სახელი: ქულა:

საბოლოო გამოცდა
30 იანვარი 10:00 – 13:00

| 1<br>10 ქულა | 2<br>25 ქულა | 3<br>40 ქულა | 4<br>35 ქულა | 5<br>70 ქულა | სულ |
|---|---|---|---|---|---|
| | | | | | |

**ამოცანა 1.** ყუელაზე პროგრამა **(10 ქულა)**

გამოთვალეთ რას დაბეჭდავს შემდეგი პროგრამა. თუკი შეცდომა მოხდება პროგრამის შესრულების დროს მიუთითეთ შეცდომის ხასიათი.

```
public class Problem extends ConsoleProgram {
	public void run(){
		println(racxa(x,2));
		println(racxa(y,x/2+x));
	}

	public int racxa(int x, int y){
		x = varesiRacxa(x/y, x+y);
		y = varesiRacxa(x, y);
		return x+y;
	}

	public int varesiRacxa(int x, int y){
		int z = x + 'z'-'w';
		return z+this.x;
	}

	private int x = 1;
	private int y = 2;
}
```

პასუხი:

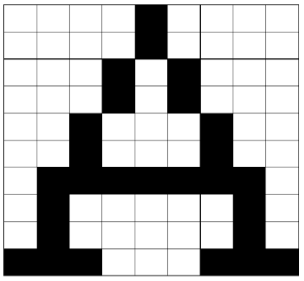
**ამოცანა 2.** ყუელაზე სიტყვა **(25 ქულა)**

თქვენი ამოცანაა დაწეროთ მეთოდი რომელსაც გადაეცემა ტექსტი და რომელიც აბრუნებს ამ ტექსტში არსებულ ყველაზე გრძელ სიტყვას, ოღონდ ისეთ სიტყვას რომელიც ერთნაირ ასოებს არ შეიცავს, ანუ რომლის ყველა ასო ერთმანეთისგან განსხვავებულია.
ჩათვალეთ რომ ტექსტში გვხვდება მხოლოდ პატარა ასოები, სფეისები და წერტილები.
თუკი ტექსტში ორი ერთიდაიგივე სიგრძის სიტყვაა რომელიც მოცემულ პირობას აკმაყოფილებს მაშინ დააბრუნეთ ერთერთი.
მაგალითად თუკი ჩვენს მეთოდს გადავცემთ ამ ამოცანის ტექსტს მაშინ მან უნდა დააბრუნოს სიტყვა – რომელსაც.

**private String getWord(String text){**

**ამოცანა 3.** ყველაზე სურათებეე **(40 ქულა)**
თქვენი ამოცანაა დაწეროთ მეთოდი რომელსაც შეუძლია სურათების შეერთება. იმისათვის რომ საქმე გავიმარტივოთ ჩავთვალოთ რომ მხოლოდ შავთეთრი სურათები გვაქვს რომლებიც მოცემულია

boolean-ების მატრიცის სახით. მატრიცაში true ნიშნავს რომ იმ კოორდინატებზე პიქსელი გაფერადებულია შავად, false კი ნიშნავს რომ გაფერადებული არ არის. იხილეთ მაგალითი:



