Module 03

Input / Output

Which of these functions completes the code below to print hooray?

```
if (addOne(2) === 3) {
    print("Hooray!");
}
```

(A)

```
function addOne(x) {
  x + 1;
}
```

(B)

```
function addOne(x) {
  return x + 1;
}
```

(C)

```
function addOne(x) {
  print( x + 1 );
}
```

(D)

```
function addOne(x){
  x = x + 1;
}
```

We can write more interesting programs when we can exchange information with the outside world!

Problem with the outside world: there’s a lot of it.

Reading and writing text

Marley was dead: to begin with. There is no doubt whatever about that. The register of his burial was signed by the clergyman, the clerk, the undertaker, and the chief mourner. Scrooge signe
Mind! I don't mean to say that I know, of my own knowledge, what there is particularly dead about a door-nail. I might have been inclined, myself, to regard a coffin-nail as the deadest piece

Scrooge knew he was dead? Of course he did. How could it be otherwise? Scrooge and he were partners for I don't know how many years. Scrooge was his sole executor, his sole administrato

The mention of Marley's funeral brings me back to the point I started from. There is no doubt that Marley was dead. This must be distinctly understood, or nothing wonderful can come of the

Scrooge never painted out Old Marley's name. There it stood, years afterwards, above the warehouse door: Scrooge and Marley. The firm was known as Scrooge and Marley. Sometimes peopl

Oh! But he was a tight-fisted hand at the grindstone, Scrooge! a squeezing, wrenching, grasping, scraping, clutching, covetous old sinner! Hard and sharp as flint, from which no steel had ev

External heat and cold had little influence on Scrooge. No warmth could warm, no wintry weather chill him. No wind that blew was bitterer than he, no falling snow was more intent upon its

Nobody ever stopped him in the street to say, with gladsome looks, "My dear Scrooge, how are you? When will you come to see me?" No beggars implored him to bestow a trifle, no children a

But what did Scrooge care? It was the very thing he liked. To edge his way along the crowded paths of life, warning all human sympathy to keep its distance, was what the knowing ones call "

Once upon a time -- of all the good days in the year, on Christmas Eve -- old Scrooge sat busy in his counting-house. It was cold, bleak, biting weather: foggy withal: and he could hear the peop

The door of Scrooge's counting-house was open that he might keep his eye upon his clerk, who in a dismal little cell beyond, a sort of tank, was copying letters. Scrooge had a very small fire, bu

"A merry Christmas, uncle! God save you!" cried a cheerful voice. It was the voice of Scrooge's nephew, who came upon him so quickly that this was the first intimation he had of his approach

Plain text is the “default” mode of information storage and communication. Being able to work with text gives us access to large amounts of real-world data.

Received: from CONNMBX02.connect.uwaterloo.ca ([129.97.149.109]) by
 connhub1.connect.uwaterloo.ca ([129.97.149.101]) with mapi id 14.03.0319.002;
 Tue, 17 Jan 2017 15:57:38 -0500
From: Rishabh Moudgil <rishabh.moudgil@uwaterloo.ca>
To: Craig Kaplan <csk@uwaterloo.ca>
CC: Kevin Harrigan <kevinh@uwaterloo.ca>, Kristina Bayda
    <kbayda@uwaterloo.ca>, Travis Bartlett <travis.bartlett@uwaterloo.ca>
Subject: A01 Marking Scheme
Thread-Topic: A01 Marking Scheme
Thread-Index: AdJw/+DUxNKRRlCRRKOZfc2CQLKSng==
Date: Tue, 17 Jan 2017 20:57:36 +0000
Message-ID: <748888CA42FDF349AF07A8978DDED060281C9EC0@connmbx02>
Accept-Language: en-CA, en-US
Content-Language: en-CA
X-MS-Exchange-Organization-AuthAs: Internal
X-MS-Exchange-Organization-AuthMechanism: 04
X-MS-Exchange-Organization-AuthSource: connhub1.connect.uwaterloo.ca
X-MS-Has-Attach:
X-MS-Exchange-Organization-SCL: -1
X-MS-TNEF-Correlator:
Content-Type: multipart/alternative;
    boundary="_000_748888CA42FDF349AF07A8978DDED060281C9EC0connmbx02_"
MIME-Version: 1.0

