ЕЛЕКТРОТЕХНИЧКИ ФАКУЛТЕТ
Одсеци ИР, ОС, ОФ Београд 22.04.2018.

Колоквијум из Објектно оријентисаног програмирања II

1) *(укупно 70 поена)* Саставити на језику *Java* следећи пакет класа:

- *(10 поена)* ***Морнар*** има име и квалитет који се задају приликом стварања и који могу да се дохвате. Квалитет је цео број у опсегу 0-100 (вредност већа од 100 или мања од 0 постављају квалитет на вредност 100 и 0 респективно). Може да се састави текстуални опис морнара у облику **`M_`***име* **`:`** *квалитет*.
- *(25 поена)* ***Брод*** има назив и може да садржи задати број морнара као посаду. Ствара се са задатим називом, бројем морнара који може да садржи и једним од тих морнара који у том тренутку представља капетана брода (не треба проверавати исправност унетих података). Капетан брода је увек тренутно најквалитетнији морнар на броду и може да се дохвати (покушај дохватања капетана из празног брода је регуларна ситуација у којој треба вратити одговарајућу вредност). Морнар се може додати на брод, може да се дохвати морнар са задатим индексом (грешка је уколико је индекс изван опсега - GIndeks), да се одреди индекс најгорег морнара, да се узме најбољи морнар, да се дохвати тренутни број морнара на броду, да се одреди једнословна врста брода и да се уклоне сви морнари са брода. Може да се израчуна реалан квалитет посаде брода као просечна вредност квалитета свих морнара. Брод може да нападне други брод само ако је испуњен одређени услов, а у том случају се бори са тим бродом. Побеђује брод са квалитетнијом посадом, без обзира на бројност посаде. Победнички брод уклања комплетну посаду са пораженог брода, а неке од њих преузима. Може да се састави текстуални опис брода у облику *врста* **`-`** *назив* **`:`** *квалитет_посаде*, након чега се у новом реду исписују морнари брода у облику **`(`***морнар***`,`** *морнар***`, ...,`** *морнар***`)`**.
- *(10 поена)* ***Гусарски брод*** је брод који може да нападне само брод чија је посада мање бројна од његове. Након успешног напада узима са пораженог брода и додаје у своју посаду капетана нападнутог брода само ако је он квалитетнији од морнара са најмањим квалитетом унутар те посаде и на броду има слободног места. Врста гусарског брода је **`G`**.
- *(10 поена)* ***Краљевски брод*** је брод који може да нападне само брод који није краљевски. Након успешног напада, посаду краљевског брода треба допунити (највише до расположивог капацитета) морнарима са пораженог брода који су квалитетнији од најмање квалитетног морнара са брода победника. Морнаре са пораженог брода узимати по опадајућем редоследу њиховог квалитета. Врста краљевског брода је **`K`**.
- *(15 поена)* ***Флота*** има назив и садржи произвољан број бродова. Ствара се празна, са задатим називом, после чега јој се бродови могу додавати редом, један по један. Први додати брод одређује врсту бродова који се могу додавати у флоту (грешка је уколико је брод који се додаје неадекватан - GNeadekvatan). Може да се дохвати брод са задатим индексом (грешка је уколико је индекс изван опсега - GIndeks), као и да се одреди реалан квалитет флоте као просечна вредност квалитета свих појединачних бродова. Може да се састави текстуални опис флоте у облику **`Flota(`***врста***`)`** *назив* **`:`** *квалитет*, након чега се у новим редовима исписују појединачни бродови, по један брод у реду. Пре него што се дода први брод у флоту, *врста* је **`-`** (цртица).

(0 поена). Приложена је класа са главном функцијом која испитује основне функционалности пакета класа уз исписивање резултата на стандардном излазу (конзоли).

