Solutions

Each answer shows the steps, then the **result**.

2.1

- (i) indexes 2 to 4: `"ane"`
- (ii) **3**
- (iii) **-1**

2.2

```
int count = 0;
for (int i = 0; i < s.length(); i++)
{
    if (s.substring(i, i + 1).equals("a")) count++;
}
```

- loops over every index
- takes one character with `substring(i, i + 1)`
- compares with `.equals` and counts

2.3

- windows `"aa"` at 0, 1, 2: **3**

3.1 A

- indexes 0, 2, 4: `b`, `n`, `n`

3.2 B

- the result of `toUpperCase()` is not stored, so `t` is still `"go"`

3.3 C

- `i` reaches `s.length() - 1`, so `substring(i, i + 2)` passes the end

3.4

(a)

```
public static boolean isPalindrome(String word)
{
    String rev = "";
    for (int i = word.length() - 1; i >= 0; i--)
    {
        rev += word.substring(i, i + 1);
    }
    return rev.equals(word);
}
```

- builds the reverse (or compares character `i` with character `length - 1 - i`)
- correct loop bounds
- compares with `.equals`
- returns the `boolean` result

(b)

```
public static int vowels(String word)
{
    int count = 0;
    for (int i = 0; i < word.length(); i++)
    {
        if ("aeiou".indexOf(word.substring(i, i + 1)) >= 0) count++;
    }
    return count;
}
```

- loops over every character
- tests each with `indexOf` against the vowels, `>= 0`
- counts and returns