Ülesannete tüüpide tutvustus!

Võimalike teemade ring hõlmab kogu kursust!!!

2019. a kevad

Tegelikult on eksamitöös 4 ülesannet

Ülesanne 1 (? punkti)

Võib eeldada, et vajalikud asjad on imporditud ja klassi algus ja lõpp ning `main`-meetod on ka korralikult olemas.

Skitseerida, mis ilmub järgmise programmi käivitamisel ekraanile. Värvused, suurused jms. märkida juurde:

```
public void start(Stage st) {
    GridPane p1 = new GridPane();

    String s = "öötöö";
    StringBuilder sb = new StringBuilder(s);

    if (s.equals(sb.reverse().toString())){
        BorderPane p2 = new BorderPane();
        Button n1 = new Button("N1");
        Button n2 = new Button("N2");
        p2.setLeft(n1);
        p2.setCenter(n2);
        p1.add(p2, 1, 1);
    }

    String s2 = "tartu";
    StringBuilder sb2 = new StringBuilder(s2);

    if (sb2.equals(sb2.reverse())){
        Circle k1 = new Circle(300, 30, 30, Color.RED);
        p1.add(k1, 1, 2);
    }

    String s3 = "OOP";
    StringBuilder sb3 = new StringBuilder(s3);

    if (sb3.toString().equals(sb3.reverse().toString())){
        Rectangle k2 = new Rectangle(50, 50, 100, 100);
        p1.add(k2, 2, 2);
    }

    Scene s1 = new Scene(p1, 400, 300);
    st.setTitle("Eksam");
    st.setScene(s1);
    st.show();
}
```

Teavet klassi `StringBuilder` kohta

`StringBuilder`	`reverse()` Causes this character sequence to be replaced by the reverse of the sequence.

Klassis `StringBuilder` ei ole meetodit `equals` ülekaetud.

Teavet klassi `Circle` kohta

Constructor and Description
`Circle(double centerX, double centerY, double radius, Paint fill)` Creates a new instance of Circle with a specified position, radius and fill.

Teavet klassi `Rectangle` kohta

Constructor and Description
`Rectangle(double width, double height, Paint fill)` Creates a new instance of Rectangle with the given size and fill.

Teavet klassi `GridPane` kohta

GridPane lays out its children within a flexible grid of rows and columns.

`add(Node child, int columnIndex, int rowIndex)`

Ülesanne 2 (? punkti)

```
import java.util.*;
public class Struktuurid {
	public static void main(String[] args) {
		HashSet<Integer> hulk1 = new HashSet<Integer>();
		hulk1.add(3);
		hulk1.add(1);
		hulk1.add(1);
		ArrayList<Integer> list1 = new ArrayList<Integer>();
		list1.add(1);
		list1.add(2);
		list1.add(1, 1);
		System.out.println(list1);
		System.out.println(hulk1);
		System.out.println(hulk1.addAll(list1));
		System.out.println(hulk1.addAll(list1));
		System.out.println(hulk1);
	}
}
```

Mis ilmub ekraanile? Miks?

Mis muutuks, kui rea `ArrayList<Integer> list1 = new ArrayList<Integer>();` asemel oleks rida `List<Integer> list1 = new ArrayList<Integer>();` ?

Mis muutuks, kui rea `ArrayList<Integer> list1 = new ArrayList<Integer>();` asemel oleks rida `List<Integer> list1 = new List<Integer>();` ?

Teavet

Liidese `Set` dokumentatsioonis on meetodi `addAll` kohta kirjas. Kuidas puutub üldse liides `Set` antud programmi?

addAll

`boolean addAll(Collection<? extends E> c)`

Adds all of the elements in the specified collection to this set if they're not already present (optional operation). If the specified collection is also a set, the `addAll` operation effectively modifies this set so that its value is the *union* of the two sets. The behavior of this operation is undefined if the specified collection is modified while the operation is in progress.

Specified by:

`addAll` in interface `Collection<E>`

Parameters:

`c` - collection containing elements to be added to this set

Returns:

`true` if this set changed as a result of the call

Ülesanne 3 (? punkti)

```
public class Erind {
     public static void main(String[] args) {
          int[] a = {1, 0};
          try {
               System.out.println(1/a[1]);
               System.out.println("Vahel");
               System.out.println(a[3]);

          } catch (ArithmeticException e) {
               System.out.println("Esimene püünis");
          } catch (RuntimeException e) {
               System.out.println("Teine püünis");
          } finally {
               System.out.println("Epiloog");
               System.out.println(a[3]);
          }
          System.out.println("Pärast");
     }
}
```

Mis ilmub ekraanile? Selgitada.