НАПОМЕНЕ: **а)** Израда решења задатка траје **100** минута.
б) Рад се предаје искључиво на предвиђеном мрежном диску.
в) Називе типова и метода ускладити са називима из приложене класе са главном функцијом.
г) На располагању је приступ *Web* адреси: `https://docs.oracle.com/javase/8/docs/api/`.
д) Није дозвољено имати поред себе друге материјале, нити уз себе имати електронске уређаје, без обзира да ли су укључени или искључени.
ђ) Резултати колоквијума биће објављени на *Web*-у на адреси: http://rti.etf.rs/rti/ir2oo2/index.html/

```java
import more.*;

public class Main {

   public static void main(String[] args) {

      Mornar m1 = new Mornar("A", 50), m2 = new Mornar("B", 100), m3 = new Mornar("C", 60), m4 = new
Mornar("D", 40);
      GusarskiBrod g1 = new GusarskiBrod("Pearl", 5, m1);
      KraljevskiBrod k1 = new KraljevskiBrod("St. John", 2, m3);

      try {
         g1.dodajMorn(m2);
         g1.dodajMorn(new Mornar("E", 60));
         k1.dodajMorn(m4);

         System.out.println("Broj mornara na gusarskom brodu: " + g1.dohvBr());
         System.out.println("Kapetan gusarskog broda: " + g1.dohvatiKapetana());
         try {
            System.out.println("Mornar kraljevskog broda na poziciji 1: " + k1.dohvMorn(1));
         } catch(GIndeks e) {}

         System.out.println("*** Ispis brodova ***");
         System.out.println(g1);
         System.out.println(k1);

         g1.napadni(k1);

         System.out.println("*** Ispis brodova nakon napada ***");
         System.out.println(g1);
         System.out.println(k1);

         Flota flota = new Flota("F");
         flota.dodaj(k1);
         try {
            flota.dodaj(g1);  //neuspesno dodavanje gusarskog broda u flotu
         } catch(GNeadekvatan e) {}
         flota.dodaj(new KraljevskiBrod("Santa Maria", 3, new Mornar("F", 40)));
         System.out.println("*** Izgled flote ***");
         System.out.println(flota);
         flota.dohvBrod(0).dodajMorn(new Mornar("G", 50));
         System.out.println("*** Nakon obnove flote ***");
         System.out.println(flota);
      } catch (Exception e) {
         System.err.println(e);
      }
   }
}

//Primer izvrsavanja

Broj mornara na gusarskom brodu: 3
Kapetan gusarskog broda: M_B : 100
Mornar kraljevskog broda na poziciji 1: M_D : 40
*** Ispis brodova ***
G - Pearl : 70.0
(M_A : 50, M_B : 100, M_E : 60)
K - St. John : 50.0
(M_C : 60, M_D : 40)
*** Ispis brodova nakon napada ***
G - Pearl : 67.5
(M_A : 50, M_B : 100, M_E : 60, M_C : 60)
K - St. John : 0.0
()
*** Izgled flote ***
Flota(K) F : 20.0
K - St. John : 0.0
()
K - Santa Maria : 40.0
(M_F : 40)
*** Nakon obnove flote ***
Flota(K) F : 45.0
K - St. John : 50.0
(M_G : 50)
K - Santa Maria : 40.0
(M_F : 40)
```

```java
package more;
public class Mornar {
   private String ime;
   private int kvalitet;

   public Mornar(String ime, int kvalitet) {
      this.ime = ime;
      this.kvalitet = kvalitet<0 ? 0 :
      (kvalitet>100 ? 100 : kvalitet);
   }
   public int dohvKval() { return kvalitet; }
   public String dohvIme() { return ime; }
   public String toString() {
      return "M_" + ime + " : " + kvalitet;
   }
}
```

```java
public abstract class Brod {
   private String naziv;
   protected Mornar []mor;
   protected int br, kapetan;

   public Brod(String naziv, int kap, Mornar m) {
      this.naziv=naziv; mor=new Mornar[kap]; mor[br++]=m;
   }
   public int najgoriMornar() {
      int m = 0, i;
      if(br==0) return -1;
      for(i = 0; i < mor.length && mor[i]==null; i++);
      for(m = i, i = m+1; i < mor.length; i++)
         if(mor[i]!=null && mor[i].dohvKval() <
            mor[m].dohvKval()) m = i;
      return m;
   }
   public Mornar izbaciNajboljegMornara() {
      int i, max, max2 = 0;
      if(br==0) return null;
      for(i = 0; i < mor.length && mor[i]==null; i++);
      for(max = i, i = max+1; i < mor.length; i++)
         if(mor[i]!=null && mor[i].dohvKval() >
            mor[max].dohvKval()) { max2 = max; max = i; }
         else if(mor[i]!=null && (mor[max2]==null ||
         mor[i].dohvKval()>mor[max2].dohvKval())) max2=i;
      Mornar m = mor[max]; mor[max] = null;
