Бележки

# HTML

## Problems\_1

1. Friends.html - Има Html елемент, който реди контента по подобен начин (на indent-a). Виж dl, dt, dd.
2. Problem\_1.html – Избягвай да използваш тагове като <p> като контейнери. Само защото искаш отстъп или по други причини. Семантично не е коректно. Параграфа по възможност трябва да съдържа просто текст .В случая заглавието и списъците трябва да са в друг контенер – такива много – div, span, (article, header, aside, и тн. от новите /не че са подходящи в случая ☺ но принципно)

## Problems\_2

1. Problem\_1.html – Всичко, ВСИЧКО, в една таблица е клетка. Няма как (правилно) да имаш съдържание в една таблица без да е в клетка. Независимо дали си в thead, tbody, tfoot или другаде, ако е съдържание то се намира в клетка, която се намира в ред. Винаги.  
   Избягвай използвнаето на <center>. Стар таг, отново отговорност на css-a.

## Problems\_3

1. Problem\_1.html – елементите ти грозно излизат от клетките си. Причината за това е box model-a. Прочети го пак и виж защо се получава. За решение може да ти помогне метод на css-a. Пример: width: **calc(50% - 20px)**
2. Problem\_2.html – Защо таблицата е във <form>?
3. Problem\_3.html – Избягвай вкарването на талбица в таблица в таблица в таблица ... Рядко има реална необходимост от такова нещо, не е семантично коректно в повечето ситуации и винаги е трудно за четене ☺ Тук си използвал таблици за layout – което е грешка. Ползвай обикновенни контейнери като div и span. LAYOUT-a e отговорност на CSS-a! Ако искам да променя начина по който се подреждат елементите, групирания и приравнявания, няма логика да трябва да си променям съдържанието, а само оформлението.

# CSS

## Problems\_1

* Много, много мънички забележчици, виж файловете. Иначе много добре.

## Problems\_2

1. Problem\_1.css – Хубаво си се сетил за first-letter селектора. Обаче избрания шрифт не е стандартен и браузерите не го поддържат по подразбиране. За целта се издърпва файла на шрифта и се добавя като шрифт в css-a. Виж файла – сложил съм го.
2. Problem\_2.css – вълшебно е ☺ супер изглежда, 99.9999999999999% еднакво е, браво. Ама е много код, не ми се чете ☺
3. Problem\_3.html – Мдам, точно така, просто тяхното е правено с много по-малко дивове.
4. Problem\_4.css – Много си се постарал, браво. Това прилича срашно много на калкулатора. Само не забравяй да оправяш колко да е широк общо елемента. (целия калкулатор).   
   Прекалено много таблици са използвани и даже си се объркал с таговете им.   
   Има някой дребни недостатъци, но те са в това какви стилове подбираш да сложиш на елементите. Т.е всичко което си искал да постигнеш технически според мен си го постигнал, остава въпроса на предпочитания и на оценка на стиловете за което си трябва просто повече опит (като някой който много я мрази тази част и не би казал че му се отдава.. довери ми се като ти казвам, че CSS-a е мистично същество, което прави каквото си поиска по-често от колкото би ти се вярвало)

## Problems\_3

1. Problem\_1.html – нееее, това си се прави с <ul> и <li>-та. Защо дивове с класове?  
   Отделно – не работи. Трябваше да пробваш с още нива навътре подсписъци. Това е дървовидна структура, и като такава може да расте на вътре на теория – безкрай. И като такава, трябва стиловете да важат и на корените навътре. Т.е когато имам 3то, 4то, 5то ... нива, всяко от тях да се отваря само когато hover-на родителския node.
2. Problem\_2 – Тука си се изложил. Липсват ти рамки, цветове, разлики в стиловете и тн.   
   Задавай width на главния родител, не го прави на цял екран!   
   В такива случаи се замисляй какъв елемент от website са тези неща, и като такива кое ще се генерира динамично и кое ще е статично. Т.е в случая това е списък с категории, като динамичното в тях ще са картинките и текста. Като картинки, там голямо значение няма, стига да си пазят ширината и дължината, което си е част от css-a. Но при текста там ще има разлика с оглед обема от текст който ще идва. Трябва в такива случаи елементите да са стилизирани така, че ако текста е много малко, пак да изглеждат като останалите категории. А когато текста е много, леко да се разтягат за да го събират, но не променяйки други части от вида – да не се местят бутоните или подобни.
3. Problem\_3 – Бива. НО, когато имаш елемент, който е сложен или очевидно правен със ефекти, които не са осъществими чрез CSS => картинка, не забравяй че неможе да нарушаваш семантичното значение, че и ФУНКЦИОНАЛНОСТТА на елемента. Т.е неможе Go бутона да е просто картинка. Как ще си събмитне формата потребителят? Не е коректно да и закачиш евенти през javascript и да го превърнеш в бутон. Бутона трябва да е бутон, или поне линк! <a> елементът често се използва при бутони със сложни стилове или картинки (защото input и button са малко неудобни за стилизиране) и е приемлив. НО, най-добре е да се стилизират input-a/и или button. Възможно е.
4. Problem\_5.css – По-добре е като не се ползват таблици за layout. Но може би си имал трудности с докарването на стиловете? Обърни повече внимание на display / position / float. Едно от сравнително сложните неща без таблица е да центрираш нещо в средата по вертикала, и то да е динамично. Т.е да не е изчислено на пиксели, и ако мръднеш размера на екрана да е в средата ама друг път.   
   <http://stackoverflow.com/questions/8865458/how-to-vertically-center-text-with-css> - отговорът тук е доста изчерпателен. Обърни повече внимание на Simulating table display.