--_000_748888CA42FDF349AF07A8978DDED060281C9EC0connmbx02_
Content-Type: text/plain; charset="Windows-1252"
Content-Transfer-Encoding: quoted-printable

//gallery.bridgesmathart.org/exhibitions/2017-joint-mathematics-meetings" "Mozilla/5.0 (Windows NT 6.1; WOW64; r
v:50.0) Gecko/20100101 Firefox/50.0"
108.62.132.133 - - [17/Jan/2017:00:00:15 -0500] "GET /tmp/cache/images/cms/arrow-right.gif HTTP/1.1" 404 195 "ht
tp://bridgesmathart.org/tmp/cache/stylesheet_combined_6fa5fb1be8f2682b13e4cf7292f5937a.css" "Mozilla/5.0 (Window
s NT 6.1; WOW64; rv:50.0) Gecko/20100101 Firefox/50.0"
108.62.132.133 - - [17/Jan/2017:00:00:16 -0500] "GET /bridges-galleries/conference-photos/ HTTP/1.1" 200 14016 "
http://bridgesmathart.org/bridges-galleries/art-exhibits/" "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:50.0) Gecko/2
0100101 Firefox/50.0"
73.64.123.57 - - [17/Jan/2017:00:01:24 -0500] "GET /2014/bridges2014-235.pdf HTTP/1.1" 200 948062 "-" "Mozilla/5
.0 (Macintosh; Intel Mac OS X 10_12_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/55.0.2883.95 Safari/537.36"
58.10.140.128 - - [17/Jan/2017:00:01:25 -0500] "GET /wp-login.php HTTP/1.1" 404 195 "-" "Mozilla/5.0 (Windows NT
 6.1; WOW64; rv:40.0) Gecko/20100101 Firefox/40.1"
58.10.140.128 - - [17/Jan/2017:00:01:26 -0500] "GET / HTTP/1.1" 200 12340 "-" "Mozilla/5.0 (Windows NT 6.1; WOW6
4; rv:40.0) Gecko/20100101 Firefox/40.1"
64.126.161.169 - - [17/Jan/2017:00:01:28 -0500] "GET /2012/cdrom/proceedings/92/paper_92.pdf HTTP/1.1" 200 21833
8 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_5) AppleWebKit/602.3.12 (KHTML, like Gecko)"
64.126.161.169 - - [17/Jan/2017:00:01:29 -0500] "GET /apple-touch-icon-precomposed.png HTTP/1.1" 404 195 "-" "Sa
fari/10602.3.12.0.1 CFNetwork/720.5.7 Darwin/14.5.0 (x86_64)"
64.126.161.169 - - [17/Jan/2017:00:01:29 -0500] "GET /apple-touch-icon.png HTTP/1.1" 404 195 "-" "Safari/10602.3
.12.0.1 CFNetwork/720.5.7 Darwin/14.5.0 (x86_64)"
64.126.161.169 - - [17/Jan/2017:00:01:29 -0500] "GET /favicon.ico HTTP/1.1" 404 195 "-" "Safari/10602.3.12.0.1 C
FNetwork/720.5.7 Darwin/14.5.0 (x86_64)"
64.126.161.169 - - [17/Jan/2017:00:01:30 -0500] "GET /apple-touch-icon-precomposed.png HTTP/1.1" 404 195 "-" "Sa
fari/10602.3.12.0.1 CFNetwork/720.5.7 Darwin/14.5.0 (x86_64)"
64.126.161.169 - - [17/Jan/2017:00:01:30 -0500] "GET /apple-touch-icon.png HTTP/1.1" 404 195 "-" "Safari/10602.3
.12.0.1 CFNetwork/720.5.7 Darwin/14.5.0 (x86_64)"
64.126.161.169 - - [17/Jan/2017:00:01:30 -0500] "GET /favicon.ico HTTP/1.1" 404 195 "-" "Safari/10602.3.12.0.1 C
FNetwork/720.5.7 Darwin/14.5.0 (x86_64)"
64.126.161.169 - - [17/Jan/2017:00:01:31 -0500] "GET /apple-touch-icon-precomposed.png HTTP/1.1" 404 195 "-" "Sa
fari/10602.3.12.0.1 CFNetwork/720.5.7 Darwin/14.5.0 (x86_64)"
64.126.161.169 - - [17/Jan/2017:00:01:31 -0500] "GET /apple-touch-icon.png HTTP/1.1" 404 195 "-" "Safari/10602.3
.12.0.1 CFNetwork/720.5.7 Darwin/14.5.0 (x86_64)"
64.126.161.169 - - [17/Jan/2017:00:01:32 -0500] "GET /favicon.ico HTTP/1.1" 404 195 "-" "Safari/10602.3.12.0.1 C
FNetwork/720.5.7 Darwin/14.5.0 (x86_64)"
64.126.161.169 - - [17/Jan/2017:00:01:32 -0500] "GET /apple-touch-icon-precomposed.png HTTP/1.1" 404 195 "-" "Sa
fari/10602.3.12.0.1 CFNetwork/720.5.7 Darwin/14.5.0 (x86_64)"

