სახელი: **კურსი:** **ქულა:**

საბოლოო გამოცდა
25 იანვარი 10:00 – 13:00

ამოცანა 0. რთული **(1 ქულა)**
ღრმად ჩაისუნთქეთ და მაქსიმალურად მოეშვით... ახლა კი წარმატებები !!! ☺

ამოცანა 1. მოკლე პასუხები **(15 ქულა)**
გამოთვალეთ რას დაბეჭდავს შემდეგი პროგრამები:

```
public class Test extends ConsoleProgram{
	public void run(){
		String []str = new String[2];
		str[0]="gama";
		str[1]="rjoba";
		println(str[0]+str[1]);
		changeStrings(str);
		println(str[0]+str[1]);
	}
	private void changeStrings(String[] s) {
		s[0] = "gagi";
		s[1] = "marjos";
		println(s[0] + s[1]);
	}
}
```

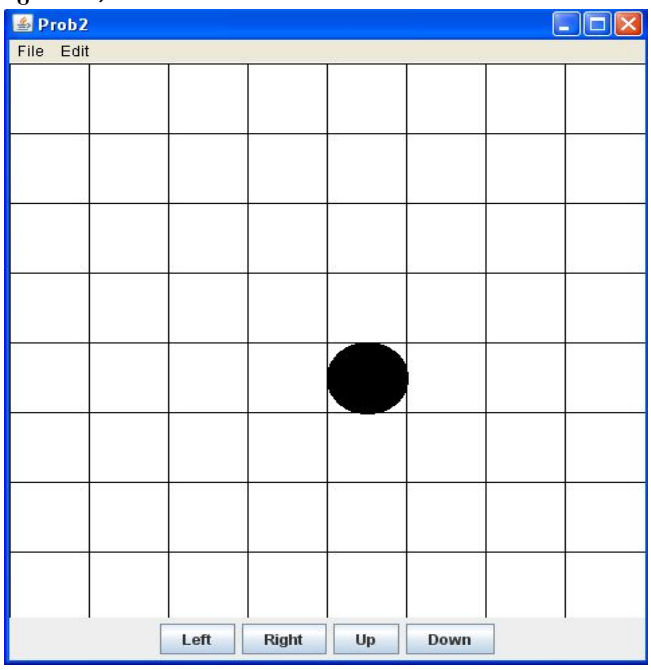
პასუხი:

```
public class Prob1b extends ConsoleProgram{
	public void run(){
		String str = "gamarjoba";
		println(str);
		changeStrings(str);
		println(str);
	}
	private void changeStrings(String s) {
		s = "gagimarjos";
		println(s);
	}
}
```

პასუხი:

ამოცანა 2. გრაფიკა **(35 ქულა)**

თქვენი ამოცანაა დაწეროთ პროგრამა რომელიც საშუალებას მოგვცემს გადაადგილოს ქვა დაფაზე. დაფა მოცემულია 8x8-ზე ბადის სახით, ხოლო ქვა კი უჯრაში ჩახატული წრეწირის სახით(იხ. სურათი)



თითოეული უჯრა არის კვადრატი რომლის ზომაც მოცემულია კონსტანტით SIZE(არ არის აუცილებელი დაფა მთელს ფანჯარას მოიცავდეს).
თქვენ რეალიზაცია უნდა გაუკეთოთ ქვის მოძრაობას. ქვა კი შეიძლება გადაადგილდეს თითო უჯრით მარჯვნივ, მარცხნივ, ქვემოთ ან ზემოთ. ქვის გადაადგილებას უზრუნველჰყოფენ ღილაკები, რომლებიც განლაგებულია კანვასის სამხრეთ(SOUTH) რეგიონში. “Left” ღილაკზე დაჭერის შემდეგ ქვა უნდა გადაადგილდეს მარცხნივ, “Right” ღილაკზე დაჭერის შემდეგ – მარჯვნივ, “Up” ღილაკზე დაჭერის შემდეგ ქვა უნდა გადაადგილდეს ზემოთ, “Down” ღილაკზე დაჭერის შემდეგ კი ქვემოთ. თავდაპირველად ქვა არის(5,4) კოორდინატებზე ისე როგორც სურათზეა გამოსახული. ყურადღება მიაქციეთ იმ ფაქტს, რომ თქვენმა პროგრამამ არ უნდა მისცეს მომხმარებელს ქვის დაფის გარეთ გადაგდების საშუალება.

