소나축 실습파일 코드해설

초기값 설정

```
const sword = [
```

무기 정보를 담고 있는 sword 배열을 선언합니다.

```
{ upcost : 1000,
  upsuccess : 0.99,
  sellcost : 0,
  img : "1.png",
  name : "나뭇가지"
},
```

sword 배열의 요소로 객체를 사용해 여러 정보를 하나의 인덱스에 담을 수 있습니다.

`upcost` : 강화에 필요한 강화 비용

`upsuccess` : 강화에 성공할 수 있는 확률

`sellcost` : 판매했을 때 얻을 수 있는 수익

`img` : 검의 이미지

`name` : 검의 이름

```
{ upcost: 1500, upsuccess: 0.95, sellcost: 1000, img: "2.png", name : "목검"},
 //2강화 검 정보
{ upcost: 2000, upsuccess: 0.9, sellcost: 1500, img: "3.png", name : "장미칼"},
 //3강화 검 정보
{ upcost: 2500, upsuccess: 0.85, sellcost: 2000, img: "4.png", name : "프로스트 블레이드"},
 //4강화 검 정보
{ upcost: 3000, upsuccess: 0.8, sellcost: 3000, img: "5.png", name : "트윈 슬래셔"},
 //5강화 검 정보
{ upcost: 3500, upsuccess: 0.75, sellcost: 4000, img: "6.png", name : "용의 송곳"},
 //6강화 검 정보
{ upcost: 4000, upsuccess: 0.7, sellcost: 5000, img: "7.png", name : "바나나"},
 //7강화 검 정보
{ upcost: 4500, upsuccess: 0.65, sellcost: 10000, img: "8.png", name : "VIP TeamLog Express"},
 //8강화 검 정보
{ upcost: 5000, upsuccess: 0.6, sellcost: 20000, img: "9.png", name : "쉐도우 블레이드"},
 //9강화 검 정보
{ upcost: 5500, upsuccess: 0.55, sellcost: 30000, img: "10.png", name : "X카츄"},
//10강화 검 정보
```

```
{ upcost: 6000, upsuccess: 0.5, sellcost: 40000, img: "11.png", name : "악마의 혀"},
 //11강화 검 정보
{ upcost: 6500, upsuccess: 0.45, sellcost: 50000, img: "12.png", name : "무한검"},
 //12강화 검 정보
{ upcost: 7000, upsuccess: 0.4, sellcost: 60000, img: "13.png", name : "붉은 망치"},
 //13강화 검 정보
{ upcost: 7500, upsuccess: 0.35, sellcost: 80000, img: "14.png", name : "사신의 낫"},
 //14강화 검 정보
{ upcost: 8000, upsuccess: 0.3, sellcost: 1000000, img: "15.png", name : "TeamLog검"},
 //15강화 검 정보
];
```

2강부터 15강 까지의 무기 정보를 각각 객체에 담아 sword 배열 안에 저장합니다.

```
const inventory = [];
```

무기 도감 정보를 담고 있는 inventory 배열을 선언합니다.

이 배열의 길이는 보유한 무기 개수 즉, 도감에 있는 무기 개수를 의미합니다.

```
let enhanceLevel = 0;
let money = 20000;
```

강화 진행도 (= 강화 단계, 강화 레벨) 인 enhanceLevel 변수를 0으로 설정합니다.

소지금인 money 변수를 20000으로 설정합니다.

게임이 시작될 때 처음 강화 진행도는 0으로, 소지금은 20000원으로 주는 것입니다.

함수 - updateWeaponInfo

```
function updateWeaponInfo() {
```

무기 정보를 업데이트하는 updateWeaponInfo 함수를 선언합니다.

이 함수가 호출되면 무기 정보, 소지금, 강화 단계 등 화면에 표시되는 요소들을 업데이트합니다.

```
const weapon = sword[enhanceLevel];
```

weapon 변수에 현재 검 정보를 저장합니다.

각 검의 정보들이 저장되어 있는 sword 배열의 인덱스로 현재 강화 단계인 enhanceLevel 을 입력해서

현재 강화 단계에 해당하는 검의 정보가 담긴 객체를 가져옵니다.

```
document.getElementById('moneyValue').textContent = `${money} 원`;
```

HTML 문서에서 getElementById 메소드로 Id가 ‘moneyValue’와 일치하는 요소를 불러옵니다.