PROCESSING P R AA1 S EH0 S IH0 NG
PROCESSION P R AH0 S EH1 SH AH0 N
PROCESSION(1) P R OW0 S EH1 SH AH0 N
PROCESSIONAL P R AH0 S EH1 SH AH0 N AH0 L
PROCESSIONAL(1) P R OW0 S EH1 SH AH0 N AH0 L
PROCESSIONS P R OW0 S EH1 SH AH0 N Z
PROCESSOR P R AA1 S EH2 S ER0
PROCESSOR'S P R AA1 S EH2 S ER0 Z
PROCESSORS P R AA1 S EH2 S ER0 Z
PROCH P R AA1 K
PROCHASKA P R AH0 HH AA1 S K AH0
PROCHAZKA P R AH0 HH AA1 Z K AH0
PROCHNOW P R AA1 N AW0
PROCIDA P R OW0 CH IY1 D AH0
PROCK P R AA1 K
PROCKTER P R AA1 K T ER0
PROCLAIM P R OW0 K L EY1 M
PROCLAIMED P R OW0 K L EY1 M D
PROCLAIMING P R OW0 K L EY1 M IH0 NG
PROCLAIMS P R OW0 K L EY1 M Z
PROCLAMATION P R AA2 K L AH0 M EY1 SH AH0 N
PROCLAMATIONS P R AA2 K L AH0 M EY1 SH AH0 N Z
PROCLIVITIES P R OW0 K L IH1 V AH0 T IY0 Z
PROCLIVITY P R OW0 K L IH1 V AH0 T IY0
PROCONSUL P R OW0 K AA1 N S AH0 L

CMU Pronunciation Dictionary

01-Jan-14,-15.6,-8.9,0.1
02-Jan-14,-17.7,-15.1,0.1
03-Jan-14,-23.4,-13.1,0
04-Jan-14,-12.7,-2.5,0
05-Jan-14,-3.7,-1.2,19.1
06-Jan-14,-19.6,-2.1,7.7
07-Jan-14,-26.1,-18.7,1.5
08-Jan-14,-19.1,-11.1,0
09-Jan-14,-22.2,-8.3,0
10-Jan-14,-8.3,2.4,0
11-Jan-14,0.3,5.4,26.4
12-Jan-14,-0.8,1.3,0
13-Jan-14,0.4,5.8,0.2
14-Jan-14,-2.5,3.3,0
15-Jan-14,-8.5,-0.4,1.4
16-Jan-14,-8.7,-4,2.7
17-Jan-14,-8,-0.3,3.9
18-Jan-14,-10.1,-4.6,1.7

