다. 내일 다시오세요
```

(10) 신문 확인하기

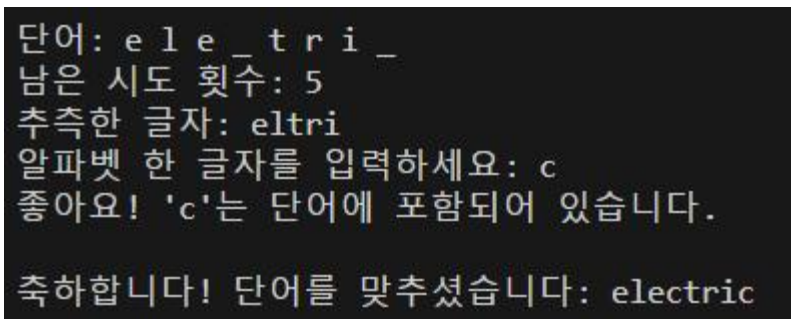
- 설명 : 하루에 한번 신문이 나오게 됩니다.

- 테스트 결과 스크린샷

- 신문



- 연관된 미니게임

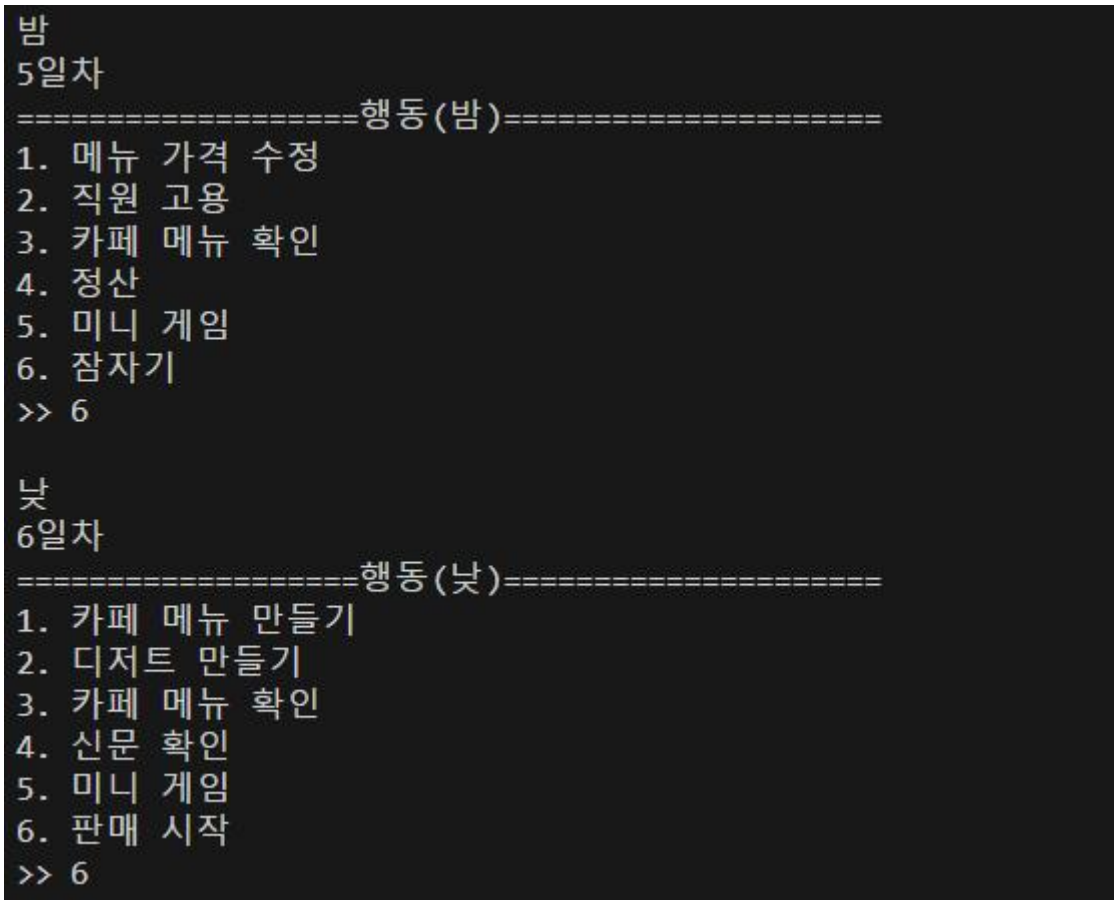


(11 / 12) 낮과 밤 / 시간

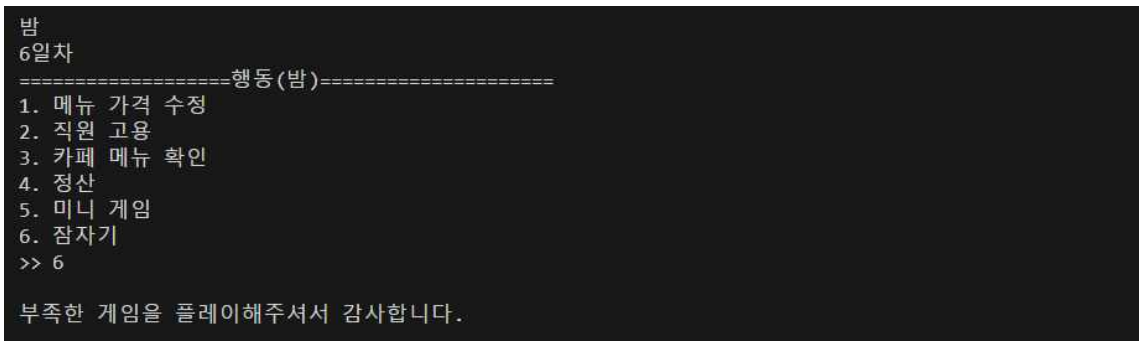
- 설명 : 플레이어는 낮과 밤에 따라 행동양식이 달라집니다.

