1. Domácí úloha 08

Základní informace:

- **Účel:** dědění implementace, využití polymorfismu v aplikaci z dosud připravených tříd
- **Kostra:** `08_Polymorfismus.zip`
- **Odevzdávaný soubor aplikace:** `08_Polymorfismus.jar`
- **Odevzdávané soubory UML zabalené do JAR:** `08_uml.jar`

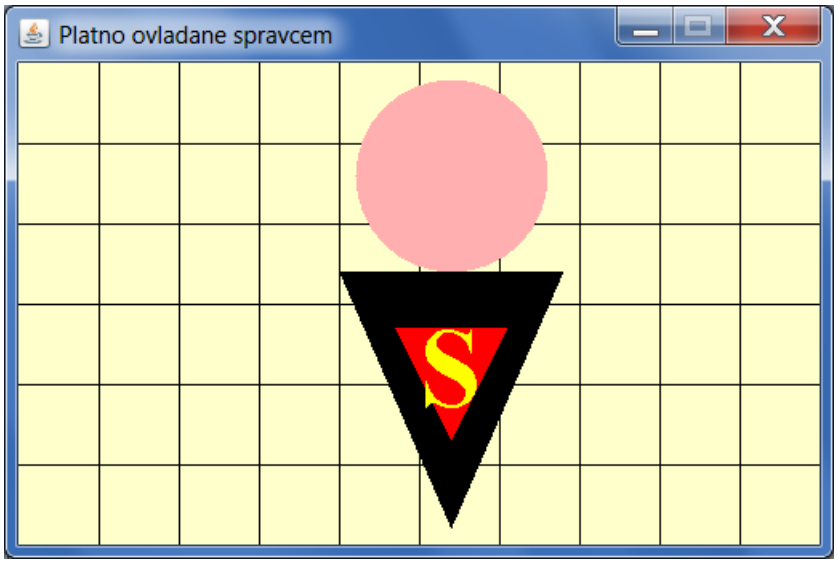
Zadání:

- připravte třídu `Superwoman`
- upravte třídu `Par`
- připravte třídu `Seznamka`
- do Portálu odevzdáte JAR soubor celého projektu
- rozšiřte UML diagram tříd

Postup řešení:

- stáhněte si soubor `08_Polymorfismus.zip`,
- rozbalte soubor - NEotvírejte projekt v BlueJ
- spusťte soubor `08_Polymorfismus.jar` pro ukázku fungování seznamky příkazem `java -jar 08_Polymorfismus.jar`
- vymažte soubor `08_Polymorfismus.jar`
- do rozbaleného adresáře `08_Polymorfismus` nakopírujte soubory `Osoba.java`, `Rozmer.java`, `Pohlavi.java`, `IMeritelny.java`, `IZvyrazneny.java`, `Zvyraznovac.java`, `Rande`, `Par` a `Superman`, které jste odevzdávali v minulém DU
- v BlueJ otevřete projekt `08_Polymorfismus`

- připravte třídu `Superwoman` která bude dědit implementaci od třídy `Osoba` - bude konstrukcí velmi podobná třídě `Superman`
  - vyznačuje se černým trojúhelníkovým tělem, červeným znakem jako má Superman a žlutým písmenem S



- **Pozor**: Nepokoušejte se dědit od třídy `Superman`.
- definujte statickou konstantu

```
private static final double POMER_ZNAK_TELO = 2.5/5.0;
```

- definujte následující instanční konstanty, které budou nastaveny v konstruktoru:
  - `Trojuhelnik superTelo`
  - `Trojuhelnik znak`
  - `int posunZnaku`
  - `Text pismenoS`
  - `int xPosunSkTelu`
  - `int yPosunSkTelu`
- vytvořte konstruktor se signaturou a kontraktem

```
  /**********************************************************
   * vytvoří instanci libovolného rozměru na zadané pozici
   * trojúhelníkové tělo {@code superTelo} je vždy černé,
   * šířky a výšky původního těla Osoby
   * na těle je červený trojúhelníkový znak {@code znak}
   * a na něm tučné žluté písmeno S {@code pismenoS}
   *
   * @param pozice pozice zobrazení
   * @param velikostHlavy velikost hlavy, podle které se vytvoří poměrově ▶
osoba
   */
  public Superwoman(Pozice pozice, int velikostHlavy) {
```

  - kostruktor nejprve pomocí `super()` vyvolá vhodný konstruktor `Osoba()` přičemž jako barvu původního těla osoby použijte `Barva.ZADNA`, což je průhledná barva