Teavet

Klass `ArithmeticException` on klassi `RuntimeException` alamklass. Klass `ArrayIndexOutOfBoundsException` on klassi `RuntimeException` (kaudne) alamklass.

Ülesanne 4 (? punkti)

```
class TrükiD {
	TrükiD() {
		System.out.print("d");
	}
	TrükiD(boolean suur) {
		if (suur) {
			System.out.print("D");
		}
	}
}

class TrükiE extends TrükiD {
	TrükiE() {
		System.out.print("e");
	}
	TrükiE(boolean suur){
		super(suur);
		if (suur) {
			System.out.print("E");
		}
	}
}

class TrükiF extends TrükiE {
	TrükiF(boolean suur) {
		if (suur) {
			System.out.print("F");
		}
	}
}
```

Ülesandeid käsitletakse üksteisest eraldiseisvatena. Näiteks üks vigane isendiloome ei takista korrektseid.

Ülesanne 4.1

Mis ilmub ekraanile, kui klassi `TrükiF` isend luuakse `new TrükiF(true);` abil? Selgitada.

Ülesanne 4.2

Mis ilmub ekraanile, kui klassi `TrükiE` isend luuakse `new TrükiE(true);` abil? Selgitada.

Ülesanne 4.3

Mis ilmub ekraanile, kui klassi `TrükiD` isend luuakse `new TrükiF(false);` abil? Selgitada.

Ülesanne 5 (? punkti)

(Pseudo)juhuslike tõeväärtuste genereerimiseks on klassis `Random` meetod `nextBoolean`.

Selleks, et saada tõeväärtusi kallutatumalt on loodud uus klass.

```
import java.util.Random;
public class KallutatudJuhuslik extends Random {
	double kallutaja;

	KallutatudJuhuslik(int protsente){
		kallutaja = protsente/100.0;
	}

	public boolean nextBoolean(){
		if (nextDouble() > kallutaja) return true;
		return false;
	}
}
```

Peaklassis on püütud erineval moel saada juhuslikke tõeväärtusi.

```
KallutatudJuhuslik juh1 = new KallutatudJuhuslik(40);
System.out.println(juh1.nextBoolean());
Random juh2 = new KallutatudJuhuslik(40);
System.out.println(juh2.nextBoolean());
Random juh3 = new Random(40);
System.out.println(juh3.nextBoolean());
KallutatudJuhuslik juh4 = new Random(40);
System.out.println(juh4.nextBoolean());
```

Kas see kood kompileerub? Kui ei, siis mis põhjusel. Kui (vajadusel vigased read välja kommenteerides) programm käivitada, siis millise tõenäosusega ilmub igal erineval juhul ekraanile `true`. Selgitada!

Teavet

Java API. Random

If two instances of `Random` are created with the same seed, and the same sequence of method calls is made for each, they will generate and return identical sequences of numbers.

`Random()`
Creates a new random number generator.

`Random(long seed)`
Creates a new random number generator using a single `long` seed.

`nextBoolean()`
Returns the next pseudorandom, uniformly distributed `boolean` value from this random number generator's sequence.

`nextDouble()`
Returns the next pseudorandom, uniformly distributed `double` value between `0.0` and `1.0` from this random number generator's sequence.

Ülesanne 6 (? punkti)

On antud neli faili. Täita lüngad nii, et ülejäänud koodi ei ole vaja muuta. Lünk võib sisaldada ka mitut sõna või jääda tühjaks. Kui mingite lünkade vahel on sõltuvused, siis needki märkida (stiilis "kui esimesse lünka panna ..., siis kolmandasse lünka saab panna ...").

Kortermaja.java:

```
public class Kortermaja _________ Hoone _________ Elatav _________ Mõõdetav {
                                                  //1., 2., 3. lünk
      private static int maksimumKõrgus;
      private int korteriteArv;
}
```

Hoone.java:

```
public abstract class Hoone {

      public _________ String hooneOmanik();       //4. lünk

      public _________ String toString() {         //5. lünk
            return "Hoone - omanik: " + hooneOmanik();
      }
}
```

Elatav.java:

```
public interface Elatav {
      public int elanikeArv();
}
```

Mõõdetav.java:

```
public _________ Mõõdetav {                        //6. lünk
      public int kõrgus();
}
```

Täita lüngad järgnevates lausetes. 4. ja 5. lauses valida sobiv sõna sulgudest.