- 설명 : 플레이어가 행동할 수 있는 시간은 7일으로 제한되어 있습니다.

- 테스트 결과 스크린샷



-엔딩

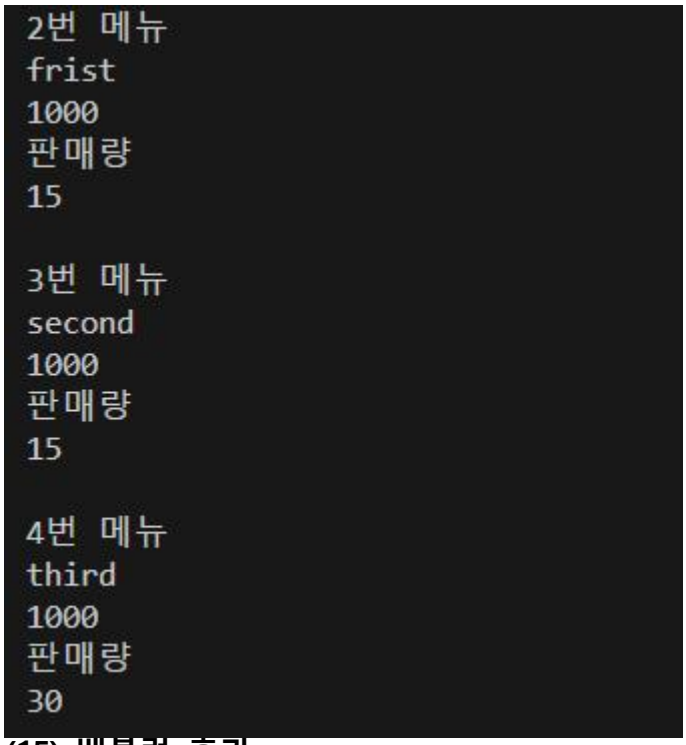


(14) 균형 가격 설정

- 설명 : 새로운 메뉴를 만들수록 더 비싼 메뉴를 제작하여 균형 가격이 각기 다릅니다.

- 테스트 결과 스크린샷

- 같은 가격에도 다른 판매량을 보입니다.

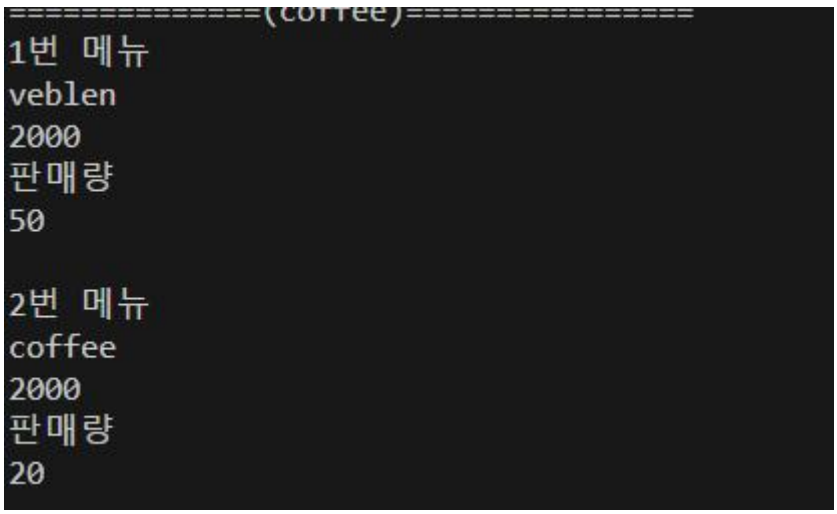


(15) 베블런 효과

- 설명 : 특정한 메뉴의 가격을 높게 설정했을 때 판매량이 증가합니다.

