Računarski praktikum 2 - drugi kolokvij, 23.06.2014.

Ime i prezime: _____ JMBAG: _____

Upute:

Kolokvij se piše na računalu, traje 2h te se sastoji od dva praktična zadatka. Svaki zadatak se mora spremiti u posebnu datoteku pod nazivom ImePrezimebroj_zadatka.php. Na kolokviju smijete koristiti W3Schools, materijale s vježbi i svoja rješenja iz zadaća. Upotreba bilo kakvog drugog materijala ili web stranice će se smatrati prepisivanjem i rezultirat će oduzimanjem kolokvija i padom kolegija. Sve materijale i kodove možete naći na:

https://github.com/ante003/materijali-RP2/

Kolokvij predajete spremanjem zip datoteke pod nazivom ImePrezime.zip, koja sadrži vaša rješenja, u Merlin.

Zadatak 1. [15 bodova]

a) [10 bodova] Svaki mobilni uređaj može slati i primati poruke. Ako se vratimo malo u prošlost kada nisu postojali pametni telefoni, tipkovnice su sadržavale brojeve od 0-9 gdje je nula označavala razmak, a svako slovo se dobilo određenim ponavljanjem pritiska neke tipke. Ako bismo htjeli napisati dva puta isto slovo koje se nalazi na jednom broju, potrebno je pričekati neko vrijeme da se može opet pritisnuti tipka. U našem zadatku, čekanje će biti reprezentirano sa znakom razmaka.

Primjer: a = 2, b = 22, aa = 2 2, bb = 22 22, rp = 777 7, na sta va = 662077778208882

Napravite klasu *Message* koja sadrži metode *receiveMessage()* i sendMessage().
Metoda *sendMessage()* prima string kao parametar, te vraća numerički zapis tog stringa.
Metoda *receiveMessage()* prihvaća numerički zapis nekog stringa i vraća konvertirani string iz danog numeričkog zapisa.

Napomena: možete biti sigurni da će te primati i slati samo riječi.

b) [5 bodova] Napravite klasu *Crypter* koja će kriptirati vaše poruke pomoću Cezarove šifre. Kao alfabet uzmite englesku abecedu. Klasa mora sadržavati metode *encrypt* i *decrypt,* tj. metode za kriptiranje i dekriptiranje poruke. Klasa *Crypter* nasljeduje klasu *Message* i reimplementira (overridea) metode *sendMessage() i receiveMessage()* tako da prilikom poziva tih metoda, poruka se mora kriptirati, odnosno dekriptirati. Za primanje poruke napravite formu u koju će se moći upisivati numerički prikaz poruke, a za primanje poruke, neka se poruka ispiše na zaslon. Parametar "shiftanja" k se uvijek sprema na početak numeričkog prikaza poruke, ili zadaje u formi prilikom slanja poruke u normalnom obliku.

Cezarova sifra je najjednostavnija metoda za kriptiranje u kriptografiji. Temelji se na principu pomaka (shiftanja) abecede za parametar k, k > 0. Slova u početnoj riječi se zamjenjuju sa slovima na istim mjestima u abecedi koja je pomaknuta za k mjesta u desno.

A B C D E F …. I pomaknimo sada abecedu za k = 2.
Y Z A B C D …. dobije se pomaknuta abeceda pomoću koje šifrirate poruku.

Primjer:

DEDA → BCBY za k = 2.

Ako niste rješili a) zadatak, b) zadatak tada nosi najviše 2.5 bodova te mora kriptirati/dekriptirati obične poruke.

Zadatak 2. [5 bodova]

Napravite klasu *ScalarProduct* koja sadrži metodu *minScalarProduct*. Metoda prima dva

niza (vektora) te traži permutaciju ta dvaju vektora koja će imati najmanji skalarni produkt koji

će metoda vratiti kao parametar.
Metoda je statička te se poziva prilikom svakog unosa neke vrijednosti vektora u formu putem Ajaxa. Ako vektori ne zadovoljavaju dane uvijete za dobar izračun, neka se obavijesti korisnika putem poruke (ako su vektori različite duljine, sadrže slova,...).

Primjer:

v 1 = [1, 3, −5]

v 2 = [−2, 4, 1]

Najmanji skalarni produkt od v1 i v2 je: −25

Svaki primjer koji se dobro izračuna spremite u tekstualnu datoteku i neka na stranici postoji tablica u koju se svi primjeri ispisuju u sljedećem formatu:

Vektor1 | Vektor2 | Najmanji skalarni produkt

Za ispis u tablicu se brine Ajax koji provjerava svakih dvije sekunde da li je unešen neki novi podatak. Način zapisivanja u tekstualnu datoteku i provjere da li postoje novi zapisi smislite sami.

Zadatak 3. [5 bodova]

Pomoću PHPa i Ajaxa napravite jednostavnu vremensku prognozu (weather report). Odaberite proizovljan API koji poslužuje podatke o temperaturama, te ispišite trenutnu temperaturu u gradu koji odaberete.

PHP skripta mora sadržavati sljedeće funkcionalnosti:
- sadrži predefinirane gradove za koje želite provjeravati temperaturu. (neka ih bude barem pet)
- implementirajte Auto Completion koji predlaže gradove s obzirom na korisnikov unos. Za mjeru sličnosti možete provjeravati da li stringovi počinju na isti način, ili implementirati neku pametniju mjeru (npr. Levenshteinovu mjeru).
- spajanje na odabrani API. Npr, pogledajte meteo.hr.

Napravite posebnu klasu u koja će se brinuti o spremanju rezultata. Na njoj implementirajte magičnu metodu koja će se automatski pozivati kod ispisa varijable koja je tog tipa, te još dvije magične metode po vlastitom odabiru. Implementacija magičnih metoda po izboru mora imati smisla. Ako je implementacija besmislena (npr. echo "ovo je implementirana magicna metoda";), smatrat će se neimplementiranom.
Npr.

```
$prognoza = new Vrijeme();
echo $prognoza; // tu se poziva magična metoda od klase Vrijeme.
```

Drugi dio je izgled Forme u koju ćete upisivati upit. Dizajn je nebitan, osim sljedećeg dijela:
- tekst se upisuje na svoje predviđeno mjesto (nemojte koristiti textarea tag).
- kada se poziva Auto Complete preko Ajaxa, ispod dijela za upis teksta se mora pojaviti lista predloženih upisa.(kao kad tražite pojam preko Googla, i onda se ispod pokazuju prijedlozi). Lista prijedloga mora biti iste širine kao i dio za unos teksta.
- nakon što upišete pravilnu vrijednost, tada PHP skripta vraća tražene vrijednosti koje se ispisuju u HTML dokumentu.

U nekom trenutku rješavanja zadatka ćete možda trebati pročitati XML ili JSON, pa si za to pogledajte već postojeća integrirana rješenja. Npr. za XML postoji određena klasa koja služi za čitanje, pisanje i generiranje XMLa.