ციკლური BCH კოდების გამოყენება

შესავალი

ამ დავალებაში უნდა დაწეროთ ციკლურ კოდებთან სამუშაო რამდენიმე სასრგებლო პროგრამა (მათ შორის კოდირება/დეკოდირების პროგრამები) და ასევე მოცემული პარამეტრების BCH კოდების მაგენერირებელი პროგრამა. ამ დავალების ფარგლებში კოდის სიმბოლოების სიმრავლე ყოველთვის მარტივი ველი, ანუ მთელი რიცხვების მოცემულ მარტივ რიცხვზე გაყოფის ნაშთების ველი იქნება.

დავალება შეგიძლიათ ნებისმიერი მიდგომით გააკეთოთ (მათ შორის თუ გინდათ სრულიად უსტრუქტუროდ), მაგრამ სავარაუდოდ საქმე გაგიმარტივდებათ თუ წინასწარ გააკეთებთ მრავალწევრის შესაბამის კლასს თავისი გამრავლების და ნაშთიანი გაყოფის მეთოდებით.

პროგრამები შეგიძლიათ ნებისმიერ ენაზე დაწეროთ, რომლის კომპილატორი ან ინტერპრეტატორი უფასოა და მარტივად მოიპოვება ინტერნეტში. არ გამოიყენოთ არასტანდარტული (ან მითუმეტეს ფასიანი) ბიბლიოთეკები. პროგრამა უნდა ეშვებოდეს თქვენს მიერ არჩეული ენის სტანდარტული კომპილატორით (ან ინტერპრეტატორით). ყველა პროგრამას უნდა დაარქვათ ის სახელი, რაც საკითხში იქნება მითითებული და ყველა პროგრამა უნდა მოათავსოთ ერთ folder-ში. გამოგზავნისას ამ folder-ს დაარქვით თქვენი freeuni მეილის სახელი (ანუ თქვენი ელ-ფოსტის მისამართი @-მდე) და ისე გამოგზავნეთ.

ზოგიერთი პროგრამა ფაილიდან კითხულობს ან ფაილში წერს ინფორმაციას. ასეთ შემთხვევებში წასასაკითხი/ჩასაწერი ფაილების სახელები პროგრამამ უნდა მიიღოს როგორც არგუმენტები (ჯერ წასაკითხი, შემდეგ ჩასაწერი).

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

1. მრავალწევრის შემოწმება და შესაბამისი შემმოწმებელი მრავალწევრის პოვნა [2 ქულა]

ამ ნაწილში მოცემული გვექნება რაიმე მრავალწევრი. ჩვენი მიზანია შევამოწმოთ მოცემული მრავალწევრი შეგვიძლია თუ არა მოცემული სიგრძის კოდის მაგენერირებელ მრავალწევრად გამოვიყენოთ და თუ შესაძლებელია მაშინ ვიპოვოთ მისი შესაბამისი შემმოწმებელი მრავალწევრი.

პირველ არგუმენტად გადმოცემულ ფაილში პირველ ხაზზე ეწერება ცილკური კოდის ანბანის ზომა p (რომელიც აუცილებლად მარტივი რიცხვი იქნება), მეორე ხაზზე ეწერება კოდის სიგრძე n და ხოლო მესამე ხაზზე space-ებით გამოყოფილი n ცალი კოეფიციენტი g_0, g_1, … g_{n-1}. ეს კოეფიციენტები 0- დან p-მდე რიცხვები იქნება და მაგენერირებელი მრავალწევრი გამოისახება როგორც: $g_0 + g_1 \cdot x + g_2 \cdot x^2 + \ldots + g_{n-1} \cdot x^{n-1}$.

