ერთად გავხდეთ Front-End დეველოპერი
ჩვენ ამას შევძლებთ!

ამ სფეროში მეც ისეთივე დამწყები ვარ,როგორიც შენ და თუ გგონია ბევრად მეტი ვიცი,ცდები.ახლა იმასაც გასწავლი,რაც მე უკვე ვისწავლე და ასე ერთად, ნაბიჯ-ნაბიჯ გავიზრდებით კარიერულად!

ალბათ ყველა დეველოპერი დამეთანხმება რომ დაუვიწყარი გრძნობაა პირველი კოდის დაწერა და მისი ბრაუზერში დანახვა.ამის გაკეთებას ცოტა ხანში თავადაც შეძლებ და დარწმუნდები რომ მართლაც ასეა.

სანამ უშუალოდ კოდის წერაზე გადავალთ, ჯერ ის გავარკვიოთ, ვინ უნდა გამოვიდეთ, რა გვევალება როგორც Front-End დეველოპერს და ჩვენი მოვალეობების შესრულებაში რა დაგვეხმარება.ამ ყველაფრის უკეთ გასააზრებლად კი საჭიროა ცოტა უფრო ღრმად ვიცოდეთ ვებ-გვერდების მუშაობის პრინციპი. ვებ-გვერდის შექმნაზე მუშაობენ Front-End დეველოპერი და Back-End დეველოპერი ან "სუპერ-დეველოპერი", აკადემიურ ენაზე Full-Stack დეველოპერი,რომელიც ორივე პროფესიისთვის საჭირო უნარებსა და ტექნოლოგიებს ფლობს. (მომავალი) Front-End დეველოპერები ვქნით საიტის ხილულ, ხელშესახებ მხარეს, თუმცა ნუ იფიქრებთ რომ ჩვენი მოვალეობა მხოლოდ ვიზუალურად გამართული ვებ-გვერდის შექმნაა და მეტი არაფერი,Front-End დეველოპერების ორი მთავარი მოვალეობაა : **მომხმარებლისგან ინფორმაციის შეგროვება (Collect Data)** (ასეთი ინფორმაციაა მაგალითად youtube-ში მოძიებული ნებისმიერი სიტყვა და წინადადება ან საიტზე რეგისტრაციის ფორმის შევსებისას შეყვანილი ჩვენი პირადი ინფორმაცია და ა.შ), რომელსაც გადასცემს Back-End დეველოპერს და მისგან დაბრუნებული პასუხის ანუ **რეზულტატის ჩვენება (Display Results)** (ეკრანზე გამოგვიტანა ჩვენ მიერ "და**search**ილი" სიტყვის ან წინადადების შესაბამისი ვიდეოები, გავიგეთ რომ წარმატებით გავიარეთ რეგისტრაცია და ა.შ). ყველანაირი ინფორმაცია რომელიც ეხება Front-End-ის მხარეს,ინახება ჩვენს კომპიუტერებსა თუ ლეპტოპებში, ხოლო Back-End დეველოპერის შესრულებული სამუშაო და მთელი ინფორმაცია,რაზეც Back-End დეველოპერები არიან პასუხისმგებელნი,ინახება სერვერებზე ანუ იგივე "ძლიერ" კომპიუტერთა ერთობლიობებზე ,რომლებიც განკუთვნილია იმისათვის რომ ჩვენ მიერ შექმნილი საიტი მუდმივად ხელმისაწვდომი და ფუნქციონირებადი იყოს მომხმარებლისთვის.სწორედ სერვერებზე განთავსებულ ინფორმაციას აბრუნებს რეზულტატის სახით Back-End დეველოპერი,როდესაც ჩვენ,მომხმარებლები რაიმეს ვეძებთ,სადმე ვრეგისტრირდებით თუ საბანკო ოპერაციებს ვასრულებთ. სანამ არც თუ ისე ბევრი ვებ-გვერდი არსებობდა,Front-End და Back-End დეველოპერის საქმიანობას ერთი ადამიანი ითავსებდა და მთელი ინფორმაციაც ინახებოდა მხოლოდ სერვერებზე,თუმცა ახლა უკვე მილიარდზე მეტი ვებ-გვერდი ფუნქციონირებს და ლოგიკურია რომ დაიზოგოს თანხაც და ადგილიც და ინფორმაციაც და საქმიანობაც გადანაწილდეს ისე,როგორც ზემოთ აღვნიშნე.