- vytvořte instanci `superTela` pomocí konstruktoru

```
public Trojuhelnik( Pozice pozice, Rozmer rozmer, Barva barva, Smer8 smer )
```

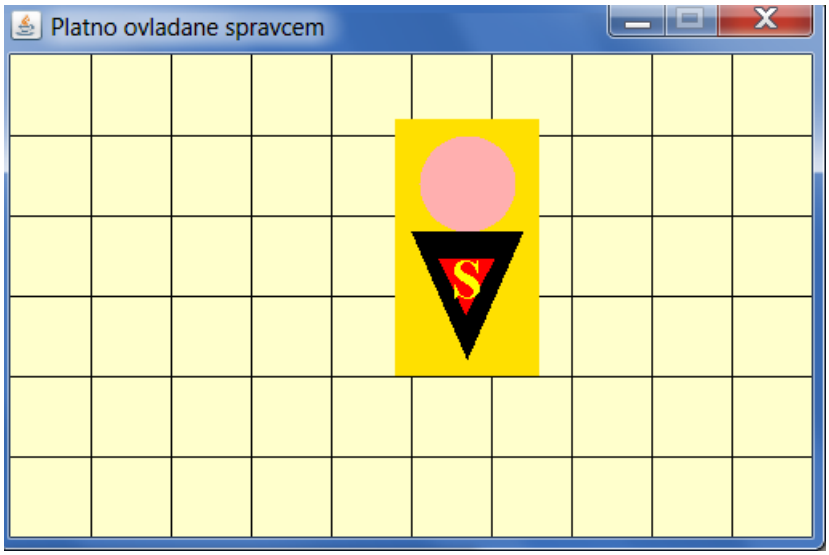
- jako ve třídě `Superman` vytvořte a správně umístěte červený znak
- písmeno S vytvořte a umístěte pomocí následujícího úseku kódu

```
int velikost = znak.getVyska();
xPosunSkTelu = (int) (velikost / 1.4);
yPosunSkTelu = velikost / 5;
pismenoS = new Text(telo.getX() + xPosunSkTelu, telo.getY() + ►
yPosunSkTelu, Barva.ZLUTA, "S");
pismenoS.setFont("Serif", Text.TUCNY, velikost);
```

Note

V kódu jsou pro určení pozice písmene S použita magická čísla, protože možnosti třídy `Text` jsou značně omezené.

- patřičně upravte (stejně jako u `Superman`) metodu `nakresli(Kreslitko kreslitko)`
- pro testovací účely připravte metody `getSuperTelo()`, `getZnak()` a `getPismenoS()`
- správnou funkci implementace ověřte pomocí testů *testKompletniKonstruktor()* a *testVelkaSuperwoman*() ze třídy `TestSuperwoman`, které před prvním použitím odkomentujte
- vytvořte konstruktory `public Superwoman(Pozice pozice)` a `public Superwoman()`
  - správnou funkci těchto konstruktorů otestujte pomocí testů *testKonstruktorPozice()* a *testBezparKonstruktor()*, které před prvním použitím odkomentujte
- překryjte metody `public void setPozice(int x, int y)` a `public void setPozice(Pozice p)` tak, aby bylo možné instance `Superwoman` přesouvat pomocí `Presouvace`
  - správnou funkci implementace ověřte pomocí odkomentovaných testů:
    - *testSetPoziceXY()*
    - *testSetPozicePoz()*
    - *testPresun()*
- pomocí testu *testZvyrazneni()* ověřte, že `Superwoman` zdědila i tuto schopnost



- zajistěte, aby třída `Par` implementovala rozhrani `IZvyrazneny`
  - správnou funkci implementace ověřte pomocí *testZvyrazneniParu*() ze třídy `TestRande`, který před prvním použitím odkomentujte

- připravte třídu `Seznamka`, která bude umožňovat schůzky a následný výběr vhodných párů
  - definujte statický třídní atribut `SP`
  - definujte defaultní hodnoty včetně jejich nastavení ve statickém inicializačním bloku:

```
private static final int DEF_RYCHLOST = 5;
private static final Pozice DEF_SRAZ_MUZI = new Pozice(10, 10);
private static final Pozice DEF_SRAZ_ZENY;
private static final Pozice DEF_SETKANI;

static {
  int sirkaPlatna = SP.getBsirka();
  int vyskaPlatna = SP.getBVyska();
  DEF_SRAZ_ZENY = new Pozice(sirkaPlatna - 100, vyskaPlatna - 100);
  DEF_SETKANI = new Pozice(sirkaPlatna / 3, vyskaPlatna / 3);
}
```

