Olio-ohjelmoinnin perusteet – Demo 3

Huom! Kaikkiin luokkiin tulee kirjoittaa tarvittavat konstruktorit ja asetus- ja havainnointimetodit attribuuteille.

1. Kirjoita rajapintaluokka *Kappale,* joka mallintaa kolmiulotteista kappaletta. Luokka sisältää kaksi metodia, *annaAla()* ja *annaKeskipiste()*. Pohdi, missä muodossa kolmiulotteisin avaruuden piste on järkevintä palauttaa ja kirjoita tätä tarkoitusta varten oma luokkansa.

   Kirjoita lopuksi luokka *Kuutio*, joka toteuttaa *Kappale*-rajapinnan. (20 p.)

2. Jatketaan viime demojen Jalkapalojoukkue- ja Jalkapallopelaaja-luokkien kehittämistä.
   a. Muuta Jalapallopelaaja-luokkaa siten, että se perii Comparable-rajapinnan. Pelaajia tulee verrata toisiinsa viikkopalkan perusteella. Toteuta lisäksi luokkaan toString() –metodi, jolla pelaajan tiedot voidaan tulostaa järkevästi
   b. Kirjoita Jalkapallojoukkue-luokkaan metodi, jolla voidaan tulostaa kaikki joukkueen pelaajat viikkopalkan mukaisessa järjestyksessä pienimmästä suurimpaan. Kirjoita lisäksi metodi, jolla voidaan palauttaa n kappaletta joukkueen kalleimpia pelaajia. (20 p.)

3. Päätit tehdä talon omin hartiavoimin. Hankit kuitenkin vahingossa väärät piirustukset ja tarvikkeet – mistä nousi aikamoinen haloo. Ostit myös vasaran ja nauloja (koko rahalla). Onneksi kuitenkin työkalupakkisi on kunnossa, sillä se ohjelmoidaan nyt! Harjoitellaan siis polymorfismia:

   Luo siis luokka *työkalupakki*, sekä luokat *vasara*, *hiomakone*, *kirves*, *puukko* ja *sähkösaha*. Työkalupakkiin tulee voida säilöä yo. *työkaluja*, mutta myös tulevaisuudessa hankittavia muitakin työkaluja.

   Luo työkalupakkiin metodi, jolla kaikki *teroitettavat* työkalut saadaan teroitettua. Luo myös metodi, jolla kaikki *ladattavat* työkalut saadaan ladattua. Kaikkia työkaluja tulee voida käyttää, ja käytön johdosta aiheutuva ääni tulee tulostaa. **Työkalupakki**-luokan **toteutuksessa** (koodissa) **ei saa** missään kohtaa **löytyä** seuraavia **sanoja**: **vasara, hiomakone, kirves, puukko tai saha**. (30 p.)

4. Muutetaan jo toteutettu kaksisuuntainen linkitetty lista **järjestämättömäksi** yksisuuntaiseksi binääripuuksi. Lapset siis eivät tiedä vanhempaansa. (Binääripuu-termin pitäisi olla tuttu TKP:ltä; google auttaa, jos ei ole. Engl. binary tree.) Binääripuu tulee täyttää tasojärjestyksessä, eli edellisen tason täytyy olla täysi (kaikilla solmuilla on kaksi lasta), ennenkuin seuraavaa tasoa aletaan populoimaan. Toteuta siis ainakin lisäys ja hakumetodit. Puusta poistamista ei tarvitse toteuttaa. (30 p.)