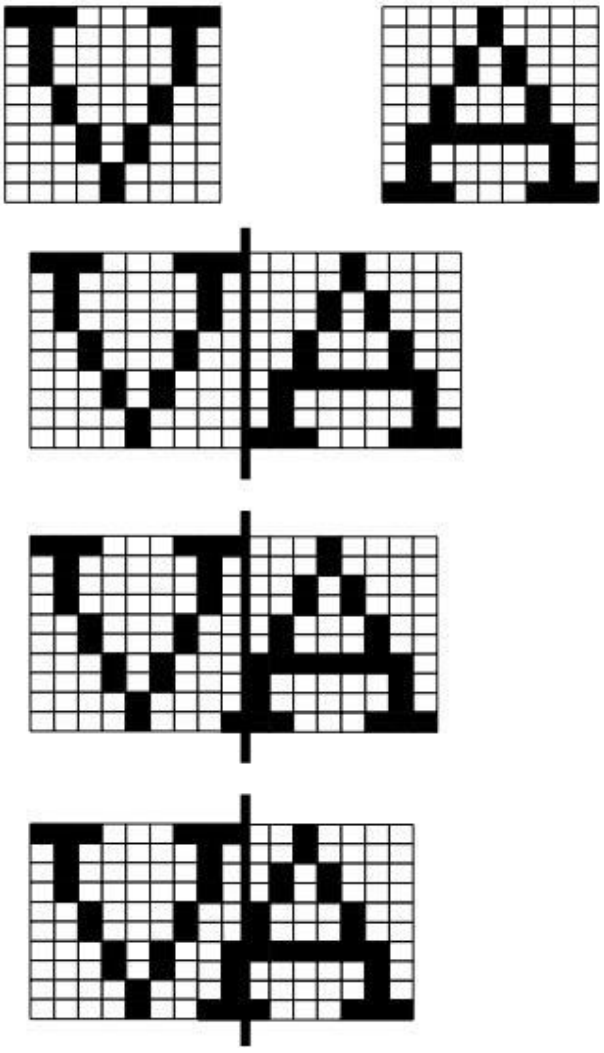
```
{
 { false, false, false, false, true,  false, false, false, false },
 { false, false, false, false, true,  false, false, false, false },
 { false, false, false, true,  false, true,  false, false, false },
 { false, false, false, true,  false, true,  false, false, false },
 { false, false, true,  false, false, false, true,  false, false },
 { false, false, true,  false, false, false, true,  false, false },
 { false, true,  true,  true,  true,  true,  true,  true,  false },
 { false, true,  false, false, false, false, false, true,  false },
 { false, true,  false, false, false, false, false, true,  false },
 { true,  true,  true,  false, false, false, true,  true,  true  }
}
```

თქვენ მოცემული გაქვთ ორი სურათი, უნდა შეძლოთ მათი შეერთება. გასაგებია რომ სურათის აღსაქმელად საკმარისია შავი პიქსელები და მათ ირგვლივ რამხელა თეთრი სივრცე იქნება დიდი მნიშვნელობა არ აქვს. თქვენი მიზანია შეაერთოდ სურათები ისე რომ მთლიანმა სურათმა მაქსიმალურად პატარა ზომა დაიკავოს და თან სხვადასხვა სურათების შავი პიქსელები ერთმანეთს არ შეეხონ და სურათები გარჩევადი დარჩეს.

ქვემოთ იხილეთ მაგალითი სადაც მოცემულია ორი სიმბოლოს სურათი. მეორე ხაზზე ეს ორი სიმბოლო უბრალოდ მიდგმულია ერთმანეთზე. ამ დროს მათ შორის საკმარისი თეთრი ადგილია და მათი ერთმანეთზე მიახლოება არის შესაძლებელი. მესამე და მეოთხე სურათებზე თითო-თითო პიქსელით არის მიახლოებული ეს სურათები ერთმანეთს. თქვენ მაქსიმალურად უნდა მიახლოვოთ სიმბოლოები ისე რომ განსხვავებული სურათების პიქსელები ერთმანეთს არ შეეხონ რითაც მთლიანი კომპოზიცია დაირღვევა. ანუ სხვადასხვა სურათების შავ პიქსელებს შორის მინიმუმ ერთი თეთრი პიქსელი მაინც უნდა იყოს.
დაწერეთ pictureUnion მეთოდი რომელსაც გადაეცემა ორი ცალი boolean-ების მატრიცა - სურათები და რომელიც აბრუნებს boolean-ების მატრიცას - გაერთიანებულ სურათს

ჩათვალეთ რომ გადმოცემული სურათების სიმაღლე ერთმანეთის ტოლია, დაბრუნებული სურათის სიმაღლეც იგივე უნდა იყოს. ასევე ჩათვალეთ რომ სურათებზე პიქსელები ისეა განლაგებული რომ პირდაპირ თუკი მივადგამთ ერთმანეთს მაშინ შავი პიქსელები ერთმანეთს არ შეეხება.

