**Step 31**

The CSS clip property is used to define the visible portions of an element.   
Set the span[class~="sr-only"] selector to have a clip property of rect(1px, 1px, 1px, 1px).

The clip-path property determines the shape the clip property should take. Use both the clip-path and -webkit-clip-path selectors to set the value to inset(50%), forming the clip-path into a rectangle within the element.

**Step 43**

Style the text within your #years element by creating a #years span[class] selector. The span[class] syntax will target any span element that has a class attribute set, regardless of the attribute's value.

**Step 45**

Before you start diving in to the table itself, your span elements are currently bolded. Create a span:not(.sr-only) selector and give it a font-weight property set to normal.

The :not() pseudo-selector is used to target all elements that do not match the selector - in this case, any of your span elements that do not have the span[class~="sr-only"] class. This ensures that your earlier rules for the sr-only class are not overwritten.

**Step 46**

Rather than having to constantly double-check you are not overwriting your earlier properties, you can use the !important keyword to ensure these properties are always applied, regardless of order or specificity.

Give each property in your span[class~="sr-only"] selector an !important keyword. Do not change any of the values.

**Step 52**

Create a selector to target your td elements within your table body. Give them a width to fill the viewport, with a minimum and maximum of 4rem. This approach ensures that the width is fixed, whereas setting width specifically would allow the elements to shrink to the container (because we are using flexbox).

**Step 54**

The [attribute="value"] selector targets any element that has an attribute with a specific value. Create a tr[class="total"] selector to target specifically your tr elements with the total class. Give it a bottom border of 4px double #0a0a23 and make the font bold.