그리고 템플릿 리터럴을 사용해 가져온 요소의 텍스트를 소지금 변수의 값과 단위인 ‘원’으로 표시합니다.

```
document.getElementById('sellCost').textContent = `${weapon.sellcost} 원`;
```

HTML 문서에서 getElementById 메소드로 Id가 ‘sellCost’와 일치하는 요소를 불러옵니다.

그리고 템플릿 리터럴을 사용해 가져온 요소의 텍스트를 현재 무기 정보가 담긴 객체에서 판매가격에 해당하는

값과 단위인 ‘원’으로 표시합니다.

```
document.getElementById('enhanceCost').textContent = `${weapon.upcost} 원`;
```

HTML 문서에서 getElementById 메소드로 Id가 ‘enhanceCost’와 일치하는 요소를 불러옵니다.

그리고 템플릿 리터럴을 사용해 가져온 요소의 텍스트를 현재 무기 정보가 담긴 객체에서 강화비용에 해당하는

값과 단위인 ‘원’으로 표시합니다.

```
document.getElementById('inventoryCount').textContent = `총 ${inventory.length}개`;
```

HTML 문서에서 getElementById 메소드로 Id가 ‘inventoryCount’와 일치하는 요소를 불러옵니다.

그리고 템플릿 리터럴을 사용해 가져온 요소의 텍스트를 무기 도감에 있는 무기들을 저장한 inventory 배열의

길이, 즉, 무기 도감에 있는 무기 개수와 단위인 ‘개’로 표시합니다.

```
const innerBox = document.querySelector('.inner_box');
```

HTML 문서에서 클래스가 inner_box인 요소를 가져와 innerBox 변수에 저장합니다.

이 요소에는 검의 이미지, 이름, 성공 확률을 표시합니다.

```
const equipImg = innerBox.querySelector('img');
```

innerBox에 저장한 요소 내부에서 img 요소를 가져와 equipImg 변수에 저장합니다.

이 요소로 검의 이미지를 표시합니다.

```
const equipName = innerBox.querySelector('.name');
```

innerBox에 저장한 요소 내부에서 클래스가 name인 요소를 가져와 equipName 변수에 저장합니다.

이 요소로 검의 이름과 강화 단계를 표시합니다.

```
const equipDetail = innerBox.querySelector('.detail');
```

innerBox에 저장한 요소 내부에서 클래스가 detail인 요소를 가져와 equipDetail 변수에 저장합니다.

이 요소로 검의 성공 확률을 표시합니다.

```
equipImg.src = `./public/images/swords/${weapon.img}`;
```

equipImg 변수에 저장된 Img 태그의 src 를 현재 검의 이미지 경로로 설정합니다.

public 폴더의 imges 폴더의 swords 폴더에 모든 검의 이미지 파일이 있으므로 마지막에 weapon 변수에

저장된 현재 무기 정보의 img 이름을 넣어서 현재 검의 이미지를 불러옵니다.

```
equipName.textContent = `+${enhanceLevel +1} ${weapon.name}`;
```

equipName 변수에 저장된 요소의 텍스트를 현재 강화 단계인 enhanceLevel과 weapon 변수에 저장된

현재 검의 이름을 가져와 표시합니다.

enhanceLevel은 0으로 초기화되어있고, 실제 강화 단계는 1부터 시작하기 때문에 강화 단계를 표시할 때

enhanceLevel에 1을 더하여 사용합니다.

```
equipDetail.textContent = `성공 확률 ${(weapon.upsuccess * 100)}.toFixed(0) %`;
```

equipDetail 변수에 저장된 요소의 텍스트를 성공확률과 현재 검의 성공확률로 표시합니다.

weapon 변수에 저장된 현재 검의 성공 확률 값을 가져와 100을 곱하고, toFixed 메소드를 사용해서

소수점 이하를 모두 반올림하여 성공 확률을 정수로 바꾸어 표시합니다.

```
if (inventory.length <= enhanceLevel) {
```

보유한 무기들을 저장한 inventory 배열의 길이와 현재 강화 단계를 비교합니다.

inventory 배열의 길이가 현재 강화 단계보다 작거나 같다면 현재 무기가 보유한 무기 목록에 저장되지 않았다는

것을 의미합니다.

```
inventory.push(weapon);
```

따라서 보유 무기 목록인 inventory 배열에 push 배열 메소드를 사용해 weapon 변수에 저장된

현재 무기 정보를 담은 객체를 마지막 요소로 추가합니다.

```
const itemWrapper = document.querySelector('.item_wrapper');
```

HTML 문서에서 클래스가 item_wrapper 인 요소를 itemWrapper 변수에 저장합니다.

