წრფივი კოდებით ინფორმაციის დაცვა

შესავალი

ამ დავალებაში, თქვენი მიზანი იქნება მაგენერირებელი მატრიცით მოცემული წრფივი კოდების გამოყენება. მათ შორის, მაგენერირებელი მატრიცის სტანდარტულ ფორმაზე დაყვანა, შემმოწმებელი მატრიცის მიღება, ინფორმაციის კოდირება და უკან ამოღება შეცდომების გამოსწორებით.

ყველა პროგრამას უნდა დაარქვათ ის სახელი, რაც შესაბამის საკითხში იქნება მითითებული. დაწერილი პროგრამები (მხოლოდ source-ები!) მოათავსეთ ერთ folder-ში. ამ folder-ს დაარქვით თქვენი freeuni ელ-ფოსტის მისამართი სიმბოლო "@"-მდე. ეს folder-ი შეკუმშეთ zip ფაილად და არქივი ატვირთეთ google classroom-ზე.

თუ სხვანაირად არ არის მითითებული, როდესაც პროგრამა ფაილიდან კითხულობს ან ფაილში წერს ინფორმაციას წასასაკითხი ფაილის სახელი პირველ არგუმენტად გადმოგეცემათ და პასუხის ჩასაწერი ფაილის სახელი მეორე არგუმენტად.

თქვენი დაწერილი პროგრამები არ უნდა მუშაობდეს ძალიან ნელა, და არ უნდა იყენებდეს ძალიან ბევრ მეხსიერებას. კონკრეტულად, პროგრამა შემოწმდება 1 ბირთვიან (საკმაოდ ნელ) პროცესორზე, ნახევარ გიგაბაიტიანი ოპერატიული მეხსიერებით. საჯარო ტესტებში მოცემული თითოეული შემთხვევისთვის თქვენს პროგრამას არ უნდა სჭირდებოდეს 15 წამზე მეტი დრო.

დავალება 8 ქულიანია და თითო ქულა კურსის ქულის 1%-ის ტოლფასია.

1. მაგენერირებელი მატრიცის სტანდარტულ ფორმაზე დაყვანა [1.5 ქულა]

ამ ნაწილში უნდა დაწეროთ პროგრამა, რომელიც პირველი შემავალი ფაილიდან წაიკითხავს ორობითი კოდის მაგენერირებელ მატრიცას და დააბრუნებს ამ კოდის ექვივალენტური კოდის მაგენერირებელ მატრიცას სტანდარტულ ფორმაში.

მატრიცის ფორმატი იქნება შემდეგი:

პირველ ხაზზე წერია ცარიელი ადგილით გაყოფილი რიცხვები n და k. n – სიტყვების სიგრძეა, ხოლო k – წრფივად დამოუკიდებელი სიტყვების რაოდენობა. შემდეგი k ხაზიდან თითოზე წერია n ციფრიანი “სტრინგი”, რომელიც მხოლოდ 0-ების და 1-ებისგან შედგება. თქვენს მიერ გამოტანილი მატრიცაც ზუსტად იგივე ფორმატში უნდა იყოს (n და k-ს ჩათვლით).

თქვენმა პროგრამამ ასევე უნდა დააბრუნოს სვეტების ის გადანაცვლება რომლის გაკეთებაც საჭიროა იმისთვის რომ საწყისი კოდიდან სტანდარტული ფორმის შესაბამისი კოდი მივიღოთ. კერძოდ გამომავალ ფაილში მარტიცის ჩაწერის შემდეგ ახალ ხაზზე ჩაწერეთ space-ებით გამოყოფილი n ცალი რიცხვი p_1, p_2, … p_n, სადაც p_i = i-ური კოდის პირველი ბიტი სტანდარტული ფორმის მატრიცის შესაბამის კოდში მერამდენე პოზიციაზეა.

პროგრამას დაარქვით “StandardForm.xxx” (სადაც xxx თქვენს მიერ არჩეულ პროგრამირების ენაზე დამოკიდებული გაფართოებაა).

2. შემმოწმებელი მატრიცის პოვნა [1.5 ქულა]

ამ ნაწილში გადმოგეცემათ კოდის მაგენერირებელი მატრიცა (არა აუცილებლად სტანდარტულ ფორმაში). თქვენი მიზანია იპოვოთ ამ კოდის შემმოწმებელი მატრიცა. შემმოწმებელი მატრიცა კონკრეტულად ამ კოდს უნდა შეესაბამებოდეს და არა მის ექვივალენტურს.

პროგრამას დაარქვით “ParityCheck.xxx”

3. გაშიფრვის ცხრილის აგება [1.5 ქულა]

ამ ნაწილში დაწერილი პროგრამის შედეგი მეხუთე ნაწილში გამოგადგებათ, ამიტომ ჯობია სანამ ამ ნაწილის კეთებას დაიწყებთ დავალების მეოთხე და მეხუთე ნაწილებიც წაიკითხოთ.

