

Detta dokument är ett exempel, cirka "andra hälften" av en tentamen för TDA545 Objektorienterad programvaruutveckling

Fulltentamen vitsord: 3=28p, 4=38p, 5= 48p, max 60p.

Max 30p i denna "halvtentamen"

Hjälpmedel: på sista sidan av dokument finns ett utdrag ur Javas API

Uppgift 4: [12 poäng totalt] *referensvärden, private/public, rekursion (i datatypen)*

Klassen `IntList` som finns nedan implementerar en lista av heltal (`int`).

```
public class IntList {  
  
    private int value;  
    private IntList next;  
  
    public IntList(int value, IntList next) {  
        this.value = value;  
        this.next = next;  
    }  
  
    public static void printList(IntList l) {  
        System.out.print("[ ");  
        while (l != null) {  
            System.out.print(l.value + " ");  
            l = l.next;  
        }  
        System.out.println("]");  
    }  
  
    public static void main(String[] args) {  
        IntList l;  
        l = new IntList(5,null);  
        l = new IntList(4,l);  
        l = new IntList(3,l);  
        IntList.printList(l);  
    }  
}
```

a) Vad skrivs ut när programmet körs? [3 poäng]

b) Skriv en metod `arrayToList` så att följande program,

```
int[] a = {3,4,5,6,7};  
IntList.printList(arrayToList(a));
```

skriver ut strängen "[3 4 5 6 7]". [4 poäng]

c) För denna delfråga bör du anta att `private` har bytts till `public` i koden för klassen `IntList`. Visa hur man kan skriva kod *utanför* `IntList` klassen som får `IntList.printList` att gå i en oändlig loop. [5 poäng]

Exempel: utskrift från `printList` kan se såhär ut när `printList` fastnar i en oändlig loop.

```
[ 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 ...
```

Tips: Tänk, hur ser listan ut för att få tillstånd sådan utskrift? Varför var det viktiga att vi bytte `private` till `public` i koden för klassen `IntList` ovan?

Uppgift 5: [10 poäng totalt] *händelsehantering, timer*

- a) Beskriv kort den inbyggda `Timer` klassen (dvs. `javax.swing.Timer`). Beskriv vad en lyssnare är och hur `Timer` klassen använder sig av lyssnare. [3 poäng]
- b) Skriv kod för en ny `Timer`-lik klass. Den nya klassen bör heta `TickTimer`.
 - Den skall ha en konstruktör med signaturen:

```
public TickTimer(int delay, ActionListener al, int ticks)
```
 - En instans av denna klass bör anropa sin lyssnare exakt `ticks` gånger med `delay` millisekunder mellan anropen. [7 poäng]

Tips: Det är kanske bäst att *inte* ärva den inbyggda `Timer` klassen. Istället kan man skapa en `Timer` inuti `TickTimer` konstruktören.

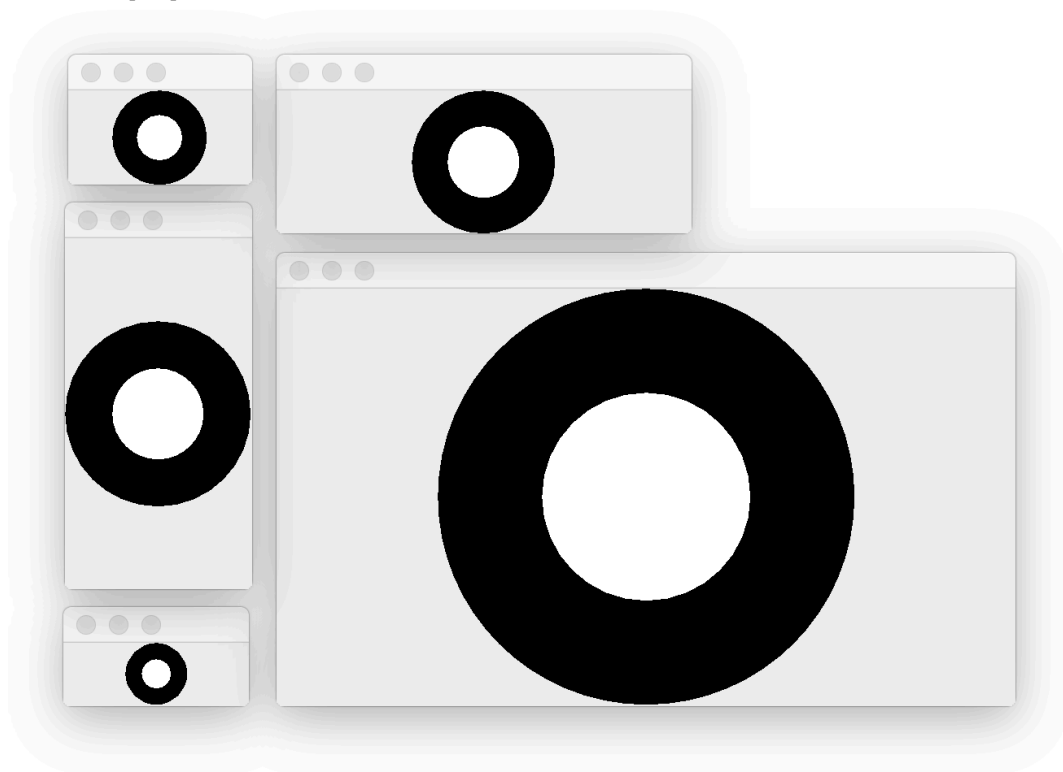
Uppgift 6: [8 poäng totalt] *grafik, arv, swing*

Skriv ett program som ritat två cirklar i mitten av ett fönster.

- Den större cirkeln bör vara svart. Den ska placeras i mitten av fönstret och bör vara så stor som möjligt utan att gå utanför fönstrets kanter.
- Den mindre cirkeln bör vara vit och ritas i mitten av den större cirkeln. Den mindre cirkeln skall ha en radius som är hälften av den stora cirkelns radius.

Cirklarna skall ritas i en `JPanel` som sitter i ett `JFrame` fönster,

Exempel på hur det bör se ut finns nedan.



Obs. Det skall vara möjligt att ändra på storleken av fönstret och då skall cirklarna hållas i mitten av fönstret.

Utdrag ur Javas API. *Obs.* Man behöver inte använda alla dessa klasser. Man får också använda annat från Javas API.

Interface ActionListener

void actionPerformed(ActionEvent e)
Invoked when an action occurs.

Class ActionEvent extends AWTEvent extends EventObject extends Object

ActionEvent(Object source, int id, String command)
Constructs an ActionEvent object.

Class Color extends Object

static Color BLACK
The color black.
static Color WHITE
The color white.

Class Component extends Object

int getHeight()
Returns the current height of this component.
int getWidth()
Returns the current width of this component.

Class Container extends Component extends Object

Component add(Component comp)
Appends the specified component to the end of this container.

Class FlowLayout extends Object, implements LayoutManager

FlowLayout()
Constructs a new FlowLayout with a centered alignment and a default 5-unit horizontal and vertical gap.

Class Graphics extends Object

void drawOval(int x, int y, int width, int height)
Draws the outline of the specified oval.
abstract void fillOval(int x, int y, int width, int height)
Fills the specified oval.
abstract void setColor(Color c)
Sets this graphics context's current color to the specified color.

Class JComponent extends Container extends Component extends Object

protected void paintComponent(Graphics g)
Calls the UI delegate's paint method, if the UI delegate is non-null.

Class JFrame extends Frame extends Window extends Container extends Component extends Object

Container getContentPane()
Returns the contentPane object for this frame.

Class JPanel extends JComponent extends Container extends Component extends Object

JPanel()
Creates a new JPanel with a double buffer and a flow layout.
JPanel(LayoutManager layout)
Create a new buffered JPanel with the specified layout manager

Interface LayoutManager

Class Object

boolean equals(Object obj)
Indicates whether some other object is "equal to" this one.
String toString()
Returns a string representation of the object.

Class Timer extends Object

Timer(int delay, ActionListener listener)
Creates a Timer and initializes both the initial delay and between-event delay to delay milliseconds.
void setActionCommand(String command)
Sets the string that will be delivered as the action command in ActionEvents fired by this timer.
void setRepeats(boolean flag)
If flag is false, instructs the Timer to send only one action event to its listeners.
void start()
Starts the Timer, causing it to start sending action events to its listeners.
void stop()
Stops the Timer, causing it to stop sending action events to its listeners.

Class Window extends Container extends Component extends Object

void setVisible(boolean b)
Shows or hides this Window depending on the value of parameter b.