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,
- 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)

Window shader (5 bodov)

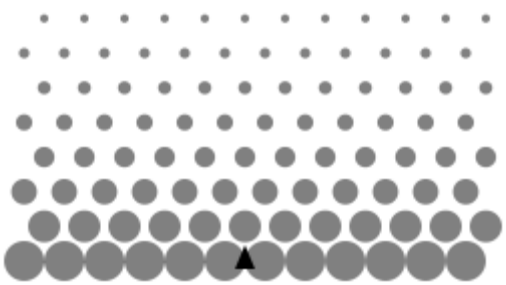
Keď Žofka išla autobusom do Košíc všimla si, že na autobus ma na okraji okna vzor ktorý má tlmiť kontrast. Uvedomila si, že niečo také vie pomocou korytnačej grafiky nakresliť aj sama,

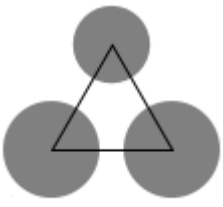
Do triedy `Midtermarka` pridajte metódu `windowShader`, ktorá nakreslí vzor ako na obrázku vpravo. Metóda má tri parametre `vyska, sirka, radius`. Parametre `vyska, sirka` určujú rozmery vzoru (na hornom obrázku vpravo má obrazec výšku 8 a šírku 5, na dolnom obrázku vpravo má obrázec výšku 6 a šírku 4). Obrazec je tvorený šedivými kruhmi (farba gray) pričom najväčšie kruhy majú polomer `radius` a každý ďalší riadok má kruhy veľkosti 80% oproti veľkosti predchádzajúceho riadku. Stredy troch susedných kruhov tvoria rovnostranný trojuholník so stranami veľkosti `radius`. Korytnačka sa na začiatku nachádza v strede spodného riadku. 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 windowShader(int vyska, int sirka, double radius)
```

Rada:

Odporúčame si vytvoriť pomocnú metódu ktorá nakreslí jeden riadok so zadaným počtom bodiek, ich veľkosťami a ich rozstupmi.





Počet výskytov najväčšej cifry (5 bodov + 2 bonusové body)

Do triedy `Midtermarka` pridajte metódu `countOccurencesOfMaxDigit`. Táto metóda dostane ako parameter nezáporné celé číslo `n` a vráti koľkokrát sa v čísle `n` nachádza jeho najväčšia cifra.

Príklady, najväčšia cifra je zobrazená hrubým písmom:

```
countOccurencesOfMaxDigit(12345) = 1
countOccurencesOfMaxDigit(16263646) = 4
countOccurencesOfMaxDigit(101101110) = 6
countOccurencesOfMaxDigit(0) = 1
```

```
public int countOccurencesOfMaxDigit(int n)
```

Pozn: Za riešenie s použitím triedy `String` udelíme maximálne 1 bod.

Pozn. pre fajnšmekrov: V závislosti od efektivity riešenia je za túto úlohu možné získať 0-2 bonusových bodov do kategórie bonusové body.

Odčísluj reťazec (5 bodov)

Do triedy Midtermarka pridajte metódu `odCislovac`. Táto metóda dostane ako parametre dve ne`null`ové referencie na reťazce (objekty triedy String) rovnakej dĺžky a vráti referenciu na novovytvorený reťazec (objekt triedy String), ktorý vznikne z prvého zadaného reťazca zámenou všetkých čísel za znaky z druhého reťazca. Pričom zamieňame znaky na rovnakých pozíciách.

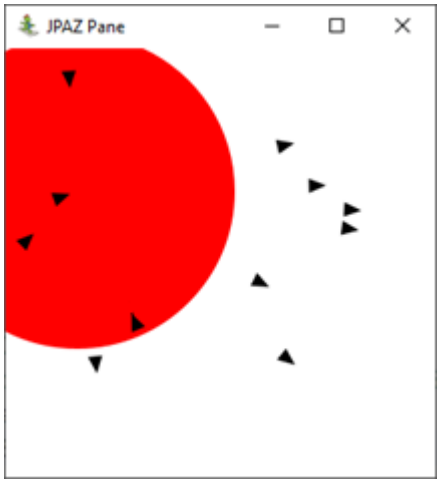
Príklady, nahrádzané znaky sú zobrazené hrubým písmom:

```
odCislovac("a5h","gci") = "ach"
odCislovac("246","abc") = "abc"
odCislovac("ahoj svet","qwertyuio") = "ahoj svet"
odCislovac("1mn4op78","ABCDEFGH") = "AmnDopGH"
```

```
public String odCislovac(String s, String nahrada)
```

Časť 2

Ďaleko blízko (5 bodov)



Obrázok ku 4 bodovej časti

- *(1 bod)* 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 náhodné natočenia. **Pozn.:** Ak riešite iba túto časť tak pridajte prázdnu metódu zo 4 bodovej úlohy kvôli evaluácii.

- *(4 body)* Do triedy `MidtermPane`, pridajte metódu, `dalekoBlizko`. Táto metóda má tri parametre `x`, `y`, `d`. Navyše táto metóda vykoná nasledovné: 1. Korytnačky, ktorých vzdialenosť od bodu na súradniciach `x`, `y` je menej ako `d` otočí k bodu `x`, `y`. Ostatné korytnačky otočí od bodu `x`, `y`. 2. Metóda vráti pole korytnačiek ktorých vzdialenosť od bodu na súradniciach `x`, `y` je menej ako `d`.
  Pozn.: Červený kruh na obrázku nekreslite. Je iba ilustračn˘y a reprezentujú vzdialenosť `d` od `x`, `y`

```
public double dalekoBlizko(double x, double y, double d)
```

Na druhej strane nájdete oficiálny ťahák.

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