***გამოცდის ფორმატი***

**\*მონიშნეთ გამოცდის ფორმატი (მიუთითეთ √)**

|  |  |
| --- | --- |
| დახურული წიგნი |  |
| ღია წიგნი | **√** |
| **\*ღია წიგნის შემთხვევაში მონიშნეთ გამოცდაზე ნებადართული ელემენტები (მიუთითეთ √)** | |
| სალექციო მასალები (პრეზენტაცია და სხვა) |  |
| ელექტრონული წიგნები | **√** |
| წიგნები |  |
| კონსპექტები |  |
| ლექსიკონი |  |
| კალკულატორი |  |
| ლეპტოპი/პლანშეტი |  |

**\* გამოცდის ჩატარების წესი იხილეთ ,,დესკტოპზე“ საქაღალდეში Exam materials**

***საგამოცდო საკითხების ფორმა***

***ვარიანტი # 1***

|  |  |  |  |
| --- | --- | --- | --- |
| ***სკოლა*/*საგანმანათლებლო პროგრამა*** | მათემატიკა და კომპიუტერული მეცნიერება | ***სტუდენტის მიერ მიღებული ქულა*** |  |
| ***საგანი*** | პროგრამირების პარადიგმები | | |
| ***ლექტორი*** | შ. ღვინეფაძე | | |
| ***კურსი*** | II | | |
| ***ჯგუფი*** |  | | |
| ***გამოცდის ფორმა*** | ღია წიგნი | | |
| ***გამოცდის***  ***ხანგრძლივობა*** | 3 საათი | | |
| ***მაქსიმალური ქულა*** | 100 | | |
| ***სტუდენტის სახელი და გვარი:*** | | | |

სახელი: ქულა:

შუალედური გამოცდა

პარადიგმებში

2018, 22 დეკემბერი 16:00 – 19:00

|  |  |  |
| --- | --- | --- |
| 1  50 ქულა | 2  50 ქულა | სულ |
|  |  |  |

შეასრულეთ შემდეგი ინსტრუქციები, წინააღმდეგ შემთხვევაში შესაძლოა თქვენი ნაშრომი არ შეფასდეს.  
1. ჩამოტვირთეთ paradigms-midterm3 ფოლდერი თქვენს დესკტოპზე. მასში უნდა იყოს 2 ფაილი threads.c და longestUniqum.scm.

2. ცვლილებები შეიტანეთ დავალების პირობით მითითებულ ფაილებში.

3. დააარქივეთ paradigms-midterm3 ფოლდერი, არქივს სახელად დაარქვით თქვენი მეილის პრეფიქსი, მაგალითად gboch12.rar.

4. ვებ ბრაუზერში გახსენით მისამართი [http://192.168.210.5](https://l.messenger.com/l.php?u=http%3A%2F%2F192.168.210.5%2F&h=ATNdNl_tUgM3RyoXqKtzPKRP12hygwyVI0pyQH9Q8ss9LJWHxdfFaufD0sZODW8AowpzwdCIygWTmkdTwBIdteLRo_hfKInsffPkNS6u9h_To4X4mdIiBDFsifj54UipvN3rxqw7KRPPZsVNx9Y) და ატვირთეთ არქივი.

5. არქივი დატოვეთ დესკტოპზე.

Command prompt-ის გამოსაძახებლად ფოლდერში Shift ღილაკთან ერთად დააკლიკეთ მაუსის მარჯვენა ღილაკს და აირჩიეთ open command prompt here

**ამოცანა 1. ნაკადები (50 ქულა)**

texts მასივის საშუალებით მოცემული გაქვთ ტექსტების სია, i ური წევრი ინახავს მისამართს მეხსიერებაში ადგილზე, სადაც ინახება ძალიან დიდი ზომის ტექსტი. ასევე მოცემული გაქვთ მზა countSymbol მეთოდი, რომელსაც გადაეცემა, მიმთითებელი start, ტექსტის ზომა len და სიმბოლო c. countSymbol მეთოდი აანალიზებს მეხსიერებაში start მისამართზე len ცალ სიმბოლოს და აბრუნებს რაოდენობას რამდენჯერ გვხვდება c სიმბოლო.

მოცემულ კოდში, main ფუნქციაში ხდება NUM\_COUNTERS ცალი ნაკადში Counter ფუნქციის შესრულება, ამ ნაკადებს დავარქვათ მთვლელები. მთვლელი ნაკადების ამოცანაა, თითოეული ტექსტისთვის დაითვალონ მასში, სიმბოლო 'a'-ის რაოდენობა. თან ისე, რომ დროის ნებისმიერ მომენტში, ყველა მთვლელი ნაკადი მუშაობდეს კონკრეტულ ერთ ტექსტში სიმბოლოების დათვლაზე. უფრო ზუსტად კი, თავიდან ყველა მთვლელი ნაკადი პარალელურად ითვლის 'a' სიმბოლოს რაოდენობას პირველ ტექსტში, დასრულების შემდეგ ყველა ნაკადი გადადის მეორე ტექსტის დამუშავებაზე და ასე შემდეგ.

