Web Technologies

WorldSkills 2024 National Competition

HUNGARY

Final

Module A – Speedtest

Submitted by:

Skills IT

Contents

Introduction

This module will test your ability to apply your knowledge of HTML, CSS and Javascript effectively and creatively. You will be asked to complete several tasks in 1 hour. There will be some easy, routine tasks and some more complex, thought-provoking tasks. You can typically spend 5-15 minutes on each task.

Tasks

Task 1 - HTML/CSS: Sticky footer *(easy)*

Create a fixed footer that sticks to the bottom of the page at all times, even if the page is not long enough to fill the screen.
The footer should be 64px tall and have a background color of #222222. The content of the footer should be:

```
Copyright © 2023
```

The text should be centered both vertically and horizontally. The text color should be `#f5f5f5`.

Work in `task1/index.html` and `task1/style.css`.
Preview your work at http://speedtest.ub2023-YY.hu/task1, where YY is your workstation number.

Task 2 - JavaScript: Replace strings *(easy)*

Write a JavaScript function that given some sentence, replaces the search term with a replacement string. Subsequent replacements replace previous versions of the string.

Work in `task2/replaceStrings.js`.

```
function replaceStrings(sentence: string, replacements: string[][]): string;
```

Example:

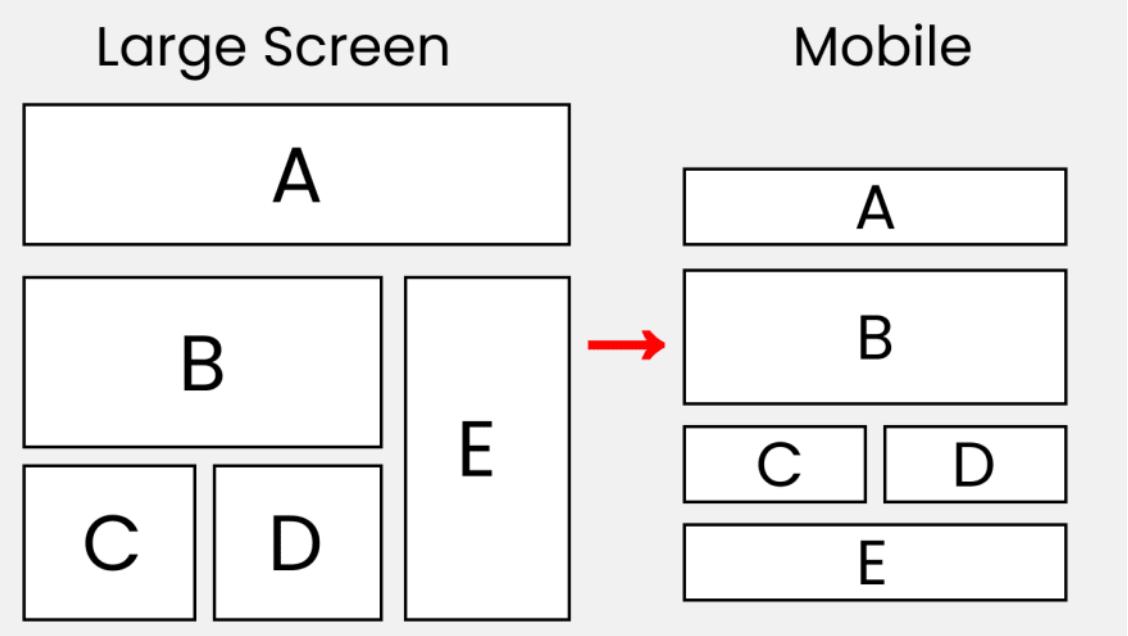
sentence: `"Hello, World!"`
replacements: `[["Hello", "Hey"], ["Hey", "Hi"]], ["World", "Universe"]]`
output: `"Hi, Universe!"`

Task 3 - HTML/CSS: Responsive grid layout *(easy)*

Your task is to create the following layout using HTML and CSS. The layout should be responsive and work on large screens and mobile devices as shown in the image below.

The breakpoint should be at 768px screen width.

Work in `task3/index.html` and `task3/style.css`.
Preview your work at http://speedtest.ub2023-YY.hu/task3, where YY is your workstation number.



Task 4 - JavaScript: Rotate string *(medium)*

Write a function that receives two strings and returns the number of characters we would need to rotate the first string forward to match the second.