이 요소에 보유한 무기들을 표시합니다.

게임을 실행했을 때 무기 도감 역할을 하는 요소입니다.

```
const newItem = document.createElement('div');
```

HTML 문서에 createElement 메소드를 사용해 div 요소를 새로 생성하고, 생성한 요소를 newItem 변수에

저장합니다.

여기서 생성한 요소에 각 검의 이미지들이 들어갑니다.

```
newItem.className = 'item';
```

윗줄에서 생성한 div 요소의 클래스를 item 으로 설정합니다.

```
newItem.innerHTML = `<img src="./public/images/swords/${weapon.img}" />`;
```

innerHTML 메소드를 사용해 newItem 변수에 저장된 요소 내부의 img 태그를 설정합니다.

img 태그의 경로를 현재 검의 이미지 경로로 설정하여 현재 검의 이미지를 불러옵니다.

```
itemWrapper.appendChild(newItem);
}}
```

appendChild 메소드를 사용해 newItem 변수에 저장된 무기 이미지 요소를 무기 도감 요소인

itemWrapper 요소의 자식 노드로 추가합니다.

보유 무기 목록에 현재 검이 있는지 확인하는 if 문의 중괄호를 닫고, updateWeaponInfo 함수의 중괄호를 닫아

updateWeaponInfo 함수를 완성합니다.

```
updateWeaponInfo();
```

updateWeaponInfo 함수를 완성하고 바로 호출해줍니다.

여기서 이 함수의 역할은 게임을 처음 실행했을 때 초기값을 표시할 요소들을

HTML 문서에서 불러오는 역할을 합니다.

따라서 여기서 이 함수를 한번 호출해 주었다고 게임을 처음 실행했을 때 초기값이 모두 잘 화면에 표시되는 것은

아닙니다.

함수 - sellWeapon

```
function sellWeapon() {
```

게임 화면에서 '판매하기' 버튼을 클릭했을 때 호출되어 무기를 판매하는 sellWeapon 함수를 선언합니다.

```
const weapon = sword[enhanceLevel];
```

weapon 변수에 현재 검 정보를 저장합니다.

각 검의 정보들이 저장되어 있는 sword 배열의 인덱스로 현재 강화 단계인 enhanceLevel 을 입력해서

현재 강화 단계에 해당하는 검의 정보가 담긴 객체를 가져옵니다.

```
money += weapon.sellcost;
```

weapon 변수에 저장된 현재 무기 정보 객체에서 판매가격인 sellcost의 값만큼 소지금인 money에 추가합니다.

```
enhanceLevel = 0;
```

현재 무기를 판매했기 때문에 검 강화 단계를 처음으로 초기화합니다.

```
updateWeaponInfo();
}
```

updateWeaponInfo 함수를 호출하여 판매하기 기능을 실행하며 변경된 정보들을 업데이트합니다.

sellWeapon 함수의 중괄호를 닫아 함수를 완성합니다.

함수 - enhanceWeapon

```
function enhanceWeapon() {
```

게임 화면에서 '강화하기' 버튼을 클릭했을 때 호출되어 무기를 강화하는 enhanceWeapon 함수를 선언합니다.

```
const weapon = sword[enhanceLevel]; // 현재 강화 상태에 해당하는 무기 정보 가져오기
```

weapon 변수에 현재 검 정보를 저장합니다.

각 검의 정보들이 저장되어 있는 sword 배열의 인덱스로 현재 강화 단계인 enhanceLevel 을 입력해서

현재 강화 단계에 해당하는 검의 정보가 담긴 객체를 가져옵니다.

```
if (money >= weapon.upcost) {
```

소지금이 저장된 money 변수와 현재 검의 정보가 저장된 weapon에서 강화에 필요한 비용인 upcost를

비교합니다.