  - třída `Seznamka` je Jedináček (podrobně viz oop-03 str. 4), proto definujte a inicializujte statickou konstantu `Seznamka INSTANCE` podle zmíněného návrhového vzoru
    - tato konstanta musí být v souboru umístěna až za statickým inicializačním blokem
  - definujte instanční atributy
    - `seznamMuzu` a `seznamZen` jako seznamy objektů třídy `Osoba`
      - zde s výhodou využijeme možností polymorfismu, kdy v seznamu mohou být instance `Osoba`, ale i libovolného jejího potomka, tj. v našem případě `Superman` i `Superwoman`
      - dále uvidíte, že aniž by bylo nutné ve třídě `Seznamka` jakkoli rozlišovat instance, všechny bytosti se budou chovat dle očekávání

- ♦ `srazMuzu`, `srazZen` a `mistoSchuzky` typu `Pozice`
- ♦ `presouvac` typu `Presouvac`

- připravte setry k těmto atributům podle následujících signatur a kontraktů

```
/**
 * Změní pozici srazu mužů
 * pokud již v seznamu mužů nějací jsou, změní všem souřadnice
 *
 * @param pozice nová pozice srazu
 */
public void setSrazMuzu(Pozice pozice) {

/**
 * Změní pozici srazu žen
 * pokud již v seznamu žen nějaké jsou, změní všem souřadnice
 *
 * @param pozice nová pozice srazu
 */
public void setSrazZen(Pozice pozice) {

/**
 * Změní místo schůzky
 *
 * @param pozice nová pozice schůzky
 */
public void setMistoSchuzky(Pozice pozice) {

/**
 * Vytvoří nový přesouvač
 *
 * @param rychlost rychlost přesouvání
 */
public void setPresouvac(int rychlost) {
```

- připravte privátní bezparametrický konstruktor, ve kterém
  - ♦ vytvořte instance seznamů mužů a žen jako instance `ArrayList`
  - ♦ nastavte atributy `presouvac`, `srazMuzu`, `srazZen` a `mistoSchuzky` pomocí volání příslušných setrů, jejichž parametry budou defaultní konstanty definované na začátku
- připravte statickou tovární metodu `getSeznamka()`, která podle návrhového vzoru Jedináček vrací odkaz na instanci
- připravte instanční metody s následujícími signaturami a kontrakty:

```
/**
 * Přidá muže do seznamu
 * přidávanému muži nastaví pozici srazu mužů
 *
 * @param muz přidávaný muž
```

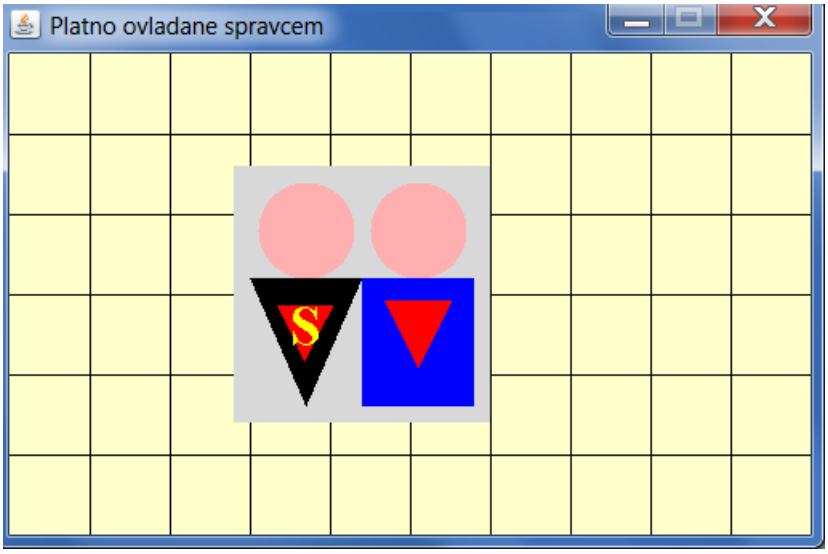
```
   */
  public void pridejMuze(Osoba muz) {

  /**
   * Přidá ženu do seznamu
   * přidávané ženě nastaví pozici srazu žen
   *
   * @param zena přidávaná žena
   */
  public void pridejZenu(Osoba zena) {
```