Reading text

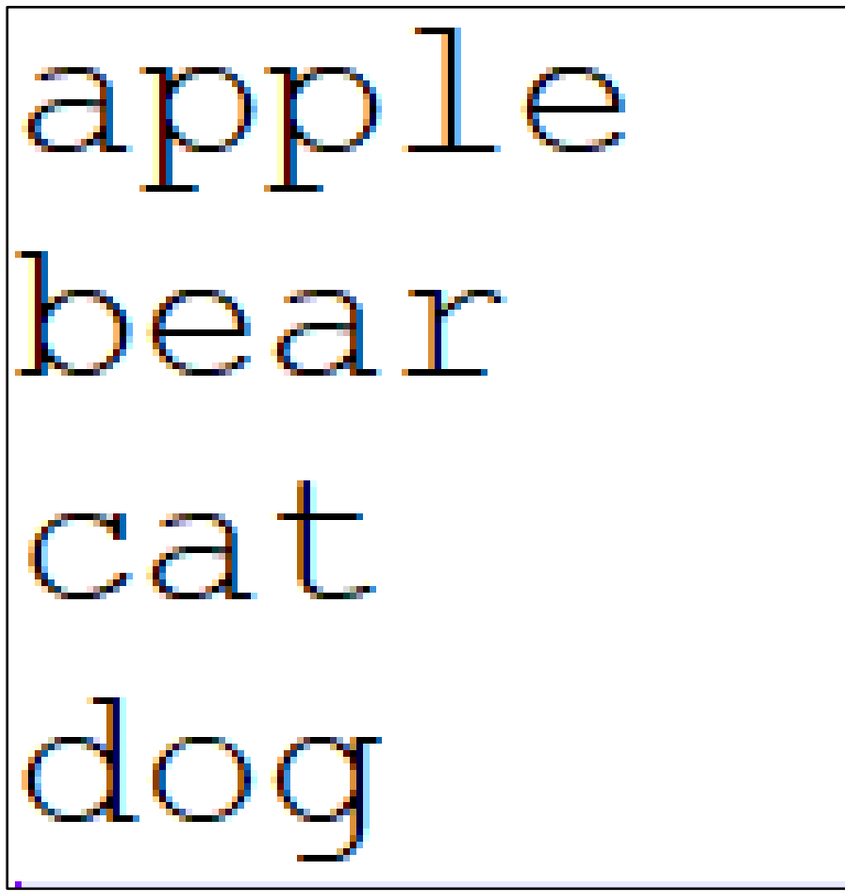
Reading text from a file can be quite painful in many programming languages. JavaScript p5 keeps it simple:

```
myArr = loadStrings(filename);
```

Load a text file, break it up into lines and **return an array of Strings**, one string for each line in the file.

Read and Display text

```
let lines = [];
function preload() {
  lines = loadStrings( "mywords.txt" );
}

function setup() {
  createCanvas( 600, 600 );
  textSize(24);
}

function draw() {
  background(220);
  for (let i = 0; i < lines.length; i++) {
    text( lines[i], 10, 30 + (i * 30));
  }
}
```



https://openprocessing.org/sketch/1068033

Shopping List

```
let lines = [];

function preload() {
  lines = loadStrings("shoppi
}

function setup() {
  createCanvas(600, 600);
  textSize(24);
}

function draw() {
  background(220);
  for (let i = 0; i < lines.length; i++) {
    text(lines[i], 10, 30 + (i * 30));
  }
}
```

```
apple   2  kg
banana  5  kg
potato  4  kg
onion   2  kg
```

https://