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 9:30,
- nie je dovolená žiadna (elektronická aj neelektronická) komunikácia s kýmkoľvek okrem dozoru,
- nie je dovolené používať žiadne zdroje ani materiály okrem oficiálneho ťaháku,
- nie je dovolené používať žiadnu inú aplikáciu než Eclipse (s výnimkou webového prehliadača pri odosielaní riešenia), monitorovací softvér musí byť spustený počas celého testu,
- 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)

Obdĺžnik s kvázi uhlopriečkou (10 bodov)

Do triedy `Midtermarka` pridajte metódu `obrazec`, ktorá nakreslí obrazec z kruhov ako na obrázku vpravo. Metóda má dva parametre `vyska` a `sirka`. Parametre `vyska` a `sirka` určujú rozmery obrazca (na obrázku vpravo má horný obrazec parametre `vyska` a `sirka` 4 a 6, stredný obrazec 5 a 3, dolný obrazec 1 a1). Obrazec je tvorený kruhmi s polomerom 10, pričom kruhy nad „kvázi uhlopriečkou“ obrazca sú farby ORANGE a ostatné kruhy sú farby MAGENTA. Korytnačka sa na začiatku nachádza v strede obrazca a natočená je v smere výšky obrazca. Po vykonaní metódy nech je korytnačka na pozícii a v smere, ako bola pred vykonaním metódy.

```
public void obrazec(int vyska, int sirka)
```

Rady:

- Odporúčame si vytvoriť metódu, ktorá nakreslí jeden riadok obrazca skladajúce sa z `n` kruhov, pričom `k` vyjadruje index prvého oranžového kruhu.

```
public void riadok(int n, int k)
```

Číselní kamaráti (10 bodov)

Do triedy `Midtermarka` pridajte metódu `friendlyNumbers`, ktorá pre zadané prirodzené čísla n, m $(n, m \geq 1)$ vráti či sú priateľské. Čísla n, m nazývame priateľské ak $\frac{\sigma(n)}{n} = \frac{\sigma(m)}{m}$, kde $\sigma(n)$ označuje súčet vlastných kladných deliteľov (delitele z množiny $\{1,2,3, \dots, n-1\}$).

```
public boolean friendlyNumbers(int m, int n)
```

Príklady:

`friendlyNumbers(6,28) = true`, lebo $\frac{1+2+3}{6} = \frac{1+2+4+7+14}{28}$

`friendlyNumbers(16,17) = false`, lebo $\frac{1+2+4+8}{16} \neq \frac{1}{17}$

Kód v texte (10 bodov)

Najstaršie šifry vznikali už v antickom svete a šifrovanie do textov sa používa dodnes. Do triedy `Midtermarka` pridajte metódu `vratKod`. Táto metóda dostane ako parameter ne`null`ovú referenciu na reťazec (objekt triedy `String`) a vráti referenciu na nový reťazec, ktorý obsahuje iba veľké písmena, ktoré nie sú na začiatkoch viet.
Pozn.: Predpokladáme, že veta vždy začína veľkým písmenom a vždy končí znakom bodka `'.'`.

Príklady:

```
vratKod("Einsteinov syn Hans Einstein bol najšikovnejší z jeho synov. V pase mal zapísane rodisko Switzerland, kde väčšinu detstva strávil so svojou sestrou Lieserl. Počas detstva opakovane navštívili Oeschinenské jazero.") = "HESLO"
vratKod ("Chcel by som navštíviť hlavne mesto Peru. Lima čaká na mňa. Na tejto výprave chcem vidieť aj Amazonský prales a tancovať pravú Zumbu.") = "PAZ"
vratKod ("Shakespeare napísal dielo Rómeo a Júlia. Ich príbeh sa odohráva vo Verone (IT).") = "RJVIT"
```

```
public String vratKod (String r)
```

Časť 2

Čas do príchodu (10 bodov)

- *(2 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 a s náhodným natočením vo viditeľnej časti kresliacej plochy.

- *(8 bodov)* Korytnačka sa dokáže otáčať rýchlosťou 1° za sekundu. Rýchlosť pohybu korytnačky je jednotková vzdialenosť („1 pixel“) za sekundu. Korytnačka sa nedokáže v jednom okamihu súčasne posúvať aj otáčať. Plochu pretína zvislá priamka na súradnici x. Za aký najkratší čas v sekundách sa ku priamke môže dostať prvá korytnačka? Nezabudnite, že korytnačka sa môže otáčať v smere aj proti smeru hodinových ručičiek.
  Pozn.: Korytnačka ide ku čiare tak, že sa k nej otočí kolmo a ide dopredu pokiaľ sa jej nedotkne.

  Metóda `casKPriamke` má parameter `x` - súradnica priamky, a vráti najmenší čas v sekundách, za aký sa niektorá z korytnačiek vie dostať na priamku.

  ```
  public double casKPriamke (double x)
  ```

  Rada (matematika ZŠ): Vonkajší uhol k uhlu α má veľkosť $360° - \alpha$.

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

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