ახლა უკვე ვიცით რა გვევალება როგორც Front-End დეველოპერებს და დროა გავიგოთ,რომელი ტექნოლოგიებია საჭირო ჩვენი მოვალეობების შესასრულებლად.მართალია ტექნოლოგიების რაოდენობა დიდია,თუმცა ეს ნუ შეგაშინებს,რადგან ახლა მხოლოდ **3** მთავარ ტექნოლოგიაზე მოგიყვები ძალიან მოკლედ და ერთ-ერთზე ცოტას ვივარჯიშებთ კიდეც.
იმისათვის რომ ჩვენი ვებ-გვერდი ვებ-გვერდობდეს ვიზუალურ ნაწილში დაგვეხმარება **HTML** (HyperText Markup Language) და **CSS** (Cascading Style Sheets).
HTML და **CSS** არ არის პროგრამირების ენები,ისინი არის ე.წ მარკირებისა და სტილის ენები. **HTML**-ის დახმარებით ვქმნით ჩვენი საიტის "ჩონჩს",ხოლო **CSS** გვეხმარება რომ ეს "ჩონჩხი" შეიმოსოს. მხოლოდ ამ ორი ტექნოლოგიითაც შეგვიძლია ლამაზი ვებ-გვერდის ვიზუალის შექმნა,თუმცა მსგავსი ვებ-გვერდი იქნება "უსიცოცხლო" ანუ მასზე ვერანაირად ვიმოქმედებთ.

იმისათვის რომ ვებ-გვერდი "გავაცოცხლოთ" დაგვჭირდება პროგრამული ენის, **JavaScript**-ის გამოყენება.

სანამ საბოლოო მიზნამდე ანუ ჩვენი პირველი “ვებ-გვერდის” შექმნამდე მივალთ,მოდით მოვიქცეთ ისე,როგორც პროფესიონალი მოიქცეოდა და ვისწავლოთ ცოტა თეორია, რამდენიმე მარტივი წესი,რაც მომავალ გაურკვევლობებსა და შეცდომებს აგვარიდებს და თავიდანვე გააზრებული და სინტაქსურად გამართული კოდის წერას შეგვაჩვევს.
ჩვენ დღეს შევეხებით **HTML** (HyperText Markup Language)-ის საწყისებსა და ცოტაოდენ თეორიას.არაფერი რთული, მოდი ერთად ვნახოთ რამდენიმე მარტივი წესი და მაგალითი და გადავიდეთ დღევანდელი დღის მთავარ საქმეზე.

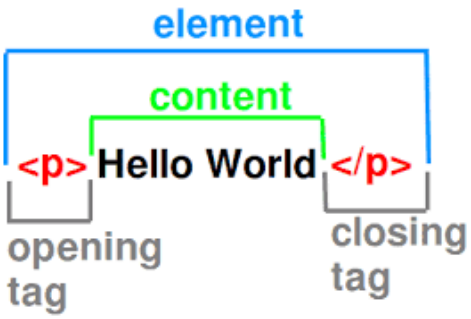
იმისათვის რომ ბრაუზერმა ჩვენ მიერ დაწერილი ტექსტი, დაკოპირებული ლინკი თუ ფოტო სწორად აღიქვას და ჩვენც ვიცოდეთ ვებ-გვერდის შემადგენელი ნაწილების სტრუქტურა და შინაარსი,გვჭირდება გამოვიყენოთ **HTML**-ის ტეგები,რომელსაც ჩვენ არ ვიგონებთ ან ვქმნით.ისინი უკვე არსებობენ, განსაზღვრული აქვთ თავიანთი სახელი, ფუნქცია და ჩვენ ისღა დაგვრჩენია დანიშნულებისამებრ გამოვიყენოთ.
მოდი რამდენიმე მარტივ ტეგს გავეცნოთ (აქედან უმეტესობასთან მომავალში შეხება აღარ გვექნება,რადგან რისთვისაც ამ ტეგებს იყენებდნენ,ახლა CSS-ით კეთდება). ტეგები შედგება ძირითადად ორი ნაწილისგან:გამხსნელი ტეგისგან,რომელიც შეიცავს ნაკლებობა-მეტობას შორის ჩაწერილი ტეგის სახელს და დამხურავი ტეგისგან,რომელსაც ნაკლებობის მერე ემატება /- დახრილი ხაზიც და ასევე არსებობს გამონაკლისი ტეგებიც,რომლებსაც არ აქვთ დამხურავი ტეგები და იწერებიან ან მხოლოდ ნაკლებობა-მეტობას შორის ან ტეგის სახელის შემდეგ ემატება /-დახრილი ხაზიც :

