Polsemestrálny test
Zadanie

Ústav informatiky
Prírodovedecká fakulta
UPJŠ v Košiciach

Dvakrát meraj (rozmýšľaj), raz rež (programuj)

Pravidlá a informácie:

- čas na riešenie úloh je **80 minút**, resp. do 11:20,
- nie je dovolená žiadna (elektronická aj neelektronická) komunikácia s kýmkoľvek okrem dozoru,
- nie je dovolené používať žiadne zdroje ani materiály okrem oficiálneho ťaháku a predmetového webu vrátane gitlabu,
- nie je dovolené používať žiadnu inú aplikáciu než Eclipse (s výnimkou webového prehliadača pri odosielaní riešenia),
- porušenie pravidiel má za následok hodnotenie FX,
- svoje riešenia odovzdávajte cez systém Moodle (http://lms.ics.upjs.sk/).

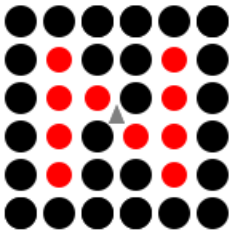
Ktoré úlohy treba riešiť:

V ***Časti 1*** je cieľom úloh vytvoriť triedu `Midtermarka`, ktorá rozširuje triedu `Turtle`. Z prvej trojice úloh si **vyberte len 2 úlohy**, ktoré **budete riešiť!!!** To, ktoré úlohy ste sa rozhodli riešiť, uveďte v komentári pri odosielaní riešenia cez Moodle (ak to nie je zrejmé z odoslaného).

V ***Časti 2*** je len jedna úloha, t.j. v tejto časti nie je možný výber úloh.

Časť 1 (dve úlohy z troch)

NetVOD – korytnačia stremovacia služba (10 bodov)



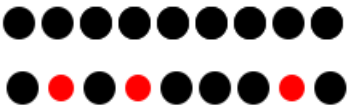
Video On Demand sa stalo populárne aj medzi korytnačkami. Preto sa rozhodli založiť si streamovaciu službu Turtle NetVOD. Nenechávajú veci na náhodu a chcú preraziť nie len kvalitnými službami ale aj na poli marketingu, virálnym logom. Aby každá korytnačka vedela kresliť logo, tak prišli za tebou ako top študentom nech ich to naučíš ;)

Do triedy `Midtermarka` pridajte metódu `turtleNetVOD`, ktorá nakreslí vzor ako na obrázku. Metóda má dva parametre `pocet, radius`. Vzor je tvorený `pocet` krát `pocet` kruhmi. Na obrázkoch je korytnačka zobrazená sivou farbou len pre jej lepšiu viditeľnosť. Prvý obrázok má parametre 6, 10, druhý obrázok má parametre 7, 10 a tretí obrázok má parametre 12, 5. Čierne kruhy majú polomer `radius` a červené kruhy majú polomer `0,8 * radius`. Červené kruhy tvoria písmeno N. Stredy dvoch susedných kruhov sú od seba vzdialené o `2,4 * radius`. Korytnačka sa na začiatku nachádza v strede obrazca. Obrazec je nakreslený v smere natočenia korytnačky. Po vykonaní metódy nech je korytnačka na pozícii a v smere, ako bola pred vykonaním metódy.

```
public void turtleNetVOD(int pocet, double radius)
```

Rada:

Odporúčame si vytvoriť pomocnú metódu, ktorá nakreslí jeden riadok. Pričom metóda má parametre `pocet, radius, cisloRiadku`.



Na druhej strane papiera nájdete oficiálny ťahák.

Collatz conjecture (10 bodov)

Do triedy `Midtermarka` pridajte metódu `collatz`. Táto metóda dostane ako parameter kladné celé číslo `n`. Ak je číslo párne tak ho vydelí dvoma, ak je nepárne, tak ho vynásobí troma a pripočíta k nemu jedna. S novým číslom pokračuje rovnako ďalej, pokiaľ z neho nebude 1. Metóda vráti koľko takýchto „operácii“ sme vykonali, kým sme dostali 1. Príklady:

`collatz(8) = 3`, lebo `8/2=4, 4/2=2, 2/2=1`, teda sme vykonali 3 „operácie“,
`collatz(3) = 7`, lebo `3*3+1=10, 10/2=5, 5*3+1=16, 16/2=8, 8/2=4, 4/2=2, 2/2=1`, teda sme vykonali 7 „operácii“,
`collatz(106) = 12`, lebo `106/2=53, 53*3+1=160, 160/2=80, 80/2=40, 40/2=20, 20/2=10, 10/2=5, 5*3+1=16, 16/2=8, 8/2=4, 4/2=2, 2/2=1`, čo je 12 „operácii“.