# Javascript

## Problems\_2

1. Problems\_1.js – моя съвет е ВИНАГИ да си пишеш скобите на For/If и всички останали, които имат {} в конструкцията си, дори и когато имаш само една операция.

## Problems\_6

1. Problem\_1.js – поздравления за решението, но задачата е грешно описана. Говорим си за web, така че е неправилно зададено условието. Записал съм какво може би се иска в файла, както и примерно решение.   
     
   Понеже решаваш задачите от Телерик, в някоя от лекциите си казват, че вместо конзола имат предвид HTML полета, и да се взима input-а с функция.   
   Но нищо де, входа не е акцента, пък останалото ти изглежда наред.

## Problems\_7

1. Problems\_3.js – javascript също има стойностни и референтни типове. Всички обекти се предават по референция. Затова може да се нуждаем от копия на обектите. Обикновенно има 2 типа копия – shallow clone и deep clone. Във файла на задачата написах няколко примера. Разгледай ги. Не са перфектни, но за прости цели работят. Jquery също има методи, които връщат shallow или deep clone: <http://stackoverflow.com/questions/122102/what-is-the-most-efficient-way-to-deep-clone-an-object-in-javascript>

Много добре изглеждат задачите. Чесно казано не проверявах логиката особенно много защото ти си много добре с алгоритмите.

# jQuery

## task2

Много добре, харесва ми. jQuery има доста методи, които са по мощни и удобни за ползаване от javascript-ските. Т.е неща като създаване, намиране и обхождане на елементи става много по-лесно.

Апи-то, което си избрал е интересно, със сигурност изпълнява всички точки на условието. Малко ми е нестандартно да пиша операции с магически стрингове. По-добър подход може би е да направиш dropdown обект, който пази нужната информация и предоставя съответните методи? (има различни усложнения от този подход, просто споменавам за разгрявка на мисълта ☺).   
Помисли си и как би имплементирал възможността да се закачам към change event-a – т.е като смениш избраната стойност да ми хвърля евент, в който разбира се я виждам.

# Overall

За целите на упражнението е нормално, но в реални условия се пише CSS – inline или в <style> таг в документа, само в много, много редки случаи, когато наистина няма нужда от зареждането на отделен файл. 99.99% от времето css се пише в отделен файл. Това е защото така по-бързо се рендира, понеже css-a се интерпретира от видео-то, не от процесора, а HTML-a съответно от процесора. Затова ако е в отделен файл е много по-чист кода и по-добре работи.  
\* Въпреки това се случва да има css в style таг или инлайн. Затова бъди добре запознат с приоритета на изпълнение. Кой css ще важи ако смениш на един и същи елемент едно и също свойство? Същия въпрос съществува и дори когато работиш в един файл. Ако има различни селектори, които адресират един и същ елемент, не е винаги до хронологичния ред. В css има приоритети – опитай се да ги научиш, защото ще ти създават проблеми. Пример: селекторът div ще е с по-малък приоритет от .mydiv и тн.

Избягвай използването на <br>. Това е стар и смотан таг. Неща като това да отиде на следващия ред, или да има отстъп от горното съдържание са отговорност на css-a.

Моя съвет е винаги да пишеш свойствата с кавички. <ul type=”1”> вместо <ul type=1>. По четимо е по този начин.Това си е изцяло по твое предпочитание, направили са го да го поддържа и със кавички и без, но моя съвет е слагай ги, а и каквото и да избереш бъди консистентен – като няма да ги пишеш не ги пиши никъде ☺

Може би вече си почнал да забелязваш, че таблиците са доста удобни за реализирането на layout. Това е много известна грешка! Не ползвай таблици за layout. Има много статии по темата харесай си няколко и ги прочети. Таблиците не се ползват за layout. Отново – това е CSS отговорност.

За целите на упражнението CSS-a e доста добре, направо чудесен. В реални условия обаче, един дизайнер ще държи на 100%, особенно на неща като шрифта (font-family) и приравнявания/отстъпи да са същите. Разбира се, в такива реални условия дизайнера ще предаде тази информация ☺ т.е ще ти каже кои са използваните шрифтове, няма да се налага да научкваш. Но в заключение, ще се изисква резултата да е абсолютно идентичен с дизайна.