 ეს ტეგი ტექსტს ამუქებს
ეს ტეგი ტექსტს ხრის
<u><u>ეს ტეგი ტექსტს ხაზს უსვამს</u></u>
<a>ამ ტეგში თავსდება <u>ლინკი</u>
ამ ტეგში იწერება პარაგრაფი
<hr> ან <hr/>(ორივე სწორია) - ეს ტეგი ჰორიზონტალურ ხაზს ავლებს

 ან
(ორივე სწორია) - ეს ტეგი კი გამოიყენება მაშინ,როცა გვინდა რომ ტექსტი ახალ ხაზზე ჩამოვიდეს.

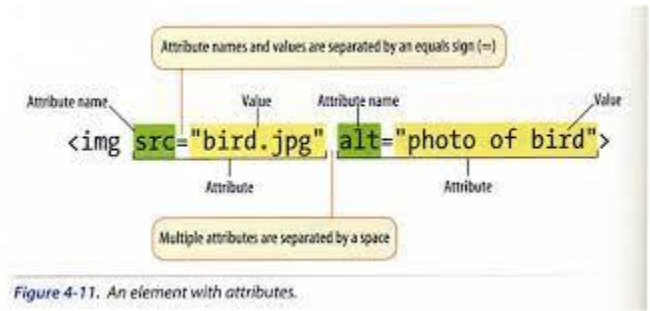
აქვე გეტყვი HTML-ის კოდის წერისას წარმოქმნილ ორ თავისებურებაზე,რამაც არ უნდა დაგაბნიოს.როდესაც ჩვენ გვინდა ბევრი space-ის ანუ გამოტოვების დაწერა სიტყვებს ან წინადადებს შორის და ამას კოდში გავაკეთებთ,ბრაუზერში ის არ აისახება და გამოჩნდება როგორც მხოლოდ ერთი დაშორება,რა თქმა უნდა შესაძლებელია ბევრი დაშორების გაკეთება ისე,რომ ეს ვებ-გვერდზეც გამოჩნდეს,მაგრამ ამაზე მოგვიანებით.მეორე თავისებურება კი ისაა,რომ კოდში წერისას enter-ით ერთმანეთის ქვეშ განაწილებული სიტყვები,ვებ-გვერდზე ისე გამოჩნდება როგორც ერთ ხაზზე დაწერილი და არა ისე როგორც კოდშია.რაღა თქმა უნდა ესეც შესაძლებელია რომ ნებისმიერი სიტყვა თუ ტექსტი ჩვენი სურვილისამებრ ახალი ხაზიდან დაიწყოს,თუმცა ამაზეც მოგვიანებით.

ჩვენი ახლადგაცნობილი ტეგები და მათ შორის მოქცეული ტექსტები ანუ შიგთავსი,ერთობლიობაში ქმნიან **HTML ელემენტს** (ეს ტერმინი მნიშვნელოვანია და დაიმახსოვრე).