소지금이 현재 검을 강화하기 위해 필요한 금액보다 많거나 같아야 강화를 시도할 수 있기 때문에 ≥ 부등호를

사용하여 비교합니다.

```
const isSuccess = Math.random() < weapon.upsuccess;
```

위의 조건식이 참이라면 강화를 시도할 수 있습니다.

Math.random() 메소드로 0 이상 1 미만의 실수를 랜덤으로 생성합니다. 여기서 생성한 수는

현재 무기 정보인 weapon의 성공확률 upsuccess와 비교하여 강화 성공 여부를 결정하게 됩니다.

Math.random() 으로 생성한 수가 성공 확률보다 작다면 부등호 방향에 따라 isSuccess 변수에 true 값을

저장하고, Math.random() 으로 생성한 수가 성공 확률보다 크거나 같다면 isSuccess 변수에 false 값을

저장합니다.

```
if (isSuccess) {
```

조건식에 isSuccess 변수가 있으므로 isSuccess의 값이 true 인지 false 인지에 따라 조건문의 참과 거짓이

결정됩니다. 만약 윗줄에서 isSuccess 에 true 값이 저장되었다면 강화에 성공했다는 것을 의미하고, 조건문

안의 코드가 실행됩니다.

```
enhanceLevel++;
```

강화에 성공했기 때문에 강화 단계를 1 증가시킵니다.

```
money -= weapon.upcost;
```

또한 소지금에서 현재 무기 정보인 weapon 에서 강화에 필요한 비용인 upcost 값만큼 감소시킵니다.

```
} else {
```

만약 `const isSuccess = Math.random() < weapon.upsuccess;` 에서 isSuccess에 false 값이 저장되었다면 강화에 실패하여 else 문이 실행됩니다.

```
enhanceLevel = 0;
```

강화에 실패하였기 때문에 강화 단계를 0으로 초기화합니다.

```
alert(`강화 실패!`);
}
```

그리고 강화에 실패했다는 것을 alert 함수로 알림창을 띄워서 알려줍니다.

else 문의 중괄호를 닫아 강화 확률이 모자라 강화에 실패했을 때 실행되는 기능을 마무리합니다.

```
else {
    alert('소지금이 부족합니다.');
  }
```

이 else 문은 애초에 소지금이 강화를 진행할 수 없을 만큼 적을 때 실행됩니다.

강화를 시도조차 하지 않았으니 강화단계나 소지금에는 변화가 없고 alert 함수로 소지금이 부족하다는

알림창을 띄워줍니다.

```
updateWeaponInfo();
```

updateWeaponInfo 함수를 호출해서 강화하기 기능을 진행하면서 변경된 정보들을 화면에 업데이트합니다.

```
const inventoryCount = inventory.length;
```

보유 무기 목록인 inventory 배열의 길이이를 inventoryCount 변수에 저장합니다.

```
if (inventory.length <= enhanceLevel) {
```

inventoryCount 변수에 저장한 값이 아니라 현재 보유 무기 목록의 길이, 즉, 보유한 무기 개수를 현재 강화 단계와 비교합니다.

```
inventoryCount += 1;
}
```

만약 보유 무기 개수가 현재 강화 단계보다 작거나 같다면 무기 개수가 더 적다는 뜻이기 때문에 inventoryCount에 1을 증가시켜 보유 무기 개수를 업데이트합니다.

그리고 중괄호를 닫아서 보유한 무기 개수를 업데이트하는 코드를 마무리합니다.

```
document.getElementById('inventoryCount').textContent = `총 ${inventoryCount}개`;
}
```

HTML 문서에서 getElementById 메소드로 id가 inventoryCount인 요소를 가져와 텍스트를 윗줄에서 업데이트한 보유 무기 개수인 inventoryCount 값으로 표시합니다.

그리고 enhanceWeapon 함수의 중괄호를 닫아 함수를 완성합니다.

```
updateWeaponInfo();
```

마지막으로 updateWeaponInfo 함수를 호출해 줍니다.

처음 호출 때 불러온 요소들에 값을 화면에 표시하는 역할을 합니다.