Work in `task4/shiftedDiff.js`.

```
function shiftedDiff(first: string, second: string): number;
```

Example:
take the strings “fatigue” and “tiguefa”. In this case, the first string can be rotated 5 characters forward to produce the second string, so 5 would be returned. Here are the steps:

- no rotations: `"fatigue"`

- 1st rotation: "efatigu"
- 2nd rotation: "uefatig"
- 3rd rotation: "guefati"
- 4th rotation: "iguefat"
- 5th rotation: "tiguefa"

If the second string isn't a valid rotation of the first string, the method should return -1.

Examples:

- "coffee", "eecoff" => 2
- "eecoff", "coffee" => 4
- "moose", "Moose" => -1
- "isn't", "'tisn" => 2
- "Esham", "Esham" => 0
- "dog", "god" => -1

Task 5 - JavaScript: Fastest runners *(medium)*

You are given an array of runners containing their paces of each kilometer after a 7 km race in the following format (the pace values are in `mm:ss` format):

```
[
    {
        "name": "Alice",
        "paces": ["5:50", "6:00", "6:06", "6:07", "6:08", "6:19", "6:28"]
    },
    ...
]
```

Return an array that only contains the runners that completed the race in less time than the average of all runners. The array should be ordered by finishing time, so the runner who finished first is first in the array.

Include only the name, the average pace and the fastest pace of each runner. The result should be in the following format:

```
[
    {
        "name": "Alice",
        "averagePace": "6:10",
        "fastestPace": "5:50"
    },
    ...
]
```

Work in `task5/fastestRunners.js`.

You can find example data in `task5/runners.json`.

```
function fastestRunners(
    runners: { name: string; paces: string[] }[]
): { name: string; averagePace: string; fastestPace: string }[];
```

Task 6 - CSS Selectors *(hard)*

Your task is to create CSS rules that will target only the rainbow colored marbles.

Open http://speedtest.ub2023-YY.hu/task6 (where YY is your workstation number) in your browser and read the instructions at the top. Make sure to follow all restrictions described.

Examine the HTML to create the correct selectors in `task6/selectors.css`. The task is composed of multiple subtasks; write only one selector for each subtask, replacing `/* @todo */` with your own code.

Please carefully follow these rules:

- The selector must begin with the task ID, e.g. `#task-1`.
- You are not allowed to edit the HTML.
- You are not allowed to use the `data-target` attribute.
- You are not allowed to use any of the following selectors:
  - `:nth-child`
  - `:nth-last-child`
  - `:nth-of-type`
  - `:nth-last-of-type`
  - any use of `+`, `~` or `,`

You are only allowed to use 1 selector per task, do not create additional rules:

- Allowed: `#task-1 .marble { ... }`
- Not allowed: `#task-1 .blue, #task-1 .red { ... }`

See `Task 0` for an example.

Work only in `task6/selectors.css`.
Preview your work at http://speedtest.ub2023-YY.hu/task6, where YY is your workstation number.
Do not modify `task6/index.html` and `task6/lib.css`. Do not use forbidden selectors.

Task 7 - CSS Counter *(hard)*

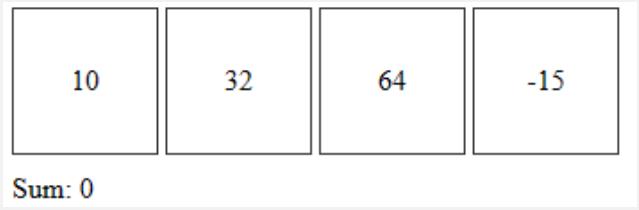
Create the following application using **only HTML and CSS**.

Create 4 boxes, with the following numbers in them consecutively: `10`, `32`, `64`, `-15`. Each box should function like a checkbox: when the user clicks on it, it should be highlighted with a gold background color, and clicking it again should remove the highlight.

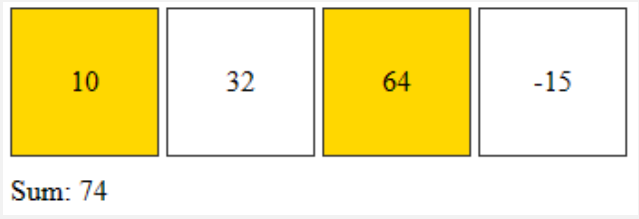
Below the boxes, display the sum of the numbers that are currently checked.

Example:

Default state:



After clicking `10` and `64`:



All combinations of checked boxes should work as expected.

Work in `task7/index.html` and `task7/style.css`.
Preview your work at http://speedtest.ub2023-YY.hu/task7, where YY is your workstation number.
Use only HTML and CSS. Do not use JavaScript.

Hint: Use Zeal, keywords: `counter`, `counter-increment`