**SlayViber**

**შესავალი:**

კოვიდ პანდემიის დროს, განსაკუთრებით აქტუალური გახდა სოციალური პლატფორმები. მათ შორის განსაკუთრებით დიდი პოპულარულობით სარგებლობდა ვიდეო ჰოსტინგ სერვისი - „TikTok”. ა.შ.შ-ში პანდემიისას მისი მუდმივი მომხმარებლების რაოდენობა 15-დან 25 წლამდე სეგმენტში 180% -ით გაიზარდა. სტუდენტს სურს, რომ შექმნას მსგავსი პლატფორმა, თუმცა განსხვავებული ფუნქციონალით. მანამ, სანამ პროექტის რეალიზაციას შეუდგება,მას სურს შექმნას კომპიუტერული სიმულაციური მოდელი, რომლითაც წინასწარ განსაზღვრავს მის ღირებულებას. აქ იწყება თქვენი დავალება!

**სპეციფიკაციები:** ამ დავალებაში თქვენ შექმნით მომხმარებლის კლასს **SlayViber** -აპლიკაციისთვის “SlayVibe”. თქვენ ასევე შექმნით ფაილს main ფუნქციისთვის (*driver.cpp*) რომელშიც შექმნით SlayViber ტიპის ობიექტებს მათი ფუნქციონალის შესამოწმებლად. ამ ტიპის ფაილს ეწოდება დრაივერი, რადგან იგი გამოიყენება რაღაცის (ახლის) შესამოწმებლად. თუ როგორ უზრუნველყოფთ ახალი ფუნქციონალის შემოწმებას, თქვენზეა დამოკიდებული.

თქვენი SlayViber კლასი შეიცავს შემდეგ

**წევრ მონაცემებს(ცვლადებს):**

* string username
* string password
* bool streaming
* int followers
* bool isRegistered
* bool hasRestriction
* vector<string> visitors (შეგიძლიათ სასურველი კონტეინერის გამოყენება)

**წევრ ფუნქციებს:**

* უპარამეტრო კონსტრუქტორი (default constructor), რომელიც ქმნის არარეგისტრირებულ მომხმარებელს. (მომხარებელი,რომელსაც შეუძლია მხოლოდ ვიდეოების ნახვა, არ შეუძლია სხვისი გამოწერა, გააჩნია რაღაც default username (მაგალითად: user4352123412), არ გააჩნია პაროლი და ა.შ)
* პარამეტრიანი კონსტრუქტორი, რომელსაც გადასცემთ მნიშვნელობებს იმისათვის, რომ განსაზღვროთ ობიექტის username, password, followers (გაჩუმების პრინციპით მნიშვნელობით 0), hasRestriction (გაჩუმების პრინციპით მნიშვნელობით false).
* ფუნქცია print, რომელიც გამოიტანს ინფორმაციას მომხმარებლის შესახებ ორგანიზებული ფორმით.
* ფუნქცია visit, რომელსაც იძახებს კლასის ობიექტი და პარამეტრად იღებს კლასის სხვა ობიექტს, ფუნქცია გამოიტანს მონახულებული მომხმარებლის შესახებ ინფორმაციას შეზღუდულ ფორმატში (შეზღუდვა თქვენ განსაზღვრეთ :) ). ამავდროულად მონახულებული მომხმარებლის visitors ველში უნდა დაემატოს მომნახულებელი ობიექტის სახელი. ასევე, მომნახულებელმა მომხმარებელმა უნდა გადაწყვიტოს სურს თუ არა მონახულებული მომხმარებლის გამოწერა (შესაბამისად გაიზარდოს ან არ გაიზარდოს მომხმარებლის followers რაოდენობა).
* ფუნქცია registration, რომელსაც გამოიძახებს კლასის არარეგისტრირებული ობიექტი და დაარეგისტრირებს ამ მომხმარებელს (ინტერაქციულ რეჟიმში).
* ფუნქცია goLive, რომელსაც გამოიძახებს ობიექტი და ობიექტს განესაზღვროს Streaming – true. ფუნქციის მუშაობისას მომხმარებელს ან მოემატოს (5% -დან 40% მდე ) ან დააკლდეს ( 0% - დან 30% ) გამომწრების რაოდენობა შემთხვევითი პრინციპით. იმ შემთხვევაში, თუ მომხმარებელს დააკლდა გამომწერების 25% ზე მეტი, პლატფორმამ მომხმარებელი სავარაუდო წესების დარღვევის გამო შეზღუდოს დაუყოვნებლივ (hasRestriction – true) და გათიშოს ლაივიდან (Streaming – false).
* ფუნქცია, რომლის იდეაც და იმპლემენტირებაც თქვენზეა დამოკიდებული, აუცილებელია რომ ამ ფუნქციას ჰქონდეს კლასის სულ მცირე 2 წევრ-ცვლადზე ზეგავლენა.

**შენიშვნა:** ძალიან კარგად დაფიქრდით ზემოთ ჩამოთვლილი ფუნქციების დაწერის დროს იმათზე, თუ როგორ უნდა იქნას გამოყენებული ისინი. ნამდვილად არსებობს გარემოებები, როდესაც ზოგიერთი ფუნქცია არ უნდა შესრულდეს. მაგალითად, თუ ობიექტს განსაზღვრული აქვს hasRestriction - true, ფუნქცია goLive გამოძახება ამ ობიექტისთვის „ჩაიშალოს“.

ინტერაქციულ რეჟიმში მონაცემების შეყვანისას გაითვალისწინეთ input handling (მაგალითად: ისეთი username-ის შეყვანა რომელიც უკვე არსებობს არასწორია :) ).

**კლასის ტესტირება:** თქვენს დრაივერ პროგრამაში უნდა გატესტოთ თქვენი კლასი სრულყოფილად. მასში უნდა გამოიძახოთ ყველა წევრ-ფუნქცია. ეკრანზე უნდა გამოიტანოთ რას აკეთებს და ამ ცვლილებების შედეგად რა ხდება, რა მოსდის შესაბამის ობიექტს.

შეფასებული იქნებება კონსტანტების გამოყენების ადეკვატურობა: ფუნქციების პარამეტრების რომლებიც არ შეცვლის ფუნქციას, უნდა იყოს გამოცხადებული const-ის გამოყენებით. წევრ ცვლადებს თან უნდა ახლდეს შესაბამისი გეთერები და სეთერები საჭიროებისამებრ.