

# DWA\_07.4 Knowledge Check\_DWA7

1. Which were the three best abstractions, and why?

- O- open closed principle

```
1  /**
2   * this factory function takes a selector from the html as an element
3   * and a object to populate that selector. It also creates the first element with the text 'any' and value =null
4   * @param {Element} element- the selector that you want to display your data on
5   * @param {Object} object - a list of items that you want to be displayed in your selector
6   */
7  const populateSelect = (element, object) => {
8      const emptyOption = document.createElement("option");
9      emptyOption.value = null;
10     emptyOption.text = 'Any';
11     element.appendChild(emptyOption);
12
13     for (const key in object) {
14         const name = object[key];
15         const newOption = document.createElement("option");
16         newOption.value = key;
17         newOption.textContent = name;
18         element.appendChild(newOption);
19     }
20 };
```

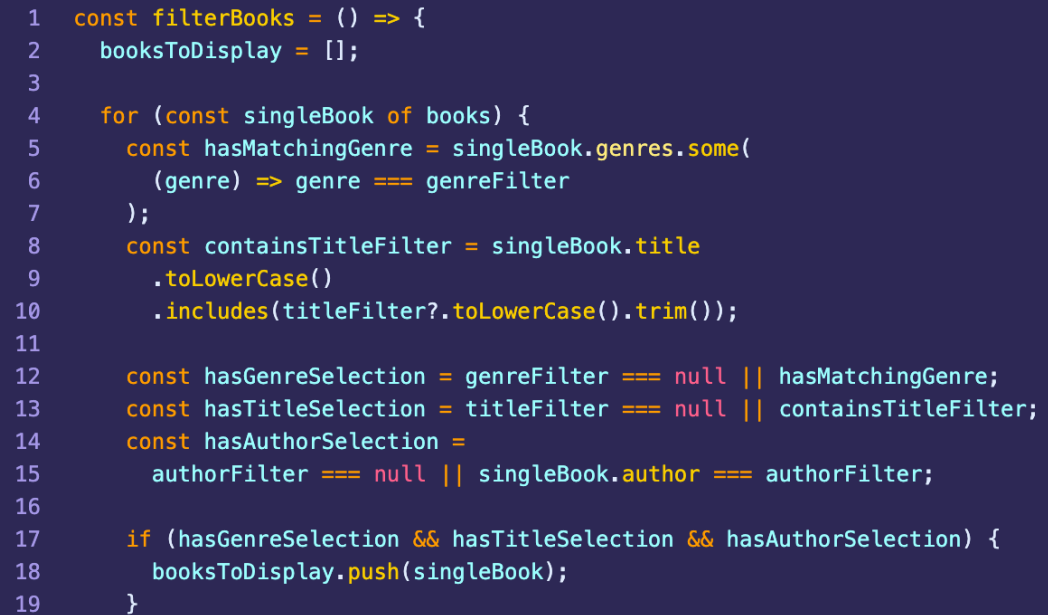
```
1  populateSelect(HTML.search.genre, genres);
2  populateSelect(HTML.search.author, authors);
```

- I believe this is part of the open closed principle. If I make a new element and new object, then call this function, without editing the function it will populate a new element.

---

2. Which were the three worst abstractions, and why?

- **S** - Single-responsibility Principle



```
1  const filterBooks = () => {
2    booksToDisplay = [];
3
4    for (const singleBook of books) {
5      const hasMatchingGenre = singleBook.genres.some(
6        (genre) => genre === genreFilter
7      );
8      const containsTitleFilter = singleBook.title
9        .toLowerCase()
10       .includes(titleFilter?.toLowerCase().trim());
11
12      const hasGenreSelection = genreFilter === null || hasMatchingGenre;
13      const hasTitleSelection = titleFilter === null || containsTitleFilter;
14      const hasAuthorSelection =
15        authorFilter === null || singleBook.author === authorFilter;
16
17      if (hasGenreSelection && hasTitleSelection && hasAuthorSelection) {
18        booksToDisplay.push(singleBook);
19      }
20    }
21  }
```

---

3. How can The three worst abstractions be improved via SOLID principles.

- 1) This should be broken up into separate functions, `filterTitle()` `filterGenre()` and `filterAuthor()`, `applyFilters()` This will also allow me to add extra filter functions and easily add them to the apply filter function as well as editing one of them could be easier
-