      kapetan = max2; br--;
      return m;
   }
   public Mornar dohvatiKapetana() {
      return br>0?mor[kapetan]:null;
   }
   public void dodajMorn(Mornar m) {
      if(br<mor.length) { int i;
         for(i = 0; i < mor.length && mor[i]!=null; i++);
            mor[i] = m;
         if(br==0 || m.dohvKval() >
            mor[kapetan].dohvKval()) kapetan = br;
         br++;
      }
   }
   public Mornar dohvMorn(int ind) throws GIndeks {
      if(ind<0 || ind>=mor.length) throw new GIndeks();
      return mor[ind];
   }
   public int dohvBr() { return br; }
   public abstract char vrsta();
   public void isprazni() {
      for(int i = 0; i < mor.length; mor[i++] = null);
      br = 0;
   }
   public double kvalitet() {
      double k = 0;
      for(int i = 0; i < mor.length; i++)
         k += mor[i]!=null?mor[i].dohvKval():0;
      return br!=0?k/br:0;
   }
   protected abstract boolean uslovNapad(Brod b);
   protected abstract void obracunajSe(Brod b);
   public void napadni(Brod b) {
      if(uslovNapad(b)) {
         if(kvalitet()>b.kvalitet()) {
            obracunajSe(b); b.isprazni(); }
         else if(b.kvalitet()>kvalitet()) {
            b.obracunajSe(this); isprazni(); } }
   }
   public String toString() {
      StringBuilder sb = new StringBuilder();
      sb.append(vrsta() + " - " + naziv + " : " +
                kvalitet() + "\n(");
      for(int i = 0, j = 0; i < mor.length; i++) {
         if(mor[i]==null) continue;
         sb.append(mor[i]); j++;
         if(j < br) sb.append(", ");
      }
      sb.append(")");
      return sb.toString();
   }
}
```

```java
public class GusarskiBrod extends Brod {
   public static char vrs = 'G';
   public GusarskiBrod(String naziv, int kap, Mornar m) {
      super(naziv, kap, m);
   }
   public char vrsta() { return vrs; }
   protected boolean uslovNapad(Brod b) {return br>b.br;}
   protected void obracunajSe(Brod b) {
      Mornar m = b.dohvatiKapetana();
      if(m!=null && m.dohvKval() >
      mor[najgoriMornar()].dohvKval()) dodajMorn(m);
   }
}
```

```java
public class KraljevskiBrod extends Brod {
   public static char vrs = 'K';
   public KraljevskiBrod(String naziv,int kap,Mornar m) {
      super(naziv, kap, m);
   }
   public char vrsta() { return vrs; }
   protected boolean uslovNapad(Brod b) {
      return b.vrsta()!=vrs;
   }
   protected void obracunajSe(Brod b) {
      while(br<mor.length) {
         Mornar m = b. uzmiNajboljegMornara();
         if(m==null) break;
         if(m.dohvKval()>mor[najgoriMornar()].dohvKval())
            dodajMorn(m);
         else break;
      }
   }
}
```

```java
public class Flota {
   private String naziv;
   private static class Elem {
      Brod b; Elem sled;
      Elem(Brod b) { this.b = b; }
   }
   private Elem prvi, posl;
   public Flota(String naziv) { this.naziv = naziv; }
   public void dodaj(Brod b) throws GNeadekvatan {
      if(prvi!=null && prvi.b.vrsta()!=b.vrsta())
         throw new GNeadekvatan();
      Elem novi = new Elem(b);
      if(prvi!=null) posl.sled = novi; else prvi = novi;
      posl = novi;
   }
   public Brod dohvBrod(int ind) throws GIndeks {
      Elem tmp = prvi;
      while(tmp!=null && ind>0) { tmp = tmp.sled; ind--; }
      if(tmp==null) throw new GIndeks();
      return tmp.b;
   }
   public double kvalitet() {
      double k = 0; int br = 0; Elem tmp = prvi;
      while(tmp!=null) {
         k += tmp.b.kvalitet(); tmp = tmp.sled; br++;
      }
      return prvi!=null?k/br:0;
   }
   public String toString() {
      StringBuilder sb = new StringBuilder();
      sb.append("Flota(" + (prvi!=null?prvi.b.vrsta():"")
      + ") " + naziv + " : " + kvalitet() + "\n");
      Elem tmp = prvi;
      while(tmp!=null) {
         sb.append(tmp.b);
         if(tmp.sled!=null) sb.append("\n");
         tmp = tmp.sled;
      }
      return sb.toString();
   }
}
```

```java
public class GIndeks extends Exception {
   public String toString() { return "*** Greska u
   indeksu! ***"; }
}
```

```java
public class GNeadekvatan extends Exception {
   public String toString() { return "*** Greska
   neadekvatan brod! ***"; }
}
```