Pozn.: Za riešenie funkčné iba na malých vstupoch bude udelených 8 bodov.

```
public int collatz(int n)
```

Podobna štruktúra (10 bodov)

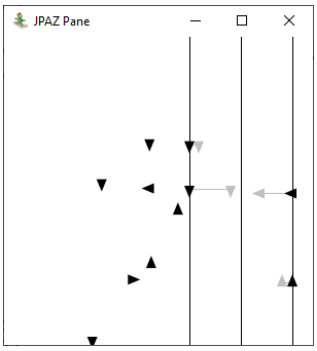
Do triedy `Midtermarka` pridajte metódu `similarStructure`. Táto metóda dostane ako parametre dva ne`null`ové referencie na reťazce (objekty triedy String) a vráti `true` alebo `false` podľa toho, či je štruktúra oboch reťazcov podobná. Povieme, že štruktúra dvoch reťazcov je podobná ak postupnosť dĺžok úsekov tvorených rovnakými znakmi je totožná.
Príklady (úseky sa striedajú s podčiarknutím a bez podčiarknutia):

`similarStructure("aab", "xxy") = true`, lebo oba reťazce majú úseky dĺžok 2, 1,
`similarStructure("aaBaaa8cc", "xxyUUU8aa") = true`, dĺžky úsekov 2, 1, 3, 1, 2,
`similarStructure("aaa", "CCT") = false`, lebo prvý reťazec má úsek 3 a druhý 2, 1,
`similarStructure("PAZ1a", "ALG") = false`, lebo 1, 1, 1, 1, 1 nie je 1, 1, 1.

```
public boolean similarStructure(String s, String t)
```

Časť 2

Na kraj (10 bodov)



Obrázok ku 7 bodovej časti

- *(3 body)* Vytvorte triedu `MidtermPane`, ktorá rozširuje triedu `WinPane`. Po vytvorení kresliacej plochy triedy `MidtermPane` nech sa v nej (automaticky v konštruktore, resp. „inicializačnej metóde“) vytvorí 11 korytnačiek triedy `Turtle` na náhodných pozíciách vo viditeľnej časti kresliacej plochy a majú natočenia 0, 90, 180 alebo 270 stupňov. **Pozn.:** Ak riešite iba túto časť tak pridajte prázdnu metódu zo 7 bodovej úlohy kvôli evaluácii.
- *(7 bodov)* Do triedy `MidtermPane` pridajte metódu `outTheWay`. Táto metóda má jeden parameter `x`. Navyše táto metóda vykoná nasledovné. Korytnačky, ktorých vzdialenosť od zvislej priamky na súradnici `x` je menej ako 50, sa posunú tak, aby ich vzdialenosť od priamky bola 50 a zároveň prešli, čo najkratšiu vzdialenosť. Ostatné korytnačky nevykonávajú nič. Metóda vráti súčet prejdených vzdialeností. Korytnačky, ktoré sa presunú majú mať na konci natočenie ako pred posunom.
  Pozn.: Čiary ako na obrázku nekreslite, sú iba ilustračné a vždy sa nachádzajú vo viditeľnej časti plochy. Sivou farbou sú znázornené pôvodné polohy korytnačiek a ich pohyb, toto netreba kresliť. Pri rozhodovaní, či korytnačka ide „doprava“ alebo „doľava“ môžete predpokladať, že žiadna korytnačka sa nenachádza na priamke. Môžete predpokladať, že všetky korytnačky sa presunú vo viditeľnej časti plochy.

  ```
  public double outTheWay(double x)
  ```

Oficiálny ťahák
Polsemestrálny test

Ústav informatiky
Prírodovedecká fakulta
UPJŠ v Košiciach

Dvakrát meraj (rozmýšľaj), raz rež (programuj)

Základné metódy objektov triedy String:

`int length()`
- vráti dĺžku reťazca

`char charAt(int index)`
- vráti znak na zadanom indexe v reťazci (znaky sú indexované od 0)

`boolean equals(String r)`
- vráti *true* práve vtedy, keď tento reťazec sa skladá z tej istej postupnosti znakov ako reťazec referencovaný parametrom `r`

`String trim()`
- vráti referenciu na novovytvorený reťazec vytvorený odstránením počiatočných a koncových medzier

`String toLowerCase()` resp. `String toUpperCase()`
- vráti referenciu na novovytvorený reťazec po zmene znakov v reťazci na malé (veľké) písmena

