PROJEKT APLIKACJI

Generator liczb wraz z ich sortowaniem

CEL DOKUMENTU

Celem dokumentu jest przedstawienie pełnej specyfikacji wymagań funkcjonalnych i niefunkcjonalnych dla aplikacji webowej umożliwiającej:

- generowanie losowej tablicy liczb o podanym zakresie i rozmiarze
- sortowanie tablicy algorytmem Bubble Sort
- wyświetlanie wyników w części interfejsu użytkownika

Dokument określa, jakie funkcje ma spełniać system oraz jakie ograniczenia i założenia obowiązują podczas implementacji.

ZAKRES SYSTEMU

System stanowi aplikację kliencką (front-end) działającą w przeglądarce. Użytkownik może:

- wprowadzić parametry tablicy (min, max, liczba elementów)
- wygenerować tablicę z unikalnymi elementami
- ukryć obraz sortowania po utworzeniu tablicy
- posortować tablicę jednorazowo
- wyświetlić wyniki

System nie przechowuje danych oraz nie wymaga back-endu.

DEFINICJE

- Bubble Sort – algorytm sortowania polegający na wielokrotnym porównywaniu i zamianie sąsiednich elementów
- UI/Interfejs użytkownika – część widoczna w przeglądarce w obszarach .left i .right
- Parametry wejściowe – minimum, maksimum, liczba elementów tablicy

OPIS OGÓLNY

Perspektywa systemu

- Aplikacja działa w modelu samodzielnego modułu front-endowego. Interakcje odbywają się poprzez:
    1. formularz HTML
    2. przyciski sterujące: Utwórz tablicę, Sortuj, Reset

Funkcje użytkownika

- Użytkownik może:
    1. Wprowadzić dane wejściowe
    2. Kliknąć „Utwórz tablicę”, aby wygenerować unikalną tablicę
    3. Zobaczyć tablicę nieposortowaną
    4. Kliknąć „Sortuj”, aby posortować tablicę tylko raz
    5. Zresetować aplikację i zacząć od nowa

ZAŁOŻENIA I ZALEŻNOŚCI

- Zakres liczb musi umożliwiać wygenerowanie zadanej liczby unikalnych elementów
- Przeglądarka obsługuje nowoczesne standardy ECMAScript
- HTML zawiera elementy: inputy, przyciski oraz obszary .left, .right oraz obraz

WYMAGANIA FUNKCJONALNE

- **F1. Pobieranie danych wejściowych**

  System musi umożliwić wprowadzenie następujących wartości:

  1. minimum liczby (#minimum)
  2. maksimum liczby (#maximum)
  3. liczba elementów tablicy (#elements)

- **F2. Generowanie tablicy unikalnych liczb**

  System ma:

  1. akceptować zakres [pMin, pMax]
  2. sprawdzić, czy można wygenerować liczbę elementów bez powtórzeń
  3. wygenerować tablicę o podanej długości
  4. elementy muszą być losowe i unikalne
  5. zwrócić null, jeśli warunek unikalności nie może zostać spełniony

- **F3. Wyświetlanie tablicy nieposortowanej**

  Po wygenerowaniu tablicy:

  1. system wyświetla tablicę w .right
  2. ustawia flagę isSorted = true
  3. ukrywa obraz sortowania (funkcja hiddenImage())

- **F4. Sortowanie tablicy algorytmem Bubble Sort**

  System ma:

  1. sortować tablicę metodą Bubble Sort zgodnie z funkcją bubbleSort()
  2. wykonać sortowanie tylko raz
  3. wyświetlić tablicę posortowaną wyłącznie wtedy, gdy wcześniej wygenerowano tablicę i atrybut isSorted = true

- **F5. Zablokowanie wielokrotnego sortowania**

  System musi uniemożliwiać:

  1. dodanie drugiego wyniku sortowania
  2. wykonanie sortowania, jeśli tablica została już posortowana

Mechanizm:

1. isSorted = true (można sortować)
2. po wyświetleniu wyniku: isSorted = false

- **F6. Reset aplikacji**

  Przycisk reset:

  1. czyści dane wejściowe
  2. usuwa komunikaty z .right
  3. ustawia newArray = null
  4. przywraca widoczność obrazu sortowania

WYMAGANIA NIEFUNKCJONALNE

- **N1. Wydajność**

  1. Sortowanie musi działać dla tablic od 1 do 1000 elementów bez zauważalnego opóźnienia.
  2. Reakcje UI poniżej 100 ms

- **N2. Użyteczność**

  1. Interfejs musi być prosty i intuicyjny
  2. Komunikaty muszą być wyświetlane w sposób czytelny (element <h4>)

- **N3. Bezpieczeństwo**

  1. System nie przechowuje danych
  2. System nie komunikuje się z serwerem
  3. Brak konieczności ochrony danych osobowych

- **N4. Kompatybilność**

  1. Obsługa aktualnych wersji Chrome, Firefox, Edge
  2. Kod zgodny z ES6+

- **N5. Jakość kodu**

  1. Kod JS musi być modularny i czytelny
  2. Funkcje powinny być krótkie i robić jedną rzecz (zasada SRP)

WYMAGANIA INTERFEJSU

- **Interfejs użytkownika**

  Interfejs musi zawierać:

  1. trzy pola input
  2. trzy przyciski: Utwórz, Sortuj, Reset

3. sekcję .right na wyświetlane tablice
4. opcjonalną grafikę #imageSort

- **Interfejs systemowy**

  Brak komunikacji z serwerem

WYMAGANIA DOTYCZĄCE ALGORYTMU BUBBLE SORT

Opis działania algorytmu:

1. Dla kolejnych indeksów i od 1 do n−1
2. Dla kolejnych indeksów j od 0 do n−i−1
3. Jeżeli arr[j] > arr[j+1], zamień elementy

PRZYPADKI UŻYCIA (USE CASES)

- **UC1 – Utworzenie tablicy**

  Aktor: Użytkownik

  Przebieg:

  1. Użytkownik wprowadza minimum, maksimum i liczbę elementów
  2. Użytkownik klika „Utwórz”
  3. System generuje tablicę
  4. System wyświetla tablicę
  5. System ustawia isSorted = true

- **UC2 – Sortowanie tablicy**

  Warunek wstępny: isSorted = true

  Przebieg:

  1. Użytkownik klika „Sortuj”
  2. System sortuje tablicę
  3. System wyświetla wynik
  4. System ustawia isSorted = false

- **UC3 – Reset**

  1. Użytkownik klika „Reset”
  2. System czyści dane i widok

OGRANICZENIA

1. Sortowanie ograniczone do algorytmu Bubble Sort
2. Brak możliwości generowania tablicy z powtórzeniami
3. Wartości muszą być liczbami całkowitymi

KRYTERIA AKCEPTACJI

- **K1. Poprawne generowanie tablicy**

    1. System generuje dokładnie tyle elementów, ile podał użytkownik
    2. Elementy są unikalne

- **K2. Jednorazowe sortowanie**

    1. Druga próba sortowania nie nadpisuje wyników
    2. Sortowanie nie działa, jeśli isSorted = false

- **K3. Poprawność sortowania**

    1. Tablica jest posortowana rosnąco
    2. Brak błędów zamiany indeksów