- 테스트 결과 스크린샷

- 같은 가격에도 다른 판매량을 보입니다.



(16) 외부 효과

- 설명 : 하루에 한 번 신문이 나오게 되면 신문에 키워드로 메뉴를 설정 시 디저트가 더 많이 팔립니다.

- 테스트 결과 스크린샷

- 더 비싼 가격에 팔립니다.

```
>> 2
======================| 주의 사항 |=========================
디저트의 이름은 영어로 작성하여 주시길 바랍니다.
왜 한글이 안되는 것일까요....
============================================================

디저트를 입력해주세요 :
chuncheon
기본 디저트 가격은 5000원입니다.
디저트가 더 비싼 가격에 팔립니다!!
```

```
==============(dessert)================
오늘의 디저트
chuncheon
7000
```

5. 계획 대비 변경 사항

1) 기능 6 (미니게임)

- 이전

미니게임을 진행하여 성공 시에 플레이에 도움이 되는 아이템을 지급합니다.

- 이후

미니게임을 통해서 그날 신문에 따라 정해지는 키워드를 맞추는 미니게임을 진행합니다, 베블런 효과에 따른 미니게임을 고려중입니다.

- 사유

아이템으로 간접적으로 2단계를 거쳐서 단어를 얻는 것 보다는 차라리 미니게임으로 바로 알려주는 것이 좀 더 직관적이고 좋은 방안이라고 생각이 들어 바꾸게 되었습니다.

2) 기능 13 (고객 피드백 시스템)

- 이전

설명 : 판매 이후에 고객들이 메뉴에 대해 피드백을 남깁니다.

- 이후

삭제

- 사유

시험 기간이 생각보다 길어지게 되어 있으면 좋은 기능이지만 핵심 기능은 아니라고 생각이 들어 시간 관계상 삭제하게 되었습니다.

3) 기능 14 (균형 가격 설정)

- 이전

새로 만든 메뉴일수록 균형가격이 조금 더 비싸게 조정됩니다.

- 이후

균형 가격의 경우에는 3가지 중 랜덤으로 제공되어집니다.

- 사유

플레이 타임이 짧은 반면에 새로 만든 메뉴마다 균형지점이 모두 다르면 마지막에는 게임 시스템상 가격을 바꾸어 가면서 가격을 유추해야하는데, 너무 어렵다고 생각이 들어, 3가지 균형지점을 두어 다른 메뉴를 만들었을 때도 간접적으로 알아낼 수 있는 방식이 더 좋다고 생각이 들었습니다.

4) 기능 16 (외부 효과)

- 이전

하루에 한 번 신문이 나오며 그 신문에 단어에 대한 힌트가 제공됩니다.

- 이후

하루에 한 번 신문이 나오며 신문을 통해 단어를 유추하거나 미니게임에서 그 단어 맞추기 게임이 진행됩니다.

- 사유

결국 미니게임으로 제공되는 아이템을 단어에 대한 힌트로 생각했는데 너무 동떨어져 있는 느낌이 들어 각 활동간에 긴밀한 연관성이 있는 것이 더 좋다고 생각이 들어 바꾸었습니다.

6. 느낀점

처음 계획을 세웠을 때는 헤더 파일을 분리하여 진행하려고 했습니다.

하지만 초석을 잡고나서 진행하려고 만들다보니 너무 많은 코드를 작성하게 되었고, 분리를 진행하려고 했을 땐 어디가 문제인지 모르게 계속 오류가 발생하게 되었습니다. 그렇게 돼서 코드를 분리하지 못하였고, 코드는 계속 길어져서 가독성이 너무 떨어져 있는 상태로 진행하게 되었습니다. 이는 진행하는데 문제가 되었고, 오류가 발생하였을 때에도 고치는데 너무 많은 시간을 소모하게 되었습니다. 이에 따라서, 코드 분리의 중요성과 가독성이 얼마나 중요한지 깨닫게 되었습니다. 다시금 이런 프로젝트를 진행하게 된다면 우선 코드를 분리를 하고 진행하여 나중에 고생하는 일이 없도록 하려고 합니다.

또한 처음에 계획을 만들었을 때 처음 계획처럼 진행하기는 중간중간 어려운 점이 있었고 수정을 몇차례 진행하면서 수업 시간에 배웠던 내용들을 많이 적용시켜보고자 하였습니다. 하지만 결과적으로 많은 내용을 담지 못했다고 생각이 들어 아쉬움도 남는 것 같습니다.

하지만 그에 따라서 다시 pdf를 읽게되면서 남은 지식도 있다고 생각이 들어서 이번 프로젝트를 통해 많은 것을 얻어가는 시간이 되었던 것 같습니다.