შემომავალ მონაცემებად პირველ ფაილში ისევ გადმოგეცემათ კოდის მაგენერირებელი მატრიცა (ისევ არა აუცილებლად სტანდარტულ ფორმაში). წინა ნაწილის გამოყენებით შეგიძლიათ ამ მატრიცის შესაბამისი შემმოწმებელი მატრიცაც იპოვოთ. პროგრამას მეორე არგუმენტად გადმოეცემა ფაილი, რომელშიც მხოლოდ ერთი მთელი რიცხვი e ეწერება. ეს რიცხვი არის შეცდომების რაოდენობა, რომლის გასწორებაც აუცილებლად გვინდა. გარანტირებულია, რომ კოდის ამდენი შეცდომის გასწორების საშუალება ექნება. თქვენი პროგრამის მიზანია შეადგინოს მიღებული ინფორმაციის გაშიფრვისთვის საჭირო მონაცემები. ეს მონაცემები უნდა მოიცავდეს:

1. კოდური სიტყვიდან როგორ უნდა მივიღოთ სინდრომი (შემმოწმებელი მატრიცა)
2. რომელ სინდრომს რა შეცდომების პოზიციები შეესაბამება (გაშიფრვის ცხრილი)

ეს მონაცემები მესამე არგუმენტად გადმოცემულ ფაილში უნდა ჩაწეროთ. შენახვის ფორმატი შეგიძლიათ თქვენ თვითონ შეარჩიოთ, მთავარია ამ ცხრილის გამოყენებით მეხუთე ნაწილის პროგრამამ წრფივ დროში იმუშავოს. გარანტირებულია რომ $n^e \le 10^5$ (ანუ თუ წესიერი ფორმატი გამოიყენეთ ამ ცხრილმა ძალიან დიდი ადგილი არ უნდა დაიკავოს). ეს ნაწილი შემოწმდება მეხუთე ნაწილთან ერთად და ქულაც მხოლოდ ამ ნაწილთან ერთად დაიწერება.

პროგრამას დაარქვით “DecodingTable.xxx”.

4. წრფივი კოდით ინფორმაციის დაშიფრვა [2 ქულა]

ამ ნაწილში თქვენი შემავალი მონაცემები ორი ფაილისგან შედგება. ამ ფაილების სახელები პირველ და მეორე არგუმენტებად გადმოგეცემათ, ხოლო პასუხი სადაც უნდა ჩაწეროთ იმ ფაილის სახელი მესამე არგუმენტად.

პირველ ფაილში წერია კოდის მაგენერირებელი მატრიცა ამ დავალების პირველ ნაწილში მოცემულ ფორმატში, მაგრამ არა აუცილებლად სტანდარტულ ფორმაში. მეორე ფალში წერია მონაცებები, რომლებიც ამ კოდით უნდა დაშიფროთ. ყურადღება მიაქციეთ იმას, რომ მონაცემები ნებისმიერია და ამიტომ სანამ დაშიფრავდეთ დავალება 0-ის SimpleRead-ით ორობითი მიმდევრობა უნდა წაიკთხოთ. გარანტირებულია რომ ფაილის ზომა ისეთი იქნება რომ მონაცემების კოდირებისას ზედმეტი ადგილი არ დაგრჩეთ (ანუ თუ მაგალითად კოდი ყოველ სიტყვაზე 3 ბიტს გადასცემს ფაილის ზომა აუცილებლად 3 ბიტის ჯერადი იქნება).

თქვენმა პროგრამამ მესამე ფაილში უნდა გამოიტანოს კოდით დაშიფრული მონაცემები. ამ მონაცემების სიგრძე შეიძლება არ იყოს ყოველთვის 8 ბიტის ჯერადი, ამიტომ ფაილში ჩაწერისას დავალება 0-ის CompleteWrite მეთოდი უნდა გამოიყენოთ.

პროგრამას დაარქვით “Encode.xxx”.

5. მიღებულ მონაცემებში შეცდომების გასწორება და საწყისი ინფორმაციის ამოღება [1.5 ქულა]

ამ ნაწილში ორი შემომავალი ფაილი გექნებათ. პირველი მათგანი იქნება მესამე ნაწილში მიღებული დამხმარე მონაცემების ფაილი. მეორე მათგანი კი იქნება მიღებული მონაცემები. მიღებულ მონაცემებში შემთხვევითი წესით დამატებული იქნება შეცდომები მაგრამ თითოეულ კოდირებულ სიტყვაში არ იქნება იმაზე მეტი შეცდომა ვიდრე მოცემულ კოდს შეუძლია გაასწოროს, ასევე გარანტირებულია რომ CompleteWrite-ს მიერ დამატებულ ფაილის დაბოლოებაში შეცდომები საერთოდ არ იქნება. ამითი ხმაურიან არხზე მონაცემების გადაცემის სიმულაცია მოხდება.

თქვენი პროგრამის მიზანია გამოიყენოს პირველი და მეორე ფაილის მონაცემები და აღმოაჩინოს ის პოზიციები რომლებზეც მოხდა შეცდომა. შემდეგ გამოასწოროს შეცდომები და აღადგინოს საწყისი (გადმოცემული) მონაცემები. ორიგინალი ინფორმაციის აღდგენა არ არის საჭირო, მხოლოდ გადმოცემული კოდური სიტყვების სწორი მიმდევრობაა საჭირო შეცდომების გარეშე. აღდგენილი მონაცემები მესამე არგუმენტად გადმოცემულ ფაილში უნდა ჩაწეროთ.

პროგრამას დაარქვით “Decode.xxx”