ამოცანა 3. ანაგრამა **(24 ქულა)**

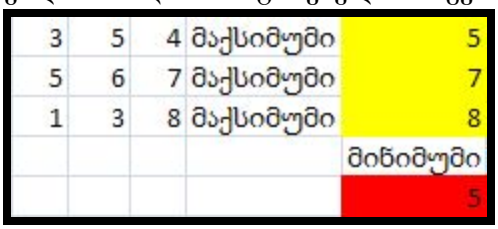
თქვენი ამოცანაა გაარკვიოთ არის თუ არა მოცემული ორი სიტყვა ერთმანეთის ანაგრამა. შეგახსენებთ რომ ანაგრამები ეწოდება სიტყვებს რომლებიც ერთიდაიგივე ასოების გადანაცვლებით ჩაიწერება. მაგალითად ერთერთი ყველაზე ცნობილი ანაგრამაა TOKYO და KYOTO.
თქვენ უნდა დაწეროთ მეთოდი isAnagram რომელსაც გადაეცემა ორი სტრინგ ტიპის ცვლადი და რომელიც აბრუნებს ture-ს თუ ეს სტრინგები ანაგრამებია და აბრუნებს false–ს წინააღმდეგ შემთხვევაში.

```
private boolean isAnagram (String str1, String str2)
```

ამოცანა 4. მინიმაქსი **(25 ქულა)**

მოცემული გაქვთ რიცხვების ცხრილი ორგანზომილებიანი მასივის სახით, თქვენი ამოცანაა გაარკვიოთ რომელია მეტი მინიმუმების მაქსიმუმი თუ მაქსიმუმების მინიმუმი. ამისათვის თქვენ უნდა დაითვალოთ თითოეულ სტრიქონის მინიმალური ელემენტი, ამ მინიმუმებს შორის ამოარჩიოთ

ყველაზე დიდი რიცხვი და შეინახოთ. შემდეგ კი უნდა ამოარჩიოთ თითოეული სტრიქონის მაქსიმალური ელემენტი, ამ მაქსიმუმებს შორის ამოარჩიოთ ყველაზე პატარა რიცხვი და შეადაროთ მანამდე შენახულ რიცხვს, თუკი მინიმუმების მაქსიმუმი(პირველად შენახული რიცხვი) მეტია მაქსიმუმების მინიმუმზე მაშინ პასუხად გამოგაქვთ 1, თუკი ნაკლებია მაშინ −1, ხოლო თუკი ეს რიცხვები ტოლია მაშინ 0. სურათზე უფრო თვალსაჩინოდ ჩანს ჩატარებული პროცესი:



ამ შემთხვევაში მინიმუმების მაქსიმუმი და მაქსიმუმების მინიმუმი ერთმანეთს დაემთხვა შესაბამისად ჩვენმა მეთოდმა უნდა დააბრუნოს 0.
გაითვალისწინეთ რომ ცხრილის სიგრძე და სიგანე ერთმანეთის ტოლი შეიძლება არ იყოს. ასევე გაითვალისწინეთ რომ ცხრილში მთელი რიცხვები წერია(შესაძლოა უარყოფითიც).
თქვენ უნდა დაწეროთ მეთოდი miniMax რომელსაც გადაეცემა ორგამზომილებიანი ინტების მასივი და რომელიც აბრუნებს 1,−1 ან 0−ს.

```
private int miniMax(int[][] a)
```

ამოცანა 5. მონაცემთა სტრუქტურები, სტეკი **(25 ქულა)**
პროგრამირებაში უამრავ საინტერესო მონაცემთა სტრუქტურას შეხვდებით ესენია მასივები, სიმრავლეები, ჰეშმეფები, რიგები და ა.შ. არანაკლებ საინტერესოა სტეკიც, რომელიც ერთერთი ყველაზე ხშირად გამოყენებადი სტრუქტურაა. თქვენ ალბათ გახსოვთ, რომ მეთოდების გამოძახების მიმდევრობა სწორედ სტეკით რეგულირდება(ე.წ. სტეკფრეიმები). თქვენი ამოცანაა სტეკის რეალიზაციის დაწერა.

