***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 # 3***

|  |  |  |  |
| --- | --- | --- | --- |
| ***სკოლა*/*საგანმანათლებლო პროგრამა*** | მათემატიკა და კომპიუტერული მეცნიერება | ***სტუდენტის მიერ მიღებული ქულა*** |  |
| ***საგანი*** | პროგრამირების პარადიგმები | | |
| ***ლექტორი*** | შ. ღვინეფაძე | | |
| ***კურსი*** | I | | |
| ***ჯგუფი*** |  | | |
| ***გამოცდის ფორმა*** | ღია წიგნი | | |
| ***გამოცდის***  ***ხანგრძლივობა*** | 2 საათი | | |
| ***მაქსიმალური ქულა*** | 120 | | |
| ***სტუდენტის სახელი და გვარი:*** | | | |

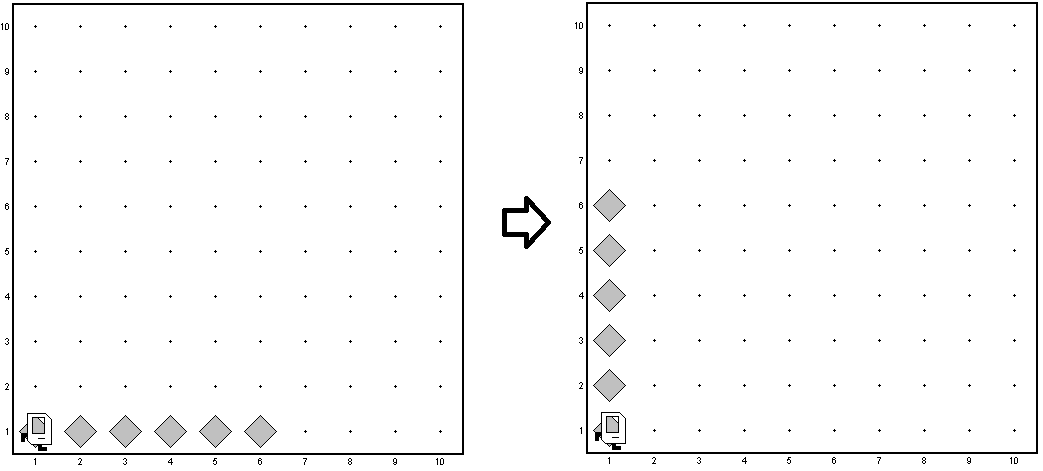
**ნაშრომის ჩაბარების წესი:**

**შექმენით დესკტოპზე თქვენი მეილის პრეფიქსის სახელის მქონე java გაფართოების ფაილი(მაგალითად gboch16.java). გახსენით ეს ფაილი notepad++-ის საშუალებით. ამისათვის შეგიძლიათ ფაილზე დააკლიკოთ მაუსის მარჯვენა ღილაკით და მონიშნოთ open with notepad++. ფაილის დასაწყისში დაწერეთ თქვენი სახელი და გვარი და ვარიანტის ნომერი. ყველა ამოცანის ამოხსნა ამ ფაილში უნდა დაწეროთ. გამოყავით ამოცანები ნომრებით 1,2,3...  
გამოცდიდან გასვლის წინ აუცილებლად გადაამოწმეთ რომ თქვენი ფაილი არის დესკტოპზე.**

**წარმატებები ;)**

**ამოცანა 1. კარელი (20 ქულა)**

მოცემული გაქვთ კარელის სამყარო სადაც პირველ სტრიქონში (1, 1) უჯრიდან (k, 1) უჯრამდე განლაგებულია ბრილიანტები, უჯრაში თითო. კარელი დგას მარცხენა ქვედა კუთხეში, სახით მარჯვნივ და ჩანთაში აქვს უსასრულო რაოდენობის ბრილიანტები. თქვენი ამოცანაა კარელს გადაალაგებინოთ ბრლიანტები პირველი ჰორიზონტალიდან, პირველ ვერტიკალში (1, 1) უჯრიდან (1, k) უჯრამდე, როგორც სურათზეა ნაჩვენები.



ჩათვალეთ რომ სამყაროს ყოველთვის აქვს ჰორიზონტალში არსებული ბრილიანტების ვერტიკალურად განლაგებისთვის საკმარისი სიმაღლე. კარელი სად იდგება მოქმედების შესრულების შემდეგ არ აქვს მნიშნელობა.

**ამოცანა 2. რას დაბეჭდავს (20 ქულა)**

დაწერეთ რას დაბეჭდავს შემდეგი პროგრამა. თუკი შეცდომა მოხდება კომპილაციის ან გაშვების შემდეგ მიუთითეთ რა სახის შეცდომაა.

**public** **class** Problem2 **extends** ConsoleProgram{

**public** **void** run(){

**int** x = 2;

**int** y = 4;

GOval o = **new** GOval(2,1); //width, height

println(racxa(x,2,o));

println(racxa(y,x/2+x,o));

}

**public** **int** racxa(**int** x, **int** y, GOval o){

String s = "hello world";

x = varesiRacxa(x/y, x+(**int**)o.getHeight(), s);

o.scale(2);

y = varesiRacxa(x, y+(**int**)o.getHeight(), s);

**return** x+y;

}

**public** **int** varesiRacxa(**double** x, **int** y, String s){

**int** z = (**int**)x + s.charAt(0)-s.charAt(1);

s = s.substring(1);

**return** z + y;

}

}

**ამოცანა 3. ფიბონაჩის მიმდევრობა (20 ქულა)**

თქვენი ამოცანაა დაადგინოთ არის თუ არა მომხმარებლის მიერ შემოყვანილი რიცხვები ფიბონაჩის მიმდევრობა. ამისათვის თქვენ უნდა დაწეროთ კონსოლის პროგრამა.