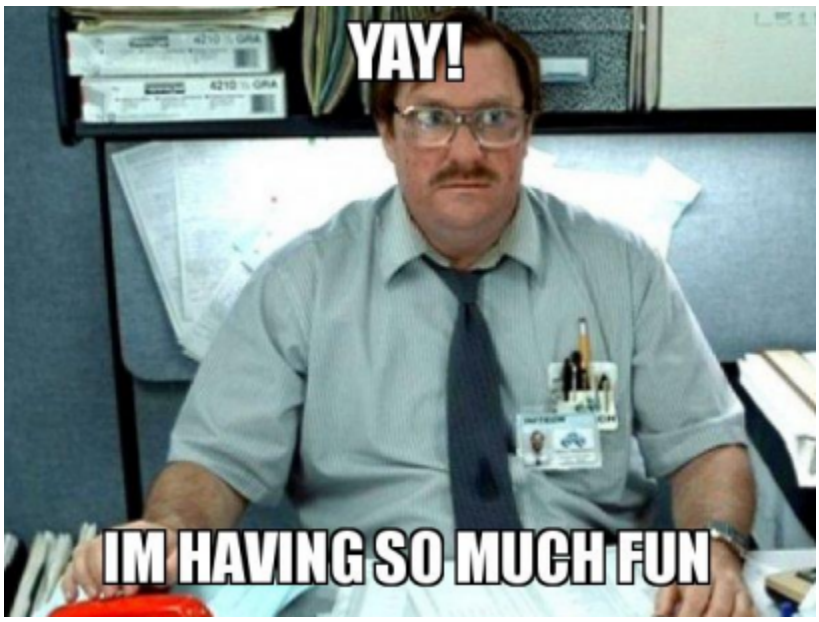


გვაქვს გამონაკლისები ანუ ე.წ Void ელემენტები და Void ელემენტები ეწოდება ისეთ ელემენტებს,რომლებსაც არ აქვთ შიგთავსი ანუ content,შესაბამისად მათი ორ ტეგს შორის მოთავსება არ არის აუცილებელი და ლოგიკური,აქედან გამომდინარე ასეთი ელემენტის ტეგებს არ აქვთ დამხურავი ტეგები და ჩვენ უკვე გავეცანით ორ მათგანს : <hr> ან <hr/> და
 ან
-ს.

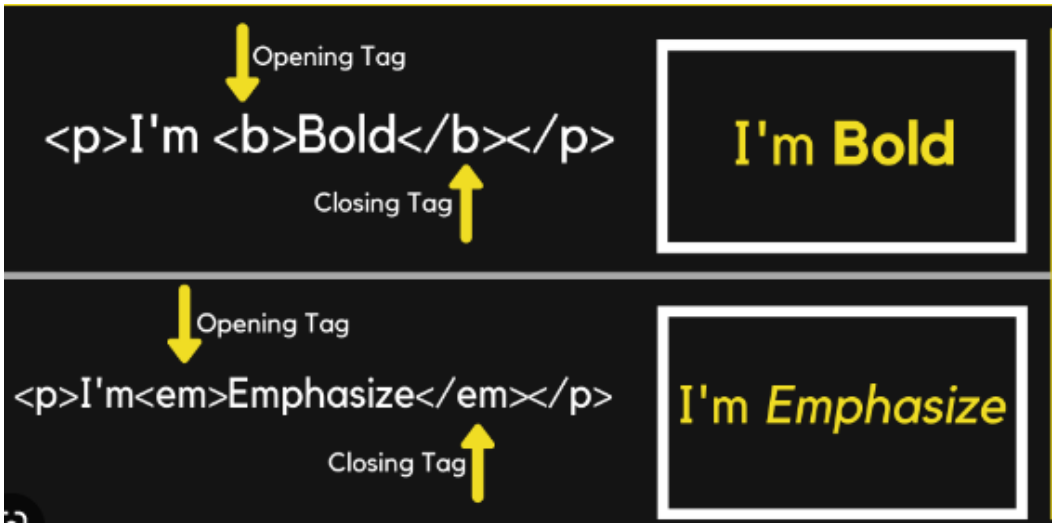
HTML ელემენტს შიგთავსის ანუ კონტენტის გარდა მოგვიწევს რომ გადავცეთ ბევრი სხვა ინფორმაციაც,ამაში კი დაგვეხმარება ატრიბუტი(Attribute). კონკრეტულ ტეგებს აქვთ კონკრეტული ატრიბუტები (შესაძლოა რამდენიმეც), მათი სახელები,შინაარსიდან გამომდინარე სხვადასხვაა,თუმცა ჩაწერის და წაკითხვის პრინციპი ერთია,რაც მკაცრად უნდა დავიცვათ და რაც შემდეგში მდგომარეობს: ატრიბუტის სახელი იწერება გამხსნელი ტეგის შიგნით, გამხსნელი ტეგის სახელის და space-ის შემდეგ, მას მოსდევს ტოლობა = და “ “ ან ‘ ‘- ში ჩასმული ის ინფორმაცია,რასაც ვაწვდით,იქნება ეს ლინკი,ფოტოს მისამართი თუ სხვა. (გამოიყენეთ ან მხოლოდ “ “-ები ან ‘ ‘ , რადგან კოდი იყოს უფრო კარგად წაკითხვადი და არადამაბნეველი სხვა ადამიანისთვის,ვისაც მასზე შენთან ერთად მოუწევს მუშაობა).როდესაც ერთდროულად ორ ატრიბუტს ვიყენებთ ან მეტს,პრინციპს არ ვარღვევთ და მათ შორის აუცილებლად ვიყენებთ space-ს.