1. Klass `Kortermaja` _____ `Hoone` _____.

2. Klass `Kortermaja` _____ `Elatav` _____.

3. Klass `Kortermaja` _____ `Mõõdetav` _____.

4. Muutuja `maksimumKõrgus` on klassi `Kortermaja` _____.

   (isendimuutuja/klassimuutuja)

5. Muutuja `korteriteArv` on klassi `Kortermaja` _____.

   (isendimuutuja/klassimuutuja)

6. Selleks, et kood kompileeruks, peavad klassis `Kortermaja` olema vähemalt meetodid

   _____.

Ülesanne 7 (? punkti)

Käsurea argumentidena anti programmile ette 1 ja 2 ning loodeti saada ekraanile 0.5. Paraku see päriselt ei õnnestunud. Järgnevas loetelus on erinevad võimalikud põhjused. Selgitada **iga** variandi korral, kuivõrd see antud juhul aktuaalne on. Võib eeldada, et vajalikud asjad on imporditud.

1. Avalik meetod ei saa privaatset meetodit välja kutsuda.

2. Antud argumentide korral ei tagasta funktsioon `f1` arvu 0.5.

3. Peameetodi päises puudub `throws ArithmeticException`.

4. Tekib `ArrayIndexOutOfBoundsException` erind.

5. Tekib `ArithmeticException` erind.

6. Väljund kirjutatakse hoopis faili.

Mis on teisiti, kui käsurea argumentidena antakse programmile ette 1 ja 0?

Mis on teisiti, kui käsurea argumente üldse ette ei anta?

```java
public class Aritmeetika {

	public static void main(String[] args) throws FileNotFoundException {
		OutputStream output = new FileOutputStream("systemout.txt");
		PrintStream printOut = new PrintStream(output);
		System.setOut(printOut);

		System.out.println(f1(Integer.parseInt(args[0]), Integer.parseInt(args[1])));
	}

	private static double f1(int a, int b) throws ArithmeticException {
		return a / b;
	}
}
```

Teavet klassi `ArithmeticException` kohta

```
public class ArithmeticException
extends RuntimeException
```

Thrown when an exceptional arithmetic condition has occurred. For example, an integer "divide by zero" throws an instance of this class. `ArithmeticException` objects may be constructed by the virtual machine as if suppression were disabled and/or the stack trace was not writable.

Ülesanne 8 (? punkti)

On antud kolm klassi. Võib eeldada, et vajalikud asjad on imporditud.

```
public class A implements Comparable<A> {
	private int a;

	public A(int a) {
		this.a = a;
	}

	public int compareTo(A a) {
		return this.a - a.a;
	}

	public String toString() {
		return Integer.toString(a);
	}
}

public class B implements Comparable<B> {
	private int b;

	public B(int b) {
		this.b = b;
	}

	public int compareTo(B b) {
		return b.b - this.b;
	}

	public String toString() {
		return Integer.toString(b);
	}
}

public class TestAB {

	public static void main(String[] args) {
		Queue<A> a = new LinkedList<>();
		a.add(new A(3));
		a.add(new A(5));
		a.add(new A(2));
		väljasta(a);

		Queue<B> b = new PriorityQueue<>();
		b.add(new B(3));
		b.add(new B(5));
		b.add(new B(2));
		väljasta(b);

		Queue<A> a2 = new PriorityQueue<A>(a);
		väljasta(a2);
	}

	public static <T> void väljasta(Queue<T> c) {
		while (!c.isEmpty()) {
			System.out.println(c.poll());
		}
	}
}
```

Mis väljastatakse ekraanile? Selgitada!

Mis juhtub, kui meetod `poll()` asendada meetodiga `peek()`?

Mis juhtub, kui esialgses (meetodiga `poll()`) programmis rida `väljasta(a)` välja kommenteerida?

Mis juhtub, kui esialgses programmis muutuja `a2` väärtustataks
`PriorityQueue<A> a2 = new PriorityQueue<A>(a);`?

Väljavõtted APIst

Teavet klassi `PriorityQueue<E>` konstruktori kohta

```
public PriorityQueue(Collection<? extends E> c)
```

Creates a `PriorityQueue` containing the elements in the specified collection. If the specified collection is an instance of a `SortedSet` or is another `PriorityQueue`, this priority queue will be ordered according to the same ordering. Otherwise, this priority queue will be ordered according to the natural ordering of its elements.

Parameters:

`c` - the collection whose elements are to be placed into this priority queue

Throws:

`ClassCastException` - if elements of the specified collection cannot be compared to one another according to the priority queue's ordering

`NullPointerException` - if the specified collection or any of its elements are null

Teavet liidese `Queue<E>` kohta

E	**peek()** Retrieves, but does not remove, the head of this queue, or returns `null` if this queue is empty.
E	**poll()** Retrieves and removes the head of this queue, or returns `null` if this queue is empty.