`String substring(int zacIndex, int konIndex)`
- vráti referenciu na novovytvorený reťazec obsahujúci podreťazec tvorený znakmi na indexoch `zacIndex` (vrátane) až `konIndex` (nie je zahrnutý)

`int indexOf(String podretazec)` resp. `int indexOf(char znak)`
- vráti index prvého výskytu podreťazca resp. znaku v reťazci. Ak sa v reťazci nenachádza vráti -1

Základné metódy objektov triedy Turtle:

`void center()`
- presunie korytnačku do stredu plochy, v ktorej sa nachádza (korytačka musí byť v ploche)

`void setPosition(double x, double y)`
- presunie korytnačku na pozíciu so súradnicami [x, y], čiara sa nekreslí

`void step(double dlzka)`
- spraví krok v smere natočenia zadanej dĺžky, čiara sa kreslí v závislosti od stavu kresliaceho pera

`void turn(double uhol)`
- otočí korytnačku o zadaný uhol v smere hodinových ručičiek

`void moveTo(double x, double y)`
- korytačka spraví krok do bodu na súradniciach [x, y], čiara v závislosti od kresliaceho pera

`void setDirection(double smer)`
- natočí korytnačku zadaným smerom (smer 0 je nahor, 90 doprava, atď.)

`double getDirection()`
- vráti smer aktuálneho natočenia korytnačky

`void turnTowards(double x, double y)`
- natočí korytnačku tak, aby bola natočená smerom k bodu na súradniciach [x, y]

`double directionTowards(double x, double y)`
- vráti smer, pri ktorom by bola korytnačka natočená smerom k bodu na súradniciach [x, y]

`double distanceTo(double x, double y)`
- vráti vzdialenosť korytnačky k bodu na súradniciach [x, y]

`void dot(double polomer)`
- nakreslí vyplnený kruh (farbou výplne) so zadaným polomerom a stredom v pozícii korytnačky

`void setFillColor(Color farba)`
- nastaví farbu výplne

`void setPenColor(Color farba)`
- nastaví farbu kresliaceho pera

`void penDown()` resp. `void penUp()`
- zapne resp. vypne kresliace pero

`void openPolygon()`
- zapne sledovanie ohraničovania oblasti ktorá bude vyplnená pri `void closePolygon()`

Základné metódy objektov triedy WinPane (kresliaca plocha):

`void add(Turtle korytnacka)`
- pridá (referencovanú) korytnačku do kresliacej plochy

`void remove(Turtle korytnacka)`
- odoberie (referencovanú) korytnačku z kresliacej plochy

`int getWidth()` resp. `int getHeight()`
- vráti šírku, resp. výšku kresliacej plochy

Java a polia

- prechod všetkými indexami poľa referencovaného z premennej *pole*:

```
for (int i=0; i<pole.length; i++) { … }
```

JPAZ a myšacie udalosti

```
protected void onMouseClicked(int x, int y, MouseEvent detail) {
    if ((detail.getButton() == MouseEvent.BUTTON1) &&
              detail.isControlDown()) {
          // pri zatlačení ľavého tlačidla myši
          // vo chvíli, keď je zatlačený aj Ctrl
    }
}
```

Farby

`Color.red, Color.blue, Color.green, Color.gray, Color.black` ... alebo
`new Color(int r, int g, int b)`, kde `r`, `g` a `b` sú celé čísla od 0 po 255.

Náhodné číslo

Vygenerovanie náhodného čísla z intervalu <0, a): `Math.random()*a`
Vygenerovanie náhodného celého čísla od 0 po n: `(int)(Math.random()*(n+1))`

Vytvorenie poľa

Vytvorenie poľa 6 celých čísel:
```
int[] pole = new int[6];
```

Vytvorenie poľa 6 celých čísel s inicializáciou hodnôt:
```
int[] pole = {3, 4, 6, 1, 2, 4};
```

Výpis poľa: `System.out.println(Arrays.toString(pole));`
Kopírovanie prvkov poľa:
```
System.arraycopy(odkiaľ, odAkéhoIndexu, kam, odAkéhoIndexu, koľkoPolíčok);
```

Čísla

`Double.MAX_VALUE` - najväčšie číslo, ktoré možno uložiť v premennej typu double
`Double.POSITIVE_INFINITY` – +∞
`double cislo = Double.parseDouble("3.14");` - prevedie reťazec na číslo
`Math.sqrt(c)` – vyráta odmocninu zadaného čísla `c`

Znaky

`Character.isUpperCase(z)` – vráti, či znak `z` predstavuje veľké písmeno
`Character.toUpperCase(z)` – vráti znak, ktorý vznikne zo `z` zmenou na veľké písmeno