ეს ყველაფერი მუშაობს ერთი მნიშვნელოვანი ჩაწერის პრინციპით,რომლის ცოდნაც აუცილებლად გამოგვადგება მომავალში,რადგან ეს პრინციპი თითქმის ყველა პროგრამირების ენაში გამოიყენება და ეს არის **Key=Value**-ს ჩაწერის პრინციპი, სადაც key აღნიშნავს ინფორმაციის სახელწოდებას,ხოლო value თვითონ იმ ინფორმაციას,რასაც ვინახავთ.თუ დავაკვირდებით ატრიბუტის სახელი გამოდის რომ key-ა,ხოლო,რასაც მასში ვინახავთ value.

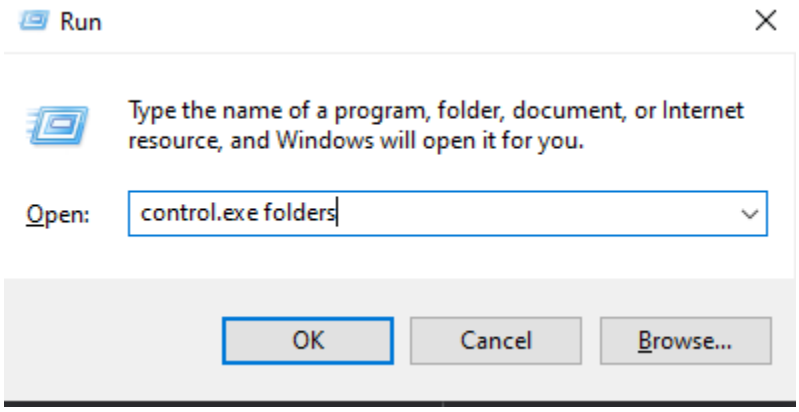


მხოლოდ ერთი თეორიული საკითხითღა მოგაწყენ თავს (თუმცა არამგონია მოგეწყინა),რაც ასევე ძალიან მნიშვნელოვანია და ამ საკითხს HTML ელემენტის nesting ეწოდება. nesting გვასწავლის ერთი html ელემენტის მეორეში სწორად განთავსების პრინციპს,რაც აგებულია ძალიან მარტივ ლოგიკაზე.არსებობს მშობელი ტეგი,შვილობილი ტეგი და ასევე თანაბარი ტეგებიც,ანუ ე.წ დედმამიშვილები 😂.არ უნდა დავივიწყოთ რომ ისინი ერთი ოჯახი არიან და არ მოვაქციოთ მშობელს შიგნით არსებული შვილის დამხურავი ტეგი,მშობლის დამხურავ ტეგს გარეთ ანუ არ დავარღვიოთ პრინციპი,რომ პირველად ვხურავთ იმ ტეგს,რომელიც ყველაზე ბოლოს გავხსენით.ვიცი,რომ ახლა ცოტა დაგაბნიე,თუმცა როგორც კი მაგალითებს შეხედავ,ყველაფერს მარტივად გაიგებ და შენს პირველ მარტივ ვებ-გვერდსაც შექმნი!

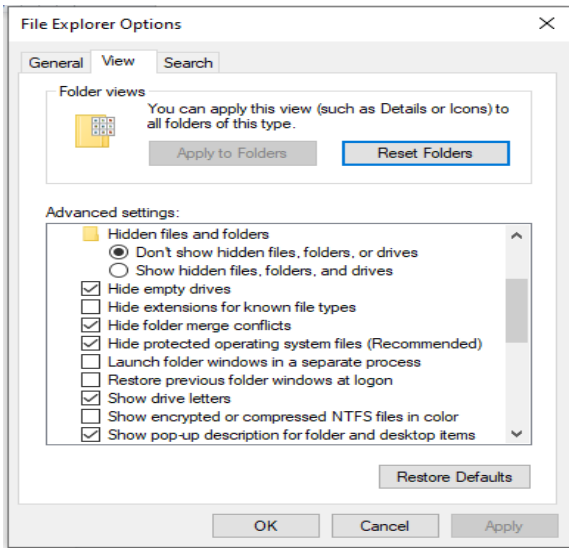


გილოცავ, შენ უკვე საკმარისი ცოდნა დააგროვე იმისათვის,რომ განიცადო დიდი ბედნიერება,რომელსაც საკუთარი ხელით დაწერილი მარტივი ვებ-გვერდის ხილვა მოგანიჭებს!

