ფაილებიდან ბიტების წაკითხვა/ჩაწერა

შესავალი

კურსის განმავლობაში ხშირად დაგვჭირდება მოცემული ფაილის ბიტებთან მუშაობა, შესაბამისად მნიშვნელოვანია შეგეძლოთ მოცემული ფაილიდან ბიტების ამოღება და მოცემული ბიტების ფაილად ჩაწერა, ისე რომ ერთი ბიტის ჩასაწერად ბაიტის გამოყენება არ დაგჭირდეთ.

ამ დავალებაში თქვენ უნდა დაწეროთ პროგრამები, რომლებიც საშუალებას მოგცემთ მოცემული ფაილიდან წაიკითხოთ ბიტების მიმდევრობა და მოცემული ბიტების მიმდევრობისთვის ჩაწეროთ ფაილი.

ამ დავალებაში დაწერილი მეთოდები ყველა სხვა დავალებაში გამოგადგებათ, ასე რომ ამ დავალების გაკეთება მაინც მოგიწევთ, თუ ერთი სხვა დავალების გაკეთება მაინც გინდათ. შესაბამისად, თქვენვე გაგმარტივდებათ საქმე თუ ამ დავალების კოდი კარგად იქნება ორგანიზებული და მისი ნაწილების გამოყენება სხვა პროგრამებში გადაკეთების გარეშე იქნება შესაძლებელი.

ყველა პროგრამას დაარქვით შესაბამის საკითხში მითითებული სახელი. დაწერილი პროგრამების source-ები მოათავსეთ ერთ folder-ში. ამ folder-ს დაარქვით თქვენი freeuni ელ-ფოსტის მისამართი სიმბოლო "@"-მდე, შეკუმშეთ zip ფაილად და ატვირთეთ classroom-ზე.

თქვენი დავალების გადასამოწმებლად მოცემული გაქვთ თითოეული ნაწილის ტესტები. ამ ტესტებიდან ზოგი ფაილი ტექსტურია, მაგრამ ზოგში უბრალოდ ორობითი კოდი წერია და ტექსტურ პროგრამაში გახსნისას გაუგებარი სიმბოლოები დაგხვდებათ. ასეთი ფაილების გასარჩევად შეგიძლიათ რომელიმე ორობითი ფაილების წამკითხავი გამოიყენოთ. მაგალითად ერთ-ერთი ასეთი ონლაინ პროგრამა არის ვებ-გვერდზე: hexed.it.

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.

1. ფაილიდან ბიტების წაკითხვა [1 ქულა]

ამ ნაწილში თქვენი მიზანია მოცემული ფაილიდან, რომლის სახელიც პროგრამას პირველ არგუმენტად გადმოეცემა, წაიკითხოთ ყველა ბიტი და მეორე ფაილში (რომლის სახელიც მეორე არგუმენტად გადმოგეცემათ) გამოიტანოთ ბიტების მიმდევრობა ჩაწერილი ჩვეულებრივი სიმბოლოებით '0' და '1' (space-ების და ყველანაირი სხვა კომენტარის გარეშე).

ამ კურსში, მოცემული ბაიტის პირველ ბიტს ვეძახით მის ყველაზე მნიშვნელოვან ბიტს. ანუ სიმბოლო 'a'-ს შეესაბამება მიმდევრობა 01100001 და არა 10000110.

მაგალითად თუ ფაილში წერია მხოლოდ “hello”, თქვენმა პროგრამამ უნდა ჩაწეროს: “01101000011001010110110001101100011011111”.

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

2. ფაილში 8-ის ჯერადი სიგრძის ბიტების ნაკადის ჩაწერა [1 ქულა]

როგორც ხვდებით პირველი ნაწილის შედეგად გამოტანილი ბიტების მიმდევრობის სიგრძე ყოველთვის 8-ის ჯერადი იქნება, იმიტომ რომ თითო ბაიტში 8 ბიტი შედის და ნებისმიერი ფაილი მთელი ბაიტებისგან შედგება.