**private boolean[][] pictureUnion(boolean [][] p1, boolean [][] p2){**

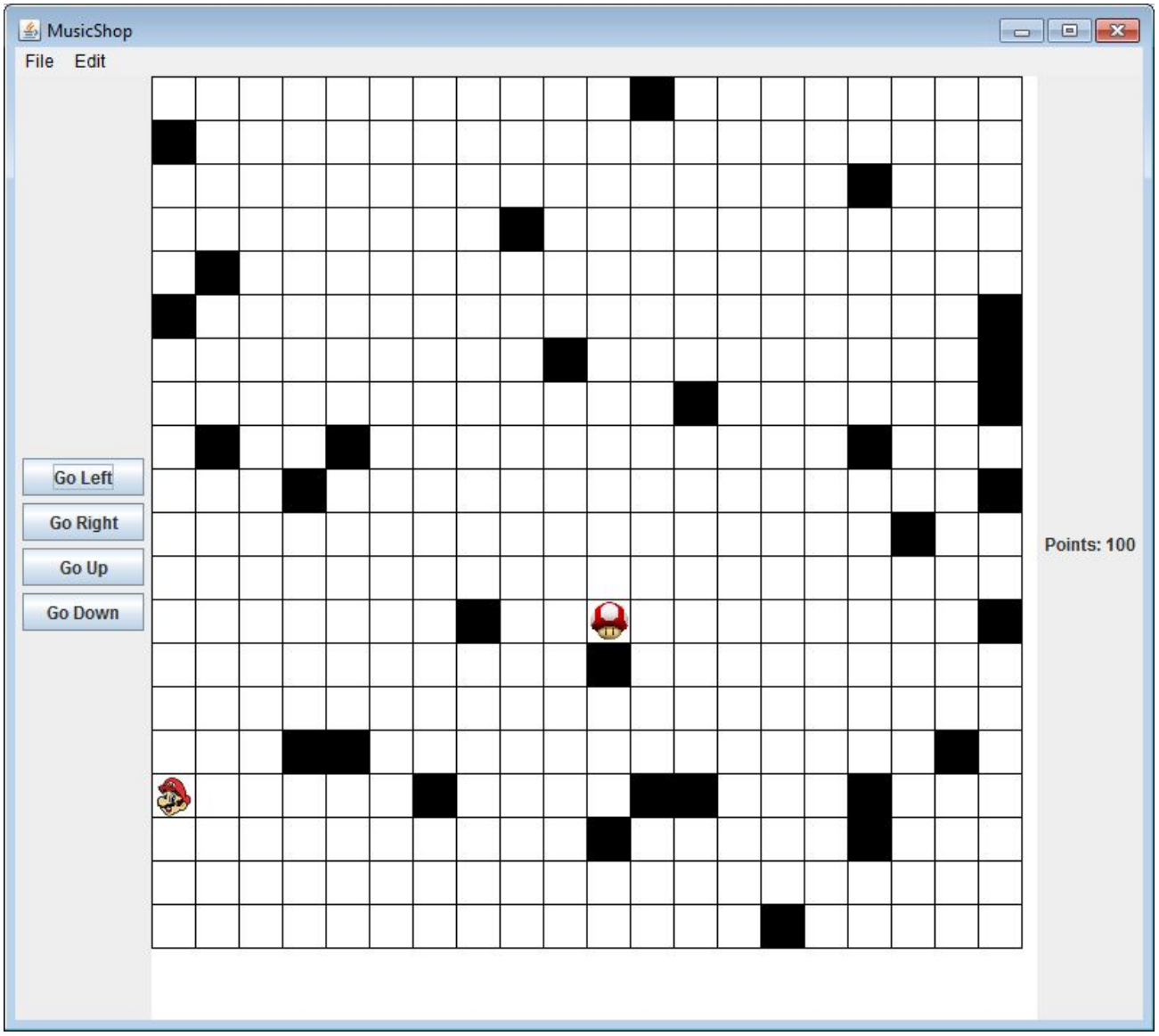


**ამოცანა 4.** ყუელაზე ჰიფსტერებეე**(35 ქულა)**

მოცემული გაქვთ ფეისბუქის მსგავსი სოციალური ქსელი, რომლის თითოეულ წევრს რაღაც ბენდები მოსწონს. ჰიფსტერი დავარქვათ ადამიანს თუკი მას მოსწონს ისეთი ბენდი რომელიც მის მეგობრებში არავის არ მოსწონს. თქვენი მიზანია დაწეროთ პროგრამა რომელიც გამოავლენს ყველა ჰიფსტერს.
სოციალური ქსელის გრაფი მოცემულია ჰეშმეფის(HashMap) საშუალებით ისევე როგორც მეშვიდე დავალებაში იყო. მეფის(Map) გასაღები(key) არის სოციალურ ქსელში არსებული ადამიანის სახელი, ხოლო მნიშვნელობა(value) კი ამ ადამიანის მეგობრების ლისტი.
ე.წ. მეგობრობის გრაფის გარდა თქვენ მოცემული გაქვთ ინფორმაცია თუ ვის რა ბენდები მოსწონს. ესეც მეფის სახით არის მოცემული სადაც გასაღები არის ადამიანის სახელი, ხოლო მნშივენლობა კი ბენდების ლისტი.
თქვენმა პროგრამამ უნდა დააბრუნოს ადამიანების ლისტი რომელშიც იქნება ყველა ჰიფსტერი.
ჩათვალეთ რომ ადამიანის სახელები უნიკალური იდენტიფიკატორებია.

```
private ArrayList<String> getHipsters(HashMap<String, ArrayList<String>> profiles,
                                      HashMap<String, ArrayList<String>> bands){
```

**ამოცანა 5.** ყუელაზე მარიო **(70 ქულა)**
თქვენი ამოცანაა დაწეროთ მარიოს ანალოგიური თამაში. სურათზე ნაჩენებია თუ როგორ უნდა გამოიყურებოდეს პროგრამა.




მოთხოვნები:

**1.** პროგრამამ board.txt ფაილიდან უნდა წაიკითხოს დაფის აღწერა და კანვასზე დახატოს შესაბამისი სურათი. ფაილის პირველ ხაზში მოცემულია დაფის სიგრძე და სიგანე(უჯრების რაოდენობა) სფეისით გამოყოფილი. მეორე ხაზში მოცემულია მარიოს მდებარეობა სვეტის და სტრიქნის ნომრით, მესამე ხაზში

მოცემულია სოკოს ადგილმდებარეობა, დანარჩენ ხაზებში მოცემულია უჯრების ჩამონათვალი სადაც კედელი მდებარეობს. იხილეთ ზემოთ ნაჩვენები სურათის შესაბამისი ფაილი:

20 20
0 16
10 12
0 1
0 5
1 4
1 8
3 9
3 15
4 8
4 15
6 16
7 12
8 3
9 6
10 13
10 17
11 0
11 16
12 7
12 16
14 19
16 2
16 8
16 16
16 17
17 10
18 15
19 5
19 6
19 7
19 9
19 12

ჩათვალეთ რომ სტრიქონების და სვეტების გადანომრვა იწყება ზედა მარცხენა კუთხიდან. თითოეული უჯრა კვადრატია და მისი ზომა შეგიძლიათ გამოთვალოთ კანვასის ზომის საშუალებით. დაფა მაქსიმალურად უნდა ფარავდეს კანვასს და ცხადია არც ერთი უჯრა არ უნდა გაცდეს საზღვრებს. მარიოს და სოკოს სურათები მოცემული გაქვთ Mario.png და Mashroom.png ფაილების საშუალებით.

2. პროგრამის მარცხენა რეგიონში განთავსებული უნდა იყოს მარიოს ნავიგაციის ღილაკები. ღილაკზე დაჭეროსას მარიო გადადის გვერდითა უჯრაში შესაბამისი მიმართულებით. თუ გვერდითა უჯრა კედელია, მარიო რჩება უმოქმედოდ. მარიო არ უნდა გაცდეს დაფის საზღვრებს.

3. თუ მარიო ხვდება უჯრაზე სადაც მდებარეობს სოკო, იგი იღებს სოკოს და ემეტაბა 100 ქულა. ჯამური ქულა ნაჩვენები უნდა იყოს პროგრამის მარჯვენა რეგიონში, ისე როგორც სურათზეა ასახული. სოკოს აღებისთანავე დაფაზე ერთერთ თავისუფალ უჯრაზე უნდა გაჩნდეს ახალი სოკო, უჯრის შერჩევა უნდა მოხდეს შემთხვევითობის პრინციპით(გამოიყენეთ RandomGenerator).

4. აუცილებელია რომ ფანჯრის ზომის ცვლილებისას დაფა გადაიხატოს ისე რომ ის ისევ მთელ კანვასს ფარავდეს.