თვენი მიზანია შეამოწმოთ მართლა არის თუ არა შესაძლებელი, რომ მოცემული მრავალწევრი კოდის მაგენერირებელი მრავალწევრი იყოს და თუ შესაძლებელია იპოვოთ შესაბამისი შემმოწმებელი მრავალწევრი. პასუხი გამოიტანეთ მეორე არგუმენტად გადმოცემულ ფაილში. პასუხის პირველ ხაზზე დაწერეთ “YES” თუ მოცემული მრავალწევრი მართლა შეიძლება იყოს კოდის მაგენერირებელი მრავალწევრი და “NO” თუ ეს არ არის შესაძლებელი. თუ პირველ ხაზზე “YES” დაწერეთ, მეორე ხაზზე გამოიტანეთ space-ებით გამოყოფილი n ცალი კოეფიციენტი ქვედა (x^0-ის) კოეფიციენტიდან ზედასკენ (x^{n-1}-ის კოეფიციენტი) დალაგებული.

პროგრამას დაარქვით “ParityCheck.xxx”, სადაც xxx პროგრამირების ენაზე დამოკიდებული გაფართოებაა.

2. ციკლური კოდით ინფორმაციის კოდირება [1.5 ქულა]

ამ ნაწილში პირველ ფაილში, მოცემული გექნებათ კოდის პარამეტრები და მაგენერირებელი მრავალწევრი (წინა ნაწილის მსგავს ფორმატში). მეორე ფაილში კი პირველ ხაზზე ეწერება რიცხვი k_{in}, მეორე ხაზზე კი მოცემული გექნებათ k_{in} ცალი space-ით გამოყოფილი მთელი რიცხვი 0-დან p-მდე (სადაც p კოდის ანბანის ზომაა). რიცხვი k_{in} აუცილებლად n·r-ის ჯერადი იქნება, სადაც r კოდის ეფექტურობაა. თქვენი მიზანია მეორე ფაილში მოცემული ინფორმაციის კოდირება მოცემული კოდით. მესამე არგუმენტად გადმოცემულ ფაილში პირველ ხაზზე უნდა გამოიტანოთ რიცხვი k_{out}, ხოლო შემდეგ ხაზზე გადმოცემული ინფორმაციის კოდირების შედეგად მიღებული k_{out} ცალი space-ით გამოყოფილი რიცხვი 0-დან p-მდე.

პროგრამას დაარქვით: “Encode.xxx”.

3. ციკლური კოდით შენახული ინფორმაციის ამოღება [1.5 ქულა]

ციკლურ კოდებში შეცდომების შესწორება იგივენაირად ხდება როგორც სხვა წრფივ კოდებში (სიმდრომების და შეცდომების ვექტორების ცხრილი აგებით). რადგან წინა დავალებაში ეს უკვე გაკეთებული გვაქვს ამ დავალებაში ჩავთვლით რომ შეცდომები უკვე შევასწორეთ და მხოლოდ შესწორებული კოდური სიტყვებიდან საწყისი ინფორმაციის აღდგენას გავაკეთებთ.

პირველ ფაილში მოცემული გექნებათ კოდის პარამეტრები და მაგენერირებელი მრავალწევრი, მეორე ფაილში ეწერება კოდირებული ინფორმაცია ისეთივე ფორმატში როგორშიც წინა ნაწილის პროგრამა აბრუნებს. გადმოცემულ კოდურ სიტყვებში არ იქნება შეცდომები, ანუ ყველა n სიმბოლოიანი მიმდევრობა სწორი კოდური სიტყვის შესაბამისი იქნება. თქვენ მესამე არგუმენტად გადმოცემულ ფაილში უნდა ჩაწეროთ ის საწყისი ინფორმაცია რომლის კოდირებაც მოხდა ამ მაგენერირებელი მრავალწევრის შესაბამისი კოდით. პასუხი იმავე ფორმატით გამოიტანეთ რა ფორმატითაც წინა ნაწილში გადმოგეცათ ეს ინფორმაცია.

პროგრამას დაარქვით “Decode.xxx”.

4. მინიმალური მრავალწევრის პოვნა [1.5 ქულა]

BCH კოდების ასაგებად როგორც იცით დაგვჭირდება გავრცობილი ველის მოცემული ელემენტისთვის მინიმალური მრავალწევრის პოვნა საწყის ველზე.

