Ponavljanje2. Osnove JS-a

1. Napisati funkciju koja vraca funkciju *after* i prima dva argumenta. Prvi argument je broj koji predstavlja koliko puta funkcija *after* treba biti pozvana prije nego se izvrsi callback funkcija, a drugi argument je callback funkcija koja ce se izvrsiti. (hint closure)

2. Napisati funkciju *movieSelector* koja prima niz objekata koji sadrze informacije o filmovima (id, title i score). Iskoristiti JS funkcije filter i map kako bi povrat funkcije *movieSelector* bio niz koji sadrzi samo imena filmova sa score-om vecim od pet i to uppercase-ane. Primjer za objekt:

```
movies = [ { id: 1, title: "Pan's Labyrinth", score: 9 },
        { id: 37, title: "Gentelman", score: 6 },
        { id: 11, title: "Batman", score: 5 },
        { id: 44, title: "Birds of Pray", score: 1 },];

        ocekivani rezultat je: [ "PAN'S LABYRINTH", "GENTELMAN" ]
```

3. Napisati funkciju numSelectString koja prima niz brojeva i vraca string. U ovoj funkciji treba iskoristiti JS funkcije filter, sort i reduce kako bi se kao povrat dobio string koji sadrzi samo neparne brojeve iz niza, odvojene zarezom koji su poredani uzlazno. Npr. za niz [17, 34, 3, 12, 23] povrat je string „3, 17, 23".

4. Napisati funkciju koja prima niz i u njoj kreirati for petlju koja iterira kroz elemente niza i vraća sumu elemenata.

5. Napisati funkciju koja kao argument prima niz elemenata, te vraća funkciju koja pri svakom pozivu vraća sljedeći član niza sa konkateniranim stringom „je n-ti element niza". (hint. Closure).

6. Kreirati funkciju russianRoulette koja prima jedan argument (broj, nazovimo ga n), i vraca funkciju. Vracena funkcija ne prima argumente, a vraca stirng „click!" prvih n – 1 poziva. Prilikom n-tog poziva, funkcija ce vratiti string „bang". Svakim sljedecim pozivom funkcije, povratna vrijednost ce biti string „reload to play again". (hint „closure")

7. Kreirati funkciju *blackJack* koja prima niz (brojevi od 1 do 11) i vraca funkciju (nazovimo je *dealer*). Funkcija *dealer* prima dva argumenta (dva broja) i vraca funkciju (nazovimo je *player*). Ako je funkcija *player* pozvana PRVI PUT, vraca sumu brojeva koji su argumenti funkcije *dealer*. Ako je funkcija *player* pozvana DRUGI PUT, vratiti ce sumu ranija dva argumenta i prvog clana niza, ukoliko je suma manja ili jednaka 21. Ako je veca od 21, funkcija *player* vraca string „bust!". Ukoliko prethodna suma nije veca od 21, tada ce se svakim sljedecim pozivom funkcije *player* vratiti nova suma - zbroj posljednje sume i sljedeceg broja u nizu koji je prosljedjen funkciji *blackJack*. Ukoliko je nova suma veca od 21, povrat funkcije player je „bust!". Ako je funkcija *player* vratila string „bust!", tada ce svaki sljedeci poziv funcije *player* vratiti string „You are done!". Ideja ovog zadatka je koristenje i shvacanje koncepta closure, te ga je obavezno korisiti.

8. Kreirati promise koji će se nakon 1000 ms resolve-ati u string „Resolved", koristeći metodu setTimeout(). Funkcija koja je zadužena za printanje resolved promisa, se prosljeđuje u promise preko then() metode.

(https://developer.mozilla.org/enUS/docs/Web/JavaScript/Reference/ Global_Objects/Promise)

15. Kreirati promise koji će se odmah reject-ati. Iz funkcije koja se pozvala kada je promise reject-an printati „Error“(hint. nad promise objektom pozvati metodu catch).

16. Promisi su asinhroni i sada ćemo to i dokazati. Kreirati promise koji će se resolve-ati sa vrijednošću „Promise has been resolved!“. Nakon toga ubaciti sljedeći kod:

```
promise.then(() => console.log('Promise has been resolved!'));
console.log("I'm not the promise!");
```

Koji redosljed ispisa očekujete? Zašto?

17. Niže je naveden objekt fakePeople koji imitira podatke izvučene iz baze podataka. Napisati funkciju fakeAPICall(i) koja vraća promise koji će se resolve-ati u podatke osobe na i-tom mjestu. Promise resolve-ati nakon random broja milisekundi između 1000 i 3000. Napisati funkciju getAllData, koja koristi Promise.all kako bi se 3 API poziva fakeAPICall(i) izvršila istovremeno. Argument i odabrati proizvoljno. U slučaju da je i veći od 3, reject-ati promise.
Promise.all
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise/all

```
const fakePeople = [
        { name: 'Ivo', hasPets: false},
        { name: 'Eva', hasPets: true},
        { name: 'Marko', hasPets: true} ]
```