Ülesanne 9 (? punkti)

Inimesed seisavad kassajärjekorras. Mõnele inimesele tundub, et teises kassas on lühem järjekord ja ta läheb sinna. Inimese ja kassajärjekorra kujutamiseks on vastavad klassid.

```
public class Inimene {
	String nimi;
	Inimene eelmine;
	public Inimene(String nimi) {
		this.nimi = nimi;
	}
	public String toString() {
		return nimi;
	}
}
public class KassaJärjekord {
	private Inimene algus;
	private Inimene lõpp;
	private int nr = 1;

	KassaJärjekord(){}

	KassaJärjekord(KassaJärjekord k){
		algus = k.algus;
		lõpp = k.lõpp;
		nr = k.nr;
	}

	void lisa(Inimene i){
		if (algus == null){
			algus = i;
			lõpp = algus;
			System.out.println("Esimene kassajärjekorras: " + i);
		}
		else { if (nr % 3 == 0)
				System.out.println("Läks teise kassa juurde: " + i);
			else {
				i.eelmine = lõpp;
				lõpp = i;
				System.out.println("Järgmisena tulnud: " + i);
			 }
		}
		nr++;
	}

	public String toString(){
		StringBuilder välja = new StringBuilder("Kassajärjekord: ");
		Inimene i = lõpp;
		while (i != null) {
			välja = välja.append(i + " -> ");
			i = i.eelmine;
		}
		return välja.substring(0, välja.length()-3);
	}

	Inimene meetod(Inimene i){
		Inimene i1 = lõpp;
		i.eelmine = lõpp.eelmine.eelmine.eelmine;
		lõpp.eelmine.eelmine.eelmine = i;
		lõpp = lõpp.eelmine;
		return i1;
	}
}
```

Peaklass on järgmine.

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
public class KassaPeaklass {
	public static void main(String[] args) {
		Inimene i1 = new Inimene("Margus");
		Inimene i2 = new Inimene("Kadri");
		Inimene i3 = new Inimene("Taavi");
		Inimene i4 = new Inimene("Jevgeni");
		List<Inimene> inimesed = new ArrayList<Inimene>(Arrays.asList(i1, i2, i3, i4));
		KassaJärjekord k = new KassaJärjekord();
		for (Inimene i : inimesed)
			k.lisa(i);
		System.out.println(k);
		System.out.println("==========");
		KassaJärjekord k2 = new KassaJärjekord(k);
		k2.lisa(new Inimene("Mart"));
		k2.lisa(new Inimene("Andres"));
		System.out.println(k2);
		System.out.println("==========");
		System.out.println(k.meetod(new Inimene("Luisa")));
		System.out.println(k);
	}
}
```

Mis väljastatakse ekraanile? Selgitada!

Ülesanne 10 (? punkti)

Seikluspargis koosneb rada rippsildadest. Selles ülesandes on raja läbimist simuleeritud.

```
public class Rippsild {
   private char sillaTähis;
   private int läbimiseAeg;
   public Rippsild(char sillaTähis, int läbimiseAeg) {
       this.läbimiseAeg = läbimiseAeg;
       this.sillaTähis = sillaTähis;
   }
   public void ületa(Seikleja seikleja) throws InterruptedException {
// public synchronized void ületa(Seikleja seikleja) throws InterruptedException {
       System.out.println(seikleja.getNimi() + " astus sillale " + sillaTähis + " "
                  + ((System.currentTimeMillis() - seikleja.getAlgusAeg()) / 1000));
       Thread.sleep(1000 * läbimiseAeg);
       System.out.println(seikleja.getNimi() + " lahkus sillalt " + sillaTähis + " "
                  + (System.currentTimeMillis() - seikleja.getAlgusAeg()) / 1000);
   }
}

public class Seikleja implements Runnable {
   private String nimi;
   private boolean kasAlgusest;
   private Rippsild[] rippsillad;
   private long algusAeg;
   public Seikleja(String nimi, Rippsild[] rippsillad, boolean kasAlgusest) {
       this.nimi = nimi;
       this.rippsillad = rippsillad;
       this.kasAlgusest = kasAlgusest;
   }
   public String getNimi() {
       return nimi;
   }
   public long getAlgusAeg() {
       return algusAeg;
   }
   public void run() {
       this.algusAeg = System.currentTimeMillis();
       try {
           if (kasAlgusest) {
               for (int i = 0; i < rippsillad.length; i++)
                   rippsillad[i].ületa(this);
           } else {
               for (int i = rippsillad.length - 1; i >= 0; i--)
                   rippsillad[i].ületa(this);
           }
       } catch (InterruptedException e) {
           throw new RuntimeException(e);
       }
   }
}
```

Peaklass on järgmine.