პირველ ფაილში პირველ ხაზზე მოცემული იქნება საწყისი მარტივი ველის ზომა p, მეორე ხაზზე გამვრცობი მრავალწევრის ხარისხი n, ხოლო მესამე ხაზზე space-ებით გამოყოფილი n + 1 ცალი გამვრცობი მრავალწევრის კოეფიციენტები დაბალი ხარისხის კოეფიციენტიდან მაღალი ხარისხისკენ. გარანტირებულია რომ ეს მრავალწევრი ე.წ. პრიმიტიულია, ანუ ამ მრავალწევრით ველის გავრცობისას ელემენტი α, რომელიც ამ მრავალწევრის ფესვია, პრიმიტიული იქნება.

მეორე ფაილში ეწერება ერთადერთი მთელი რიცხვი i. თქვენი მიზანია ელემენტი α^i-ს მინიმალური მრავალწევრი იპოვოთ.

ნაპოვნი მინიმალური მრავალწევრი მესამე არგუმენტად გადმოცემულ ფაილში გამოიტანეთ შემდეგ ფორმატში: პირველ ხაზზე გამოიტანეთ მრავალწევრის სიმბოლოების ანბანის ზომა p, მეორე ხაზზე მრავალწევრის ხარისხი – m, მესამე ხაზზე კი m + 1 ცალი space-ით გამოყოფილი კოეფიციენტი (დაბალი რიგის კოეფიციენტიდან მაღალი რიგის კოეფიციენტისკენ).

პროგრამას დაარქვით “MinimalPolynomial.xxx”.

5. მოცემული BCH კოდის მაგენერირებელი მრავალწევრის პოვნა [1.5 ქულა]

ამ ნაწილში საპოვნელი გექნებათ F_p ანბანზე $N = p^n - 1$ სიგრძის და მინიმუმ δ მინიმალური მანძილის მქონე BCH კოდი. საქმის გასამარტივებლად, ველის გასავრცობად მარტივი

მრავალწევრის ძებნა და შემდეგ პრიმიტიული ელემენტის ძებნა არ მოგიწევთ. კოდის პარამეტრებთან ერთად გადმოგეცემათ მოცემული n ხარისხის ერთ-ერთ პრიმიტიული მრავალწევრიც.

პირველ არგუმენტად გადმოცემულ ფაილში პირველ ხაზზე ეწერება p (სიმბოლოების ანბანის ზომა), მეორე ხაზზე ეწერება – n (თქვენი კოდის ზომა N უნდა იყოს $p^n - 1$), მესამე ხაზზე კი ეწერება n + 1 ცალი space-ით გამოყოფილი კოეფიციენტი. ეს კოეფიციენტები n ხარისხის პრიმიტიული მრავალწევრის კოეფიციენტები იქნება.

მეორე ფაილში გადმოგეცემათ ერთადერთი მთელი რიცხვი δ – მინიმალური მანძილი რომელიც თქვენმა აგებულმა BCH კოდმა აუცილებლად უნდა მოგვცეს. მესამე არგუმენტად გადმოცემულ ფაილში უნდა გამოიტანოთ აგებული BCH კოდის პარამეტრები და მაგენერირებელი მრავალწევრი შემდეგ ფორმატში: პირველ ხაზზე გამოიტანეთ რიცხვი p, მეორე ხაზზე რიცხვი $N = p^n - 1$, მესამე ხაზზე კი თქვენს მიერ ნაპოვნი BCH კოდის მაგენერირებელი მრავალწევრის N ცალი კოეფიციენტი (x^0-ის კოეფიციენტიდან x^{N-1}-ის კოეფიციენტის ჩათვლით). სავარაუდოდ ნაპოვნ მრავალწევრს (N – 1)-ზე ნაკლები ხარისხი ექნება, მაგრამ ბუნებრივია ასეთ შემთხვევაში დარჩენილი ხარისხების კოეფიციენტებად ნულები უნდა გამოიტანოთ.

თქვენს მიერ გამოტანილი BCH კოდი აუცილებლად გადმოცემული პრიმიტიული მრავალწევრის ფესვი α და მისი ხარისხების $\{\alpha^1\ \alpha^2\ \dots\ \alpha^{\delta-1}\}$ მინიმალური მრავალწევრებით უნდა იყოს აგებული.

პროგრამას დაარქვით “BCH.xxx”.