Høgskolen i Østfold

EKSAMEN

Emnekode: ITF10213	Emne: Innføring i programmering (Høst 2013)
Dato: 03.12.2013	Eksamenstid: kl 09.00 til kl 13.00
Hjelpemidler: Fire egenproduserte A4-sider.	Faglærer: Harald Holone Børre Stenseth
Eksamensoppgaven: Oppgavesettet består av 15 sider inklusiv denne forsiden og vedlegg. Kontroller at oppgaven er komplett før du begynner å besvare spørsmålene.	
Sensurdato: 03.01.2014 Karakterene er tilgjengelige for studenter på studentweb senest 2 virkedager etter oppgitt sensurfrist. Følg instruksjoner gitt på: www.hiof.no/studentweb	

Eksamen i innføring i programmering

Høsten 2013

Les hele oppgavesettet før du begynner å løse oppgavene. Husk å planlegge tiden din godt.

1 Begreper (20%)

a) Forklar med korte eksempler hva vi forstår med disse nøkkelordene i Processing: `final` og `new`.

b) Forklar med korte eksempler hva vi forstår med disse nøkkelordene: `extends` og `instanceof`.

c) Hva er forskjellen på en array og en `ArrayList`?

d) Hva betyr nøkkelordet `return`?

2 Forståelse (30%)

I en tekstfil ligger det et antall linjer. Hver linje inneholder et antall positive heltall med semikolon (;) som skilletegn. Vi vet ikke hvor mange linjer filen har, og vi vet ikke hvor mange tall det er på hver linje. Det vi vet er at det er minst én linje, at det finnes minst ett tall på hver linje og at alle linjene har riktig format. Sketchen nedenfor benytter en slik fil.

```
int[]chosen;
int best;

void setup(){
  size(500,500);
  frameRate(1);
```

```
  loadData("resultater.csv");
}

void loadData(String filename){
// a
  String[] lines=loadStrings(filename);
  chosen=new int[lines.length];
  best=0;
// b
  for(int ix=0;ix < lines.length;ix++){
    chosen[ix]=0;
    String[] numbers=lines[ix].split(";");
// c
    for(int j=0;j<numbers.length;j++){
        int number=int(numbers[j]);
        if (number > chosen[ix])
          chosen[ix]=number;
    }
  }
// d
  for (int ix=0;ix < chosen.length;ix++){
    if(best < chosen[ix])
      best=chosen[ix];
  }
// e
  for(int ix=0;ix < chosen.length;ix++)
    println(chosen[ix]);
  println("------------");
  println(best);
}

void draw(){
  background(255);
  fill(0);
  text(chosen[(int)frameRate % chosen.length],100,100);
}
```

a) Forklar hva som skjer i `loadData()`. Bruk kommentarmerkene (a,b,c,d,e) som referansepunkter i din beskrivelse.

b) Hva blir utskriften fra `loadData()` dersom datafila, "resul-

tater.csv" ser slik ut:

```
12;34;76;1;91;20;12;1;87;12
112;314;7;1;19;201;19
125;34;6;1;92;202;29;1
312;34;61;1;29;203;128
```

c) Forklar hva som skjer i `draw()`.

d) Under ser du to eksempler på bruk av tekststrenger i lister.
Hva oppnår vi med å bruke `StringList` istedet for `ArrayList<String>`?
Tips: Vedlagt er dokumentasjonen for datatypen `StringList`
slik den er beskrevet i Processing.

```
StringList sl = new StringList();
sl.add("Huha!");

ArrayList<String> al = new ArrayList();
al.add("Huha!");
```