```
public class Seikluspark {
   public static void main(String[] args) {
       Rippsild[] sillad = {new Rippsild('A', 10), new Rippsild('B', 20)};
       Seikleja seikleja1 = new Seikleja("Rasmus", sillad, true);
       Seikleja seikleja2 = new Seikleja("Grete", sillad, false);
       Thread t1 = new Thread(seikleja1);
       Thread t2 = new Thread(seikleja2);
       t1.start();
       t2.start();
   }
}
```

Ülesanne 10.1

Mis ilmub ekraanile. Miks?

Ülesanne 10.2

Mis ilmuks ekraanile, kui meetod `ületa` ees oleks lisaks võtmesõna **`synchronized`**? Miks?

Teavet klassi `Thread` kohta

```
static void sleep(long millis)
```

Causes the currently executing thread to sleep (temporarily cease execution) for the specified number of milliseconds, subject to the precision and accuracy of system timers and schedulers.

Teavet klassi `System` kohta

```
static long currentTimeMillis()
```

Returns the current time in milliseconds.

Ülesanne 11 (? punkti)

Taheti koostada programm, mis simuleeriks laste mängimist mänguasjadega mängutoas. Programm pidi tagama järgmiste tingimuste kehtimise:

- Mängutoas saab korraga mitu last mängida.
- Laps saab korraga võtta mänguasjariiulist ainult ühe mänguasja.
- Konkreetse mänguasjaga saab korraga mängida vaid üks laps.
- Laps eelistab alati mänguasja, millel on kõige pikem nimi.
- Pärast mänguasjaga mängimist paneb laps selle riiulile tagasi.

Paraku see päriselt **ei õnnestunud**. Märkida järgnevast loetelust, mis põhjus(t)el see programm korrektselt ei tööta. Selgitada **iga** vastusevariandi korral, kuivõrd see võib olla põhjuseks.

1. Laps ei saa kasutada mängutuba, sest tema lõim võib enne tööd alustada, kui `Mängutuba` on loodud.

2. Meetodit `run` klassist `Laps` ei ole üldse välja kutsutud.

3. Meetod `leiaMänguasi` klassis `Mängutuba` peab olema sünkroniseeritud.

4. Meetod `tagastaMänguasi` klassis `Mängutuba` peab olema sünkroniseeritud.

5. Meetod `run` klassis `Laps` peab olema sünkroniseeritud.

6. Klassi `Laps` isenditele ei saa üldse konstruktori parameetreid anda, sest `Laps` realiseerib liidest `Runnable`.

7. Teine laps ei saa kunagi mängimist alustada, sest esimese lapse tegevus on lõpmatus tsüklis.

8. Isend `laps1` on loodud valesti, talle tüübiks on pandud `Runnable`, seda ei tohi teha.

9. Isend `laps2` on loodud valesti, talle tüübiks on pandud `Laps`, aga peab olema kindlasti `Runnable`.

```java
public class Laps implements Runnable {
   private Mängutuba mängutuba;
   private String nimi;

   public Laps(Mängutuba mängutuba, String nimi) {
       this.mängutuba = mängutuba;
       this.nimi = nimi;
   }

   public void run() {
       while (true) {
           String mänguasi = mängutuba.leiaMänguasi();
           System.out.println(nimi + " mängib mänguasjaga " + mänguasi);
           mängutuba.tagastaMänguasi(mänguasi);
       }
   }
}

public class Mängutuba {
   private String[] riiul;

   public Mängutuba(String[] mänguasjad) {
       this.riiul = mänguasjad;
   }

   public String leiaMänguasi() {
       int parim = -1;
       for (int i = 0; i < riiul.length; i++) {
           if (riiul[i] != null
                   && (parim == -1 || riiul[i].length() > riiul[parim].length())) {
               parim = i;
           }
       }
       String valik = riiul[parim];
       riiul[parim] = null;
       return valik;
   }

   public void tagastaMänguasi(String mänguasi) {
       for (int i = 0; i < riiul.length; i++) {
           if (riiul[i] == null) {
               riiul[i] = mänguasi;
               break;
           }
       }
   }

   public static void main(String[] args) {
       Mängutuba mängutuba = new Mängutuba(new String[]{"rong", "puzzle", "nukk"});
       Runnable laps1 = new Laps(mängutuba, "Volli");
       Laps laps2 = new Laps(mängutuba, "Martin");
       Thread t1 = new Thread(laps1);
       Thread t2 = new Thread(laps2);
       t1.start();
       t2.start();
   }
}
```