შექმენი ახალი Text Document შენს desktop-ზე და ჩაწერე სასურველი ტექსტი.მაგალითად: **მე შესანიშნავი Front-End დეველოპერი გავხდები კურსის ბოლოს**. შეინახე Ctrl s-ის დახმარებით.
სანამ საბოლოო შედეგით და შენი პროგრესით დატკბები ერთი პატარა რამეც ვისწავლოთ,თუ რა არის ფაილის გაფართოვება და როგორ იყოს ჩვენთვის ხილვადი. ფაილის გაფართოვების ხილვადობა მომავალში აუცილებლად დაგჭირდება და გამოგადგება.ფაილის გაფართოებები გამოიყენება ოპერაციული სისტემის მიერ იმის დასადგენად, თუ ფაილის რომელ ტიპებს უკავშირდება - სხვა სიტყვებით რომ ვთქვათ, რომელი აპი იხსნება ფაილზე ორჯერ დაწკაპუნებისას.
იმისათვის რომ ფაილის გაფართოვებას მუდამ ხედავდე გააკეთე შემდეგი რამ : დააჭირე Windows+R-ს,რაც ამოგდებს შემდეგ ფანჯარას,სადაც ჩაწერ control.exe folders-ს,როგორც მე და დააჭერ Ok-ს.

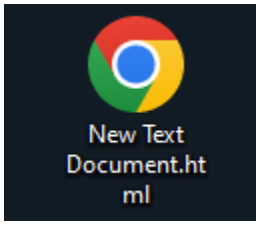


რაც გადაგიყვანს შემდეგ ფანჯარაზე,სადაც დააწვები View-ს და Hide extensions for known file types-ს მოუხსნი მონიშვნას,როგორც ჩემთანაა და ისევ OK-ს.

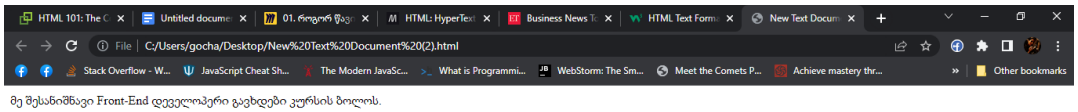


ამის შემდეგ შენს Text Document ფაილს გაუჩნდება დაბოლოვება .txt, რომელიც შეგიძლია შეცვალო .html-ით,რაც მოგცემს შედეგად იმას,რომ შენი ტექსტური დოკუმენტი,ორჯერ

დაწკაპუნებისას გაიხსნება პირდაპირ შენს default ბრაუზერში და ვიზუალურადაც ექნება default ბრაუზერის სახე.



შეგიძლია მაუსს ორჯერ დააჭირო და იხილო შენ მიერ შექმნილი პირველი ვებ-გვერდი!



და დასასრულს,სამოტივაციოდ ბილ მილსის,ფიზიკოსისა და ინტერდისციპლინარული პროგრამისტის სიტყვებს მოგიყვან: “არასოდეს დაგავიწყდეთ, რომ არ არსებობს თანდაყოლილი ოსტატობა. ადვილია, შეხედოთ დიდებულ პროგრამისტებს და თავი იგრძნოთ ძალიან არასრულფასოვნად, თითქოს მათ დონეს ვერასოდეს მიაღწევთ ან ვერ გახდებით 'ნამდვილი' პროგრამისტი. მაგრამ მე მქონდა ბედნიერება, შევხვედროდი რამდენიმე საუკეთესო ვებ პროგრამისტს და მათ ყველას ჰქონდათ იგივენაირი გამოცდილება - თითოეული მათგანი თვითმარქვიად მიიჩნევდა თავს, როდესაც ახალი დაწყებული ჰქონდათ პროგრამირება და არც ერთი მათგანი არ თვლიდა, რომ პროგრამისტობა ადვილია (მიუხედავად იმისა, რამდენად ადვილად მოგვაჩვენებენ ხოლმე ისინი ამას ახლა). თუკი კოდის ერთი ხაზი მაინც დაგიწერიათ, მაშინ ნამდვილი პროგრამისტი ხართ - და თუ გააგრძელებთ კოდის წერასა და სწავლას, ერთ დღეს ოსტატიც გახდებით.”