ჩათვალეთ რომ მომხმარებელს მხოლოდ მთელი რიცხვები შემოჰყავს მანამ სანამ არ შემოიყვანს -1. როგორც კი მომხმარებელი -1 შემოიყვანს თქვენმა პროგრამამ უნდა დაბეჭდოს სტრინგი “Fibonacci series” თუკი შეყვანილი მიმდევრობა ფიბონაჩის მიმდევრობაა და უნდა დაბეჭდოს სტრინგი “Not a Fibonacci series” წინააღმდეგ შემთხვევაში. რის შემდეგაც პროგრამამ უნდა დაასრულოს მუშაობა.

შეგახსენებთ რომ ფიბონაჩის მიმდევრობაა მთელი რიცხვების მიმდევრობა რომლის პირველი წევრია 0, მეორე 1, ხოლო ყოველი მომდევნო წევრი წინა ორის ჯამის ტოლია.

გაითვალისწინეთ ის ფაქტი, რომ მომხმარებელი მიმდევრობის შეყვანას პირველი წევრიდან იწყებს. ანუ, თუკი პირველივე წევრი 0-ისგან განსხვავებულია, მაშინ შეყვანილი მიმდევრობა არ არის ფიბონაჩის.

**public** **class** FibonacciSeries **extends** ConsoleProgram{

**public** **void** run (){

}

}

**ამოცანა 4. ტექსტის განკუმშვა (25 ქულა)**

მოცემული გაქვთ ტექსტის შეკუმშვის ალგორითმი.

თუკი ტექსტში რომელიმე სიმბოლო ზედიზედ ერთზე მეტჯერ გვხვდება ერთმანეთის გვერდით, მაშინ შეგვიძლია ეს ტექსტი შევამციროთ და ამ სიმბოლოების მაგივრად დავწეროთ თუ რამდენჯერ მეორდება და რა სიმბოლოა. მაგალითად, “aaaaa”-ს მაგივრად შეიძლება დავწეროთ “5a”.

თქვენი ამოცანაა დაწეროთ მეთოდი longestRunDecompression(), რომელსაც გადაეცემა სტრინგი და აბრუნებს სტრინგს. გადაცემული სტრინგი მოცემული ალგორითმით შეკუმშული ტექსტია. მეთოდმა უნდა დააბრუნოს ტექსტის თავდაპირველი ვარიანტი, ანუ როგორი იყო ტექსტი შეკუმშვამდე.

შეგიძლიათ ჩათვალოთ, რომ თავდაპირველ ტექსტში მხოლოდ პატარა ლათინური ასოები გვხვდებოდა და მასში მაქსიმუმ 9 ერთნაირი სიმბოლო იყო ერთმანეთის გვერდით.

მაგალითად

longestRunCompress(“2a4b2c”) -> “aabbbbcc”

longestRunCompress(“abbbbcccdee”) -> “a4b3cd2e”

**private** **String** longestRunDecompression(**String** text){

}

**ამოცანა 5. გამწვანება (35 ქულა)**

დაწერეთ გრაფიკული პროგრამა, რომელიც აკმაყოფილებს შემდეგ მოთხოვნებს:

* მაუსით ცარიელ ადგილზე კლიკის დროს უნდა გაჩნდეს CIRCLE\_D დიამეტრის მქონე წრე ცენტრით კლიკის წერტილში, რომელიც გაფერადებულია ნებისმიერი შემთხვევითი ფერით. ფერის შემთხვევითობა აუცილებელია რათა თამაში ფერადი გამოვიდეს.
* მაუსით რომელიმე წრეზე კლიკის დროს წრე უნდა აციმციმდეს ფერებით მწვანე, წითელი, ლურჯი, შავი და ყვითელი. ფერის შერჩევა უნდა ხდებოდეს შემთხვევით, მხოლოდ ამ 5 ფერიდან, ყოველ DELAY მილიწამში ერთხელ. ციმციმი უნდა გაგრძელდეს მანამ სანამ წრე არ გაფერადდება მწვანედ.

**public** **class** Green **extends** GraphicsProgram{

**private** **final** **static** **int** ***CIRCLE\_D*** = 40;

**private** **final** **static** **int** ***DELAY*** = 50;

**public** **void** run(){

}

}

**ამოცანა 6**. ბონუსი **(20 ქულა)**

თუკი ბონუსის მიღების სურვილით ხართ შეპყრობილი აირჩიეთ და დაწერეთ მთელი რიცხვი 0-დან 100-მდე შუალედში(0-ის და 100-ის ჩათვლით). გამოცდის შემდეგ დავითვლით თითოეული თქვენგანის მიერ არჩეული რიცხვების საშუალო არითმეტიკულს, გავამრავლებთ -ზე და ავიღებთ მიღებული რიცხვის მთელ ნაწილს. ვისი არჩეული რიცხვიც დაემთხვევა ჩვენს მიერ გამოთვლილ რიცხვს ის მიიღებს ბონუს 20 ქულას.