

Frontend

Zadanie 1: "foobar".info

Napisz kod który sprawi, że `"foobar".info()`; zwróci obiekt zawierający informacje o długości łańcucha znaków, jego pierwszym i ostatnim znaku, podobnie jak w podanym przykładzie:

```
console.log("foobar".info());  
Object {length: 6, firstChar: "f", lastChar: "r"}
```

Zadanie 2: jQuery

Poniższa funkcja zawiera wiele zbędnego kodu. Sprowadź ją do jak najprostszej postaci. W przypadku nieistnienia poszukiwanego elementu funkcja powinna zwrócić wartość o wartości logicznej `false` (`""`, `undefined`, `null`, itp. są dopuszczalne). Przyjmij, że masz dostępne jQuery.

```
function getTargetContent() {  
    var foo = $('.foo');  
    if (foo.length > 0) {  
        var bar = foo.find('.bar');  
        if (bar.length > 0) {  
            var content = bar.find('#target');  
            return content.text();  
        }  
    }  
}
```

Struktura HTML na której operuje funkcja:

```
<html><body>  
<div class="foo">  
    <div class="bar">  
        <span id="target">  
            content!  
        </span>  
    </div>  
</div>  
</body></html>
```

Zadanie 3: liczba z random.org

Napisz z użyciem **jQuery** kod, który uruchomi się w momencie załadowania strony i wypełni następujące zadanie:

1. pobierze z serwera <http://www.random.org/> losową liczbę z przedziału 0-60
2. poprosi użytkownika o potwierdzenie liczby z punktu 1.
3. jeśli użytkownik potwierdzi liczbę, wróci do punktu pierwszego po odczekaniu liczby sekund pobranej w punkcie 1. W przypadku braku potwierdzenia zakończy działanie.

Zadanie 4: budowanie układu

Za pomocą **HTML + IMG + CSS** (bez JS i frameworków CSS) stwórz układ jak w załączonej grafice (szary blok jest grafiką o podanych wymiarach). Następnie zrób aby ten układ był responsywny (RWD) adaptując wygląd desktopowy, na potrzeby mobile.

TOP TEXT

LONG BOTTOM TEXT LONG BOTTOM
TEXT LONG BOTTOM TEXT LONG
BOTTOM TEXT LONG BOTTOM TEXT
LONG BOTTOM TEXT LONG BOTTOM
TEXT LONG BOTTOM TEXT LONG
BOTTOM TEXT LONG BOTTOM TEXT



text lorem ipsum dolor text lorem ipsum dolor
text lorem ipsum dolor text lorem ipsum dolor
text lorem ipsum dolor text lorem ipsum dolor
text lorem ipsum dolor text lorem ipsum dolor
text lorem ipsum dolor text lorem ipsum dolor
długi

Zadanie 5: React

Poniższy kod zawiera błędy. Wskaż je (wypisz hasłowo) i popraw w kod tak by był prawidłowy.

```
class MyComponent extends React.component {  
  constructor() {  
    this.state = {
```

```

        counter: 0,
        items: [1,2]
    };
}

increase(this) {
    this.state.counter++;
}

addItem() {
    this.state.items = this.state.items.push(this.state.items.length+1);
}

render() {
    <div class="data">
        <div>{this.state.counter}</div>
        <ul>
            {this.state.items.map(item => (
                <li key={item}>element {item}</li>
            ))}
        </ul>
    </div>
    <div class="actions">
        <button type="button" onclick={this.increase}>zwiększ</button>
        <button type="button" onclick={this.addItem}>dodaj</button>
    </div>
}
}

```

```
ReactDOM.render(<MyComponent>, document.querySelector("body"));
```

Zadanie 6: JavaScript

Napisz kod HTML + JS, który będzie realizował poniższe wytyczne:

1. zawiera formularz
 - a. pole typu input (musi być walidowane czy zawiera liczbę całkowitą > 0)
 - b. przycisk do wysłania formularza
2. po załadowaniu strony
 - a. w polu typu input ustawia liczbę pobraną z URLa ze zmiennej **sec**
 - b. jeżeli takiej zmiennej nie ma, lub nie jest liczbą, to przyjmuje, że **sec=5**
 - c. na przycisku wyświetla tekst **X sek** gdzie **X** to liczba ze zmiennej **sec**
 - d. rozpoczyna odliczanie sekund, od liczby ze zmiennej **sec**, zmieniając przy tym aktualną liczbę sekund wyświetloną na przycisku

3. w momencie dojścia odliczania do 0
 - a. blokuje pole typu input przed zmianą wartości
 - b. blokuje formularz przed wysłaniem przez użytkownika strony
 - c. zmienia tekst na przycisku na **wysyłanie**
 - d. wywołuje wysłanie formularza z poziomu JS
4. jeżeli zanim czas dojdzie do 0, użytkownik sam wywoła wysłanie formularza postępujemy tak jakby czas dotarł do 0
5. wysyłanie formularza ma ponownie odpalić cały kod z przekazaniem do zmiennej **sec**, wartości z pola input