შეგახსენებთ რომ სტეკის მოქმედების პრინციპი არის ,,პირველი მოვიდა ბოლო წავიდა“(first in last out). იგი მარტივი სტრუქტურაა და სულ სამი მეთოდი გააჩნია: push(), pop() და size(). მეთოდი push(a) ამატებს სტეკში მეთოდისთვის გადაცემულ ელემენტ a-ს. მეთოდი pop() შლის სტეკიდან და ამავდროულად აბრუნებს ბოლოში დამატებულ ელემენტს(ანუ ელემენტს რომელიც სულ ბოლოს დავამატეთ), ხოლო მეთოდი size() კი აბრუნებს სტეკში არსებული ელემენტების რაოდენობას.
თქვენ უნდა დაწეროთ StringStack კლასი, რომლის ობიექტიც მოახდენს სტეკის მოდელირებას სტრინგებისთვის. რაც იმას ნიშნავს, რომ მას ექნება ზემოთ აღწერილი სამი მეთოდი რომლებიც მოქმედებენ სტრინგებზე. ბევრად გაგიმარტივდებათ ამოცანა თუკი სტეკის ელემენტებს შეინახავთ სტრინგების არაილისტში(ArrayList<String>).
სტეკის გამოყენების მაგალითი:

```
StringStack stack = new StringStack();
stack.push(“pirveli”);      <-სტეკში დაემატა სტრინგი “pirveli”, ელემენტების რაოდენობა 1.
stack.push(“meore”);        <-დაემატა სტრინგი “meore”, ელემენტების რაოდენობა 2.
println(stack.pop());       <-დაბეჭდავს “meore”, სტეკში დარჩება მხოლოდ სტრინგი “pirveli”
println(stack.size());      <-დაბეჭდავს 1
stack.push(“mesame”);       <-სტეკში დაემატა სტრინგი “mesame”, ელემენდების რაოდენობა 2
```

და ა.შ.
საწყისი კოდი:

```
public class StringStack {
        public void push(String s){
        }

        public String pop(){
        }

        public int size(){
        }
}
```

ამოცანა 6. პოპულარული სიტყვა **(30 ქულა)**
ამჯერად თქვენი ამოცანაა ტექტში მოძებნოთ ყველაზე ხშირად შემხვედრი სიტყვა. თქვენ გაქვთ სტრინგი რომელიც შეიცავს დიდი მოცულობის ტექსტს(იგულისხმება რომ ტექსტში არ შეგხვდებათ არაფერი ასოებისა და პრაბელების გარდა). თქვენ უნდა დაწეროთ მეთოდი topWord–ის რეალიზაცია, რომელსაც გადაეცემა სტრინგ ტიპის ცვლადი და რომელიც აბრუნებს ყველაზე ხშირად შეხვედრილ სიტყვას(სტრინგ ტიპის ცვლადს).

```
private String topWord(String str)
```

ამოცანა 7. ჰეშმეფების გაერთიანება **(25 ქულა)**
ვინაიდან და რადგანაც, გარკვეული მიზეზების გამო დავალებების შემოწმება დაგვიანდა, გადავწყვიტეთ გამოვიყენოთ თქვენი დახმარება. საქმე იმაში მდომარეობს, რომ დავალების ქულებს ჩვენ ვინახავთ ჰეშმეფში(HashMap) რომლის გასაღებია(key) სტუდენტის გვარი, ხოლო მნიშვნელობა(value) კი სტუდენტის მიერ ამ დავალებაში დაგროვებული ქულა.
მაგალითად პირველი ორი დავალების ჰეშმეფები დაახლოებით ასე გამოიყურება:

assignment 1	
alavidze	3,07
alpenidze	3,17
bregadze	3,41

asignment 2	
alavidze	3,33
chakvetadze	0
bregadze	3,23
bochorishvili	-5,78

თქვენ უნდა დაგვეხმაროთ ამ მონაცემების გაერთიანებაში. ანუ უნდა შეადგინოთ ახალი ჰეშმეფი რომელიც იქნება ორი ჰეშმეფის გაერთიანება. გაითვალისწინეთ რომ ზოგიერთი(და არა ყველა) გასაღები(გვარი) შეიძლება ერთმანეთს დაემთხვეს. დამთხვევის შემთხვევაში ახალ ჰეშმეფში უნდა ჩაიწეროს მნიშვნელობების(ქულების) ჯამი. ამრიგად ზემოთ მოყვანილი მაგალითისთვის ჰეშმეფების გაერთიანება იქნება:

assignment 1+assignment 2	
alavidze	6,4
alpenidze	3,17
bregadze	6,64
chakvetadze	0
bochorishvili	-5,78

თქვენი მიზანია დაწეროთ მეთოდი hashMapUnited რომელსაც გადაეცემა ორი HashMap ტიპის ცვლადი და რომელიც აბრუნებს ასევე HashMap ტიპის ცვლადს – იმ ორი ჰეშმეფის გაერთიანებას. გაითვალისწინეთ რომ ყველა ჰეშმეფის გასაღები არის სტრინგ ტიპის ხოლო მნიშვნელობა კი დაბლი(Double და არა double!).
გამოიჩინეთ გულისხმიერება და დაგვეხმარეთ დავალებების გასწორებაში, წინასწარ დიდი მადლობა ☺

```
private HashMap<String, Double>
        hashMapUnited(HashMap<String, Double> map1, HashMap<String, Double>
map2)
```