ამ ნაწილში თქვენი მიზანია წინა პროცესის შებრუნება, ანუ ბიტების მიმდევრობა უნდა წაიკითხოთ პირველ არგუმენტად გადმოცემული ფაილიდან და ამ ბიტების შესაბამისი ფაილი ჩაწეროთ მეორე არგუმენტად გადმოცემულ ფაილში. წაკითხული ბიტების მიმდევრობის სიგრძე ყოველთვის იქნება 8-ის ჯერადი.

პროგრამას დაარქვით “SimpleWrite.xxx”.

3. სხვა სიგრძის ბიტების მიმდევრობის ჩაწერა [1 ქულა]

კურსის განმავლობაში ხშირად დაგვჭირდება არა რვის ჯერადი სიგრძის ბიტების მიმდევრობების ფაილში ჩაწერა.

ასეთ შემთხვევაში შეგვიძლია ბიტების მიმდევრობას ბოლოში 0-ები მივუწეროთ სანამ მთელ ბაიტს არ შეავსებს, მაგრამ ეს არასწორი იქნება, იმიტომ რომ წაკითხვისას არ გვეცოდინება ბოლოს მიწერილი 0-ები მართლა შედიოდა ჩვენს ბიტების მიმდევრობაში თუ ბაიტამდე შევსებისთვის დაემატა. შესაბამისად მიმდევრობები “0101” და “010100” ერთნაირად ჩაიწერება და აღდგენისას ართმანეთისგან არ გაირჩევა.

ამის გამოსასწორებლად გამოვიყენებთ შემდეგ მეთოდს: ბიტების მიმდევრობას ბოლოში მივაწერთ ბიტ 1-ს და შემდეგ იმდენ 0-ს რამდენის დასჭირდება ბაიტამდე შესავსებათ. წაკითხვისას კი გვეცოდინება, რომ წაკითხულ ბიტების მიმდევრობას ბოლოდან უნდა მოვაჭრათ პირველ 1-მდე ყველაფერი (ამ 1-ის ჩათვლით). ეს წესი ჩვენი ფაილის ზომას მაქსიმუმ 1 ბაიტით გაზრდის, მაგრამ სამაგიეროდ ყველა მიმდევრობას თავისი შესაბამისი ფაილი ექნება და პირიქით. მნიშვნელოვანია რომ ბოლოში 1000...-ს მიწერა მაინც გავაკეთოთ თუ ბიტების მიმდევრობა თავიდანვე 8-ის ჯერადი იყო. ამ შემთხვევაში 10000000-ს მიწერა მოგვიწევს. ეს რომ არ გავაკეთოთ ადვილი შესამოწმებელია რომ ზემოთ აღწერილი პრობლემა მაინც გვექნება.

ამ ნაწილში თქვენი მიზანია პირველ არგუმენტად გადმოცემული ფაილიდან წაიკითხოთ ნებისმიერი სიგრძის ბიტების მიმდევრობა, აღწერილი სქემით გადაიყვანოთ 8-ის ჯერადი სიგრძის მქონე მიმდევრობაში და შემდეგ წინა ნაწილის ანალოგიურად მეორე არგუმენტად გადმოცემულ ფაილში ჩაწეროთ ამ მიმდევრობის შესაბამისი ბაიტები.

8-ის ჯერადი სიგრძის მიმდევრობებისთვის ეს პროგრამა ერთი ბაიტით უფრო დიდ ფაილს უნდა აბრუნებდეს ვიდრე წინა პროგრამა.

პროგრამას დაარქვით: “CompleteWrite.xxx”.

4. ნებისმიერი სიგრძის ბიტების მიმდევრობის აღდგენა [1 ქულა]

ამ ნაწილში თქვენი მიზანია წინა ნაწილში ჩაწერილი ფაილიდან ბიტების საწყისი მიმდევრობის აღდგენა. ეს პროგრამა პირველ არგუმენტად გადმოცემული ფაილიდან უნდა კითხულობდეს ბაიტების მიმდევრობას, გარდაქმნიდეს ბიტების მიმდევრობად, წინა ნაწილში დამატებულ კუდს შლიდეს და რაც დარჩება, '0' და '1' სიმბოლოებით გამოქონდეს მეორე არგუმენტად გადმოცემულ ფაილში. (როგორც პირველ ნაწილში)

პროგრამას დაარქვით “CompleteRead.xxx”.