Vježba 10. – CALLBACK, PROMISE, AWAIT

Današnja vježba odnosi se na posljednji dio sadržaja s predavanja i vježbi. Kako biste uspješno položili vježbu, potrebno je riješiti barem tri zadatka, ne računajući primjere. Preostali zadatak možete riješiti kod kuće i predati putem Moodle-a. Ako imate pitanja ili vam nešto nije jasno, slobodno se obratite putem e-maila ili dođite na nadoknade u sljedećem tjednu.

U sljedećem tjednu nema novih vježbi. U tjednu od 29.5. do 2.6. održat će se drugi kolokvij u terminu vježbi.

1. Napravite funkciju delayedGreeting, koja će koristiti Promise objekt za ispisivanje pozdrava nakon određenog vremenskog kašnjenja.
   Funkcija treba prihvatiti vrijeme kašnjenja (u milisekundama) i tekst pozdrava kao argumente.
   Nakon zadanog kašnjenja, funkcija treba ispisati tekst pozdrava.

2. Napravite funkciju checkWeather, koja će koristiti Promise objekt za provjeru vremenskih uvjeta na temelju unesenog grada.
   Funkcija treba prihvatiti ime grada kao argument. U ovom zadatku, koristit ćemo jednostavnu **simulaciju** dohvaćanja podataka o vremenskim uvjetima iz API-ja. Ako je vrijeme sunčano, Promise treba biti riješen (resolve) s porukom "Vrijeme je sunčano u {grad}". U suprotnom, Promise treba biti odbijen (reject) s porukom "Vrijeme nije sunčano u {grad}".

3. **Primjer** zadatka sa await funkcijom. Prepišite i pokrenite program.

```
function delay(ms) {
   return new Promise(resolve => setTimeout(resolve, ms));
}

async function example() {
  console.log('Prije čekanja');
  await delay(2000); // čeka 2 sekunde
  console.log('Nakon čekanja');
}

example();
```

U ovom primjeru, funkcija delay vraća obećanje (Promise) koje se rješava nakon određenog vremenskog razdoblja pomoću funkcije setTimeout.

U funkciji example, koristimo ključnu riječ await kako bismo zaustavili izvršavanje programa dok se obećanje ne riješi. Nakon čekanja od 2 sekunde, program nastavlja izvršavati liniju console.log('Nakon čekanja').

Ovo omogućava asinkrono izvršavanje funkcija i čekanje na rezultate prije nastavka izvršavanja ostatka koda.

Prepišite navedeni primjer i promijenite vrijeme čekanja.

Što se promijenilo? Jesu li se sve naredbe izvršile očekivanim redoslijedom?

4. **Primjer** zadatka sa callback funkcijom. Prepišite i pokrenite program.

```
function getData(callback) {
   setTimeout(function() {
     const data = { message: 'Ovo su podaci' };
     callback(data);
   }, 4000);
 }

 function processData(data) {
   console.log('Obrada podataka:', data.message);
 }

 getData(processData);
 console.log('Nastavak izvršavanja');
```

Callback funkcija je funkcija koja se predaje kao argument u drugu funkciju i izvršava se nakon završetka određene operacije ili događaja. Kada se callback funkcija pozove, ona omogućava prenošenje rezultata ili obavijesti nazad u izvornu funkciju. Glavna svrha callback funkcija je podržati asinkrono izvršavanje u JavaScriptu.

Iako je vrijeme kašnjenja postavljeno na 4 sekunde, kôd još uvijek izvršava console.log('Nastavak izvršavanja') prije nego što se callback funkcija pozove. To pokazuje da se callback funkcija izvršava asinkrono i ne čeka na završetak ostatka koda.

5. Napišite funkciju calculateAndPrintSum koja prima dva broja a i b. Funkcija treba koristiti await funkciju calculateSum kako bi asinkrono izračunala zbroj brojeva a i b. Nakon izračuna, ispišite rezultat zbroja. Funkcija calculateSum simulira izračun zbroja brojeva s određenim vremenskim kašnjenjem. Funkcija calculateSum vraća obećanje (Promise) koje se rješava (resolve) s rezultatom zbroja.

6. Napišite funkciju checkNumber koja prima broj kao argument. Funkcija treba provjeriti je li dani broj paran ili neparan te vratiti odgovarajuću poruku.
   Koristite obećanje (Promise) s callback funkcijama resolve i reject. Ako je broj paran, obećanje treba biti riješeno s porukom "Broj je paran.". Ako je broj neparan, obećanje treba biti odbijeno s porukom "Broj je neparan.".
   Pozovite funkciju checkNumber s proizvoljnim brojem i ispišite rezultat korištenjem metoda then za hvatanje riješenog obećanja i metode catch za hvatanje odbijenog obećanja.