- připravte metodu - **Pozor:** metoda je neobvykle dlouhá

```
  /**
   * postupně zařizuje schůzky všem kombinacím párů z obou seznamů
   * pokud se sejde instance Superman a Superwoman, jsou považovány za ► 
ideální pár,
   * který společně odejde na určené místo
   *
   * @param spolecnyOdchod cílová pozice společného odchodu vybraného páru
   * @param dobaCekani v milisec mezi jednotlivými schůzkami
   * @return počet uskutečněných setkání
   */
  public int realizujSchuzky(Pozice spolecnyOdchod, int dobaCekani) {
```

- přes oba seznamy musíte iterovat pomocí iterátoru (podrobně viz oop-06), protože nalezené páry již nemají žádné další schůzky, tj. jsou ze seznamu odstraněny
  - vnější cyklus je přes seznam žen, vnitřní přes seznam mužů
- muže a ženu pošlete na schůzku pomocí metody `jdouNaRande()` třídy `Rande`
  - konstruktor třídy `Rande` nevytváří novou instanci muže a ženy, ale použije již existující
  - každá instance třídy `Rande` představuje jedno uskutečněné setkání
- instanci `Par` získáte pomocí `parJdeSpolecne()`, kdy pár jde na totéž místo, jako je místo schůzky
- po setkání počkejte pomocí `IO.cekej(dobaCekani);`
- v případě, že muž je `Superman` a žena `Superwoman` (zjistíte pomocí `instanceof`)
  - zvýrazněte obdélníkem tento pár - pomocí `Zvyraznovac`
  - vypište zprávu `IO.zprava("Ideální pár");`
  - odstraňte zvýraznění
  - pomocí `parJdeSpolecne()` odešlete pár na místo `spolecnyOdchod`
  - pomocí služby `remove()` iterátorů odstraňte muže i ženu z příslušných seznamů
  - pomocí `SP.odstranVse();` odstraňte pár z plátna

– pomocí `continue návěští;` vynuťte další průchod vnějším cyklem, kdy `návěští` uvozuje vnější cyklus



- ♦ v opačném případě přesuňte ženu i muže zpět na místa jejich srazů
- ♦ posledními příkazy vnitřního cyklu jsou

```
IO.cekej(dobaCekani);
SP.odstranVse();
```

- ♦ návratovou hodnotou je počet všech uskutečněných setkání

■ správnou funkci implementace ověřte pomocí *testSeznamky*() ze třídy `TestSeznamky`

- po úspěšném otestování zakomentujte v metodě `realizujSchuzky()` příkaz `IO.zprava("Ideální pár");`
- pokud na to při odevzdání zapomenete, skončí validace DU timeoutem validátoru

■ všechny vytvořené a upravované třídy prověřte pomocí PMD a odstraňte případné problémy

■ na závěr otestujte pomocí Duck-testů celou svoji práci a odstraňte případné problémy

Warning

Duck-test pro `Seznamka` trvá poměrně dlouho - mnohem déle, než testy ostatních tříd. Nenechte se tím znervóznit.

■ celý projekt již známým způsobem zabalte do JAR souboru `08_Polymorfismus.jar`, který budete odevzdávat

■ rozšiřte UML diagram tříd z minulého DÚ o nové třídy `Superwoman` a `Seznamka`

- zakreslete správnou vazbu dědičnosti ke třídě `Osoba`

Warning

Vyberte z nabídky správnou vazbu a natáhněte ji ve směru její šipky.

Pokud se pokusíte obrátit textovou úpravou směr vazby (šipky) v pravém dolním boxu, validátor tuto vazbu vyhodnotí jako chybnou.

- zakreslete správné vazby mezi třídami `Seznamka`, `Rande` a `Par`
- protože třída `Seznamka` přímo využívá třídy `Superman` a `Superwoman` musí mezi těmito třídami přibýt další vazby
- vztahy mezi třídami zakreslete včetně kardinalit
- ve výsledném schématu přemístěte všechny třídy tak, aby výsledný diagram byl co nejpřehlednější a vzájemné vazby se nekřížily (je to také otázka estetického vkusu, takže neexistuje jen jediné správné řešení)
- nezapomeňte soubor uložit pod jménem začínajícím `08` a v poznámce změnit číslo DU na 08
- výsledek uložte do souboru `.uxf` a také exportujte jako PNG soubor (ten si zobrazte a ujistěte se, zda obsahuje všechnu informaci)
  - jména souborů budou `08_A11B0987P.uxf` a `08_A11B0987P.png` - každý samozřejmě použije své osobní číslo
  - oba tyto soubory zabalíte do souboru `08_uml.jar` příkazem

```
jar cMf 08_uml.jar 08_*.uxf 08_*.png
```

  - tento soubor budete odevzdávat do **Blok 18-OOP-UML**