მთვლელი ნაკადები შეძლებისდაგავარად თანაბრად ინაწილებენ ტექსტს და პარალელურად ითვლიან სიმბოლოების რაოდენობას თავთავიანთ ნაწილებში. მაგალითად თუ მაქვს ოთხი მთვლელი ნაკადი და ტექსტი "abcdefghk", ერთერთ ნაკადს შეხვდება "ab" ტექსტი გასაანალიზებლად, მეორეს "cd", მესამეს "ef", ხოლო მეოთხეს "ghk", ანუ ტექსტის ბოლოში დარჩენილი კუდი. რომელ მთვლელს რომელი ნაწილი შეხვდება, განისაზღვრება დასწრების პრინციპით, ანუ პირველად რომელ ნაკადსაც დაეთმობა რესურსი აირჩევს პირველ ნაწილს "ab"-ს, მომდევნო რომელსაც დაეთმობა რესურსი მეორეს და ასე შემდეგ.

რომელიმე ტექსტის ანალიზის დროს, თუ ერთერთი ნაკადი სხვებზე ადრე დაითვლის თავის ნაწილში სიმბოლოს რაოდენობას, იგი უნდა დაიბლოკოს და დაელოდოს სხვა მთვლელ ნაკადებს. მთვლელი რომელიც ბოლოს მორჩება თავის ნაწილში სიმბოლოების დათვლას, აუწყებს დანარჩენ ნაკადებს ამის შესახებ, რათა განიბლოკონ და ყველა ერთად გადავიდეს texts მასივის შემდეგი ელემენტის ანალიზზე.

main ფუნქციიდან მთვლელი ნაკადების გარდა ეშევება ლოგირების ნაკადი, რომელიც თავიდან გაჩერებულია და ელოდება პირველი ტექსტის ანალიზის დასრულებას. როგორც კი პირველ ტექსტში სიმბოლოების დათვლა დასრულდება, ლოგერმა უნდა დაბეჭდოს, ამ ტექსტის ნომერი texts მასივში(ანუ ამ შემთხვევაში 0), 'a' სიმბოლოს რაოდენობა და იმ ნაკადის ნომერი, რომელმაც ყველაზე ბოლოს დაასრულა თავისი ნაწილის ანალიზი. ამის შემდეგ ლოგერი ისევ მოლოდინის რეჟიმში გადადის, მანამ სანამ არ დასრულდება მეორე ტექსტის ანალიზი და ასე შემდეგ.

შენიშვნები:

1. არ უნდა იყენებდეს busy waiting მექანიზმს.

2. არ უნდა გქონდეთ dead lock-ები და ამოცანის შესრულების შემდეგ ყველა ნაკადი უნდა სრულდებოდეს.

3. ჩათვალეთ რომ მთვლელი ნაკადების რაოდენობა ყოველთვის ნაკლებია ტექსტების რაოდენობაზე.

კოდი დაწერეთ counters.c ფაილში.

**ამოცანა 2. უნიკუმი (50 ქულა)**

თქვენი ამოცანაა ლისტში იპოვოთ ქველისტი, რომელიც ყველაზე გრძელი უნიკუმია. უნიკუმი ეწოდება ლისტს თუკი მასში ელემენტები არ მეორდება. მაგალითად ‘(1 2 3) ან ‘(4 3 5 2 1) დაწერეთ longestUniqum ფუნქცია, რომელსაც გადაეცემა ლისტი და აბრუნებს ლისტს. თუკი ლისტში რამდენიმე ერთიდაიგივე სიგრძის უნიკუმია მაშინ ფუნქციამ უნდა დააბრუნოს ყველაზე ბოლოს შემხვედრი უნიკუმი. ჩათვალეთ, რომ ფუნქციას მთელი რიცხვების ლისტი გადაეცემა, რომლის ელემენტების რაოდენობაც ნაკლებია 1000-ზე. ფუნქცია უნდა სრულებოდეს 1 წამში.

მაგალითები:

(longestUniqum ‘(1 2 3 4 2 3 4 1 2)) -> ‘(3 4 1 2)

(longestUniqum ‘(1 2 4 3 5 6 5 4 3)) -> ‘(1 2 4 3 5 6)

ამოხსნა დაწერეთ longestUniqum.scm ფაილში. კოდი უნდა დაწეროთ ფუნქციონალური პარადიგმის სრული დაცვით. არ გამოიყენოთ ბრძანება let.

დესკტოპზე იპოვით kawa.jar ფაილს, გახსენით და შეამოწმეთ თქვენი ამოხსნა სასურველ ტესტებზე.