3 Utvikling (50%)

Anta følgende klasse:

```
class Curve{
  String title;
  float[] data;
  float maxValue;

  Curve(String title, float[]data){
    this.title=title;
    this.data=data;
    maxValue=0;
    for(int ix=0;ix < data.length;ix++){
      if(data[ix]>maxValue)
        maxValue=data[ix];
    }
  }
```

```
  /*
  Tegner et histogram med vertikale søyler.
  Hver søyle representerer en verdi
  */
  void drawHistogram(){
  }

  /*
  Tegner en sammengengende linje mellom verdipunktene.
  Verdipunktene markeres med en liten sirkel.
  */
  void drawCurve(){
  }

  /*
  Tegner et kakediagram
  der hvert kakestykke representerer en verdi
  */
  void drawPieChart(){
  }
}
```

a) Du skal implementere enten `drawHistogram()` eller `drawCurve()`. Histogrammet/kurven skal benytte hele vinduets høyde og bredde (`height`, `width`). Tittel skal være med i nærheten av diagrammet. Bruk fullstendig og kommentert kode. Husk at gode kommentarer til en viss grad kan kompensere for mangelfull kode.

b) Lag en skisse av hvordan resultatet av din `drawHistogram()` eller `drawCurve()` vil se ut. Anta følgende datasett: `{4, 20, 8, 10}`

c) Du skal implementere metoden `drawPieChart()`. Kakediagrammet skal benytte hele vinduets høyde og bredde (`height`, `width`). Tittel skal være med i nærheten av diagrammet. Tips: Se dokumentasjonen for `arc(...)` som er vedlagt oppgavesettet. Bruk fullstendig og kommentert kode. Husk at gode kommentarer til en viss grad kan kompensere for mangelfull kode.

d) Lag en enkel skisse av hvordan resulatet av din implementasjon av `drawPieChart()` vil se ut. Anta samme datasett som i oppgave b).

e) Lag et utkast til en hovedsketch som lar brukeren velge mellom de forskjellige fremstillingsmåtene (kurve, histogram, kakediagram). Bruk fullstendig og kommentert kode.

SLUTT PÅ OPPGAVESETTET

Vedlegg: Processing-dokumentasjon for `StringList`, `arc()` og `map()`.

Cover
Download
Exhibition
Reference
Libraries
Tools
Environment
Tutorials
Examples
Books
Overview
People
Foundation
Shop
» Forum
» GitHub
» Issues
» Wiki

This reference is for Processing 2.0+. If you have a previous version, use the reference included with your software. If you see any errors or have suggestions, please let us know. If you prefer a more technical reference, visit the Processing Javadoc.

Name

StringList

Examples

```
StringList inventory;

void setup() {
  size(200, 200);
  inventory = new StringList();
  inventory.append("coffee");
  inventory.append("flour");
  inventory.append("tea");
  println(inventory);
  noLoop();
  fill(0);
  textAlign(CENTER);
}

void draw() {
  String item = inventory.get(2);
  text(item, width/2, height/2);
}
```

» FAQ
» Twitter
» Facebook

Description

Helper class for a list of Strings. Lists are designed to have some of the features of ArrayLists, but to maintain the simplicity and efficiency of working with arrays.

Functions like sort() and shuffle() always act on the list itself. To get a sorted copy, use list.copy().sort().

Methods

size()	Get the length of the list
clear()	Remove all entries from the list
get()	Get an entry at a particular index
set()	Set an entry at a particular index
remove()	Remove an element from the specified index
append()	Add a new entry to the list
hasValue()	Check if a value is a part of the list
sort()	Sorts the array in place
sortReverse()	Reverse sort, orders values from highest to lowest
reverse()	To come...
shuffle()	Randomize the order of the list elements

lower()	Make the entire list lower case
upper()	Make the entire list upper case
array()	Create a new array with a copy of all the values

Constructor `StringList()`

Related

IntList
FloatList

Updated on November 5, 2013 03:43:35pm EST

‹

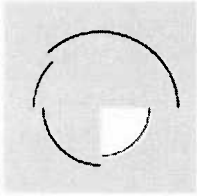
Cover
Download
Exhibition
Reference
Libraries
Tools
Environment
Tutorials
Examples
Books
Overview
People
Foundation
Shop
» Forum
» GitHub
» Issues
» Wiki

This reference is for Processing 2.0+. If you have a previous version, use the reference included with your software. If you see any errors or have suggestions, please let us know. If you prefer a more technical reference, visit the Processing Javadoc.

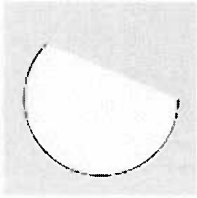
Name

arc()

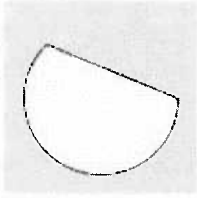
Examples



```
arc(50, 55, 50, 50, 0, HALF_PI);
noFill();
arc(50, 55, 60, 60, HALF_PI, PI);
arc(50, 55, 70, 70, PI, PI+QUARTER_PI);
arc(50, 55, 80, 80, PI+QUARTER_PI, TWO_PI);
```

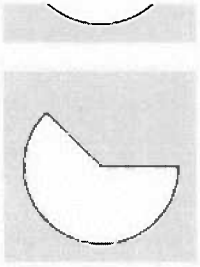


```
arc(50, 50, 80, 80, 0, PI+QUARTER_PI, OPEN);
```



```
arc(50, 50, 80, 80, 0, PI+QUARTER_PI, CHORD);
```

» FAQ
» Twitter
» Facebook



```
arc(50, 50, 80, 80, 0, PI+QUARTER_PI, PIE);
```

Description Draws an arc to the screen. Arcs are drawn along the outer edge of an ellipse defined by the `a`, `b`, `c`, and `d` parameters. The origin of the arc's ellipse may be changed with the `ellipseMode()` function. Use the `start` and `stop` parameters to specify the angles (in radians) at which to draw the arc.

There are three ways to draw an arc; the rendering technique used is defined by the optional seventh paramter. The default mode is OPEN, and the other options are CHORD and PIE. Each is shown in the above examples.

Syntax

```
arc(a, b, c, d, start, stop)
arc(a, b, c, d, start, stop, mode)
```

Parameters

`a`	float: x-coordinate of the arc's ellipse
`b`	float: y-coordinate of the arc's ellipse
`c`	float: width of the arc's ellipse by default
`d`	float: height of the arc's ellipse by default

	`start`	float: angle to start the arc, specified in radians
	`stop`	float: angle to stop the arc, specified in radians

Returns `void`

Related

```
ellipse()
ellipseMode()
radians()
degrees()
```

Updated on November 5, 2013 03:43:31pm EST

‹

Cover
Download
Exhibition
Reference
Libraries
Tools
Environment
Tutorials
Examples
Books
Overview
People
Foundation
Shop
» Forum
» GitHub
» Issues
» Wiki

This reference is for Processing 2.0+. If you have a previous version, use the reference included with your software. If you see any errors or have suggestions, please let us know. If you prefer a more technical reference, visit the Processing Javadoc.

Name

map()

Examples

```
size(200, 200);
float value = 25;
float m = map(value, 0, 100, 0, width);
ellipse(m, 200, 10, 10);
```

```
float value = 110;
float m = map(value, 0, 100, -20, -10);
println(m);  // Prints "-9.0"
```

```
void setup() {
  size(200, 200);
  noStroke();
}

void draw() {
  background(204);
```

» FAQ
» Twitter
» Facebook

```
  float  x1 = map(mouseX, 0, width, 50, 150);
  ellipse(x1, 75, 50, 50);
  float x2 = map(mouseX, 0, width, 0, 200);
  ellipse(x2, 125, 50, 50);
}
```

Description

Re-maps a number from one range to another.

In the first example above, the number 25 is converted from a value in the range of 0 to 100 into a value that ranges from the left edge of the window (0) to the right edge (width).

As shown in the second example, numbers outside of the range are not clamped to the minimum and maximum parameters values, because out-of-range values are often intentional and useful.

Syntax

```
map(value, start1, stop1, start2, stop2)
```

Parameters

`value`	float: the incoming value to be converted
`start1`	float: lower bound of the value's current range
`stop1`	float: upper bound of the value's current range
`start2`	float: lower bound of the value's target range
`stop2`	float: upper bound of the value's target range

Returns float

Related norm()
lerp()

Updated on November 5, 2013 03:43:30pm EST

Processing is an open project intiated by Ben Fry and Casey Reas. It is developed by a small team of volunteers.

© Info \ Site hosted by Media Temple.