



Polsemestrálny test

Zadanie



Ústav informatiky
Prírodovedecká fakulta
UPJŠ v Košiciach

Dvakrát meraj (rozmyšľaj), 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,
- 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:

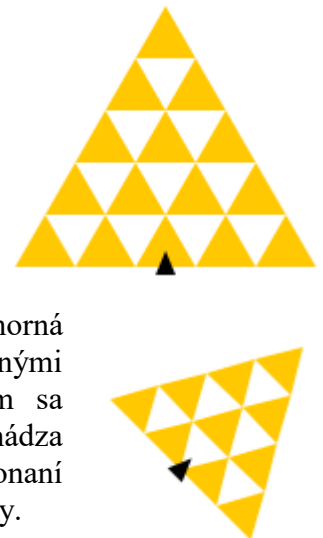
V **Časti 1** je cieľom úloh vytvoriť triedu Midtermarka, ktorá rozširuje triedu Turtle. Z prvej trojice úloh si **vyberte len 2 úlohy**, ktoré **budete riešiť!!!** To, ktoré úlohy ste sa rozhodli riešiť, uveďte v komentári pri odosielaní riešenia cez Moodle (ak to nie je zrejmé z odoslaného).

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

Časť 1 (dve úlohy z troch)

Pyramída (5 bodov)

Do triedy Midtermarka pridajte metódu `pyramida`, ktorá nakreslí pyramídu ako na obrázku vpravo. Metóda má dva parametre `strana` a `poschodia`. Parameter `strana` určuje dĺžku každej zo strán pyramídy, parameter `poschodia` určuje počet poschodí pyramídy (na obrázku vpravo má horná pyramída 5 poschodí, dolná 4 poschodia). Pyramída je tvorená vyplnenými rovnostrannými trojuholníkmi s dĺžkou strany `strana/poschodia`, pričom sa používa aktuálne nastavená farba výplne. Korytnačka sa na začiatku nachádza v strede jednej zo strán pyramídy a natočená je v smere výšky pyramídy. Po vykonaní metódy nech je korytnačka na pozícii a v smere, ako bola pred vykonaním metódy.



```
public void pyramida(double strana, int poschodia)
```

Rady:

- Odporúčame si vytvoriť metódu, ktorá nakreslí vyplnený trojuholník so zadanou dĺžkou strany a s počiatočnou a koncovou pozíciou ako je naznačené na obrázku.

```
public void vyplnenyTrojuholnik(double strana)
```

- Odporúčame si vytvoriť metódu, ktorá nakreslí jedno poschodie pyramídy skladajúce sa z n trojuholníkov so zadanou dĺžkou strany, pričom odporúčaná začiatočná a koncová pozícia sú naznačené na obrázku.

```
public void poschodie(double strana, int n)
```



Počet núl (5 bodov)

Do triedy Midtermarka pridajte metódu `pocetNul`, ktorá pre zadané prirodzené číslo c ($c \geq 1$) vráti koľkými nulami končí toto číslo. Uvažujeme zápis čísiel v desiatkovej sústave.

```
public int pocetNul(long c)
```

Príklady:

```
pocetNul(10300) = 2
```

```
pocetNul(13000) = 3
```

Hodnotenie: Maximálne 1 bod za riešenie, ktoré využije niektorú z metód na prevod čísla na jeho zápis v desiatkovej sústave vo forme postupnosti znakov.

Dĺžka rýmu (5 bodov)

Verš je základná jednotka básnického rytmu, spravidla jeden riadok básnického textu. Rým je zvuková zhoda na konci slov alebo skupín slov na konci veršov prípadne polveršov.

Do triedy `Midtermarka` pridajte metódu `dlzkaRymu`. Táto metóda dostane ako parametre `nnull`ové referencie na 2 reťazce (objekty triedy `String`) reprezentujúce verše. Metóda nech vráti dĺžku rýmu týchto dvoch reťazcov – t.j. maximálny počet spoločných písmen na konci jednotlivých veršov, ktoré sú zhodné, pričom medzery ignorujeme.

Príklady:

```
dlzkaRymu("od obloka po zemi",  
          "hned' do dlane vbehne mi") = 3
```

```
public int dlzkaRymu(String vers1, String vers2)
```

Za jednoduchšiu verziu, ktorá bude fungovať len na reťazcoch bez medzier, je možné získať 5 bodov. Ak ste sa rozhodli riešiť jednoduchšiu verziu, napíšte to v komentári.

```
dlzkaRymu("od obloka po zemi",  
          "hned' do dlane vbehne mi") = 2
```

Časť 2

Útek z plochy (5 bodov)

- (1 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 korytnáčiek triedy `Turtle` na náhodných pozíciách vo viditeľnej časti kresliacej plochy a náhodne otočené smerom 0, 90, 180 alebo 270.
- (4 bodov) Predpokladajme, že korytnačky sa nemôžu otočiť (t.j. každá korytnačka je fixne natočená smerom 0, 90, 180 alebo 270) a pohybujú sa rýchlosťou 1 pixel za sekundu. Za aký čas sa dokážu všetky korytnačky skryť (vyjsť/ujst' z kresliacej plochy), ak sa začnú hýbať naraz? Vytvorte metódu `unikovyCas`, ktorá vráti tento čas v sekundách.

```
public double unikovyCas()
```



Základné metódy objektov triedy String:

int length()

- o vráti dĺžku reťazca

char charAt(**int** index)

- o vráti znak na zadanom indexe v reťazci (znaky sú indexované od 0)

boolean equals(String r)

- o vráti *true* práve vtedy, keď tento reťazec sa skladá z tej istej postupnosti znakov ako reťazec referencovaný parametrom r

String trim()

- o vráti referenciu na novovytvorený reťazec vytvorený odstránením počiatočných a koncových medzier

String toLowerCase() resp. String toUpperCase()

- o vráti referenciu na novovytvorený reťazec po zmene znakov v reťazci na malé (veľké) písmena

String substring(**int** zacIndex, **int** konIndex)

- o vráti referenciu na novovytvorený reťazec obsahujúci podreťazec tvorený znakmi na indexoch zacIndex (vrátane) až konIndex (nie je zahrnutý)

int indexOf(String podreťazec) resp. **int** indexOf(**char** znak)

- o vráti index prvého výskytu podreťazca resp. znaku v reťazci. Ak sa v reťazci nenachádza vráti -1

Základné metódy objektov triedy Turtle:

void center()

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

void setPosition(**double** x, **double** y)

- o presunie korytnačku na pozíciu so súradnicami [x, y], čiara sa nekreslí

void step(**double** dlzka)

- o spraví krok v smere natočenia zadanej dĺžky, čiara sa kreslí v závislosti od stavu kresliaceho pera

void turn(**double** uhol)

- o otočí korytnačku o zadaný uhol v smere hodinových ručičiek

void moveTo(**double** x, **double** y)

- o korytačka spraví krok do bodu na súradniciach [x, y], čiara v závislosti od kresliaceho pera

void setDirection(**double** smer)

- o natočí korytnačku zadaným smerom (smer 0 je nahor, 90 doprava, atď.)

double getDirection()

- o vráti smer aktuálneho natočenia korytnačky

void turnTowards(**double** x, **double** y)

- o natočí korytnačku tak, aby bola natočená smerom k bodu na súradniciach [x, y]

double directionTowards(**double** x, **double** y)

- o vráti smer, pri ktorom by bola korytnačka natočená smerom k bodu na súradniciach [x, y]

double distanceTo(**double** x, **double** y)

- o vráti vzdialenosť korytnačky k bodu na súradniciach [x, y]

void dot(**double** polomer)

- o nakreslí vyplnený kruh (farbou výplne) so zadaným polomerom a stredom v pozícii korytnačky

void setFillColor(Color farba)

- o nastaví farbu výplne

void setPenColor(Color farba)

- o nastaví farbu kresliaceho pera

void penDown() resp. **void** penUp()

- o zapne resp. vypne kresliace pero

Základné metódy objektov triedy WinPane (kresliaca plocha):

void add(Turtle korytnacka)

- o pridá (referencovanú) korytnacku do kresliacej plochy

void remove(Turtle korytnacka)

- o odoberie (referencovanú) korytnacku z kresliacej plochy

int getWidth() resp. **int** getHeight()

- o vráti šírku, resp. výšku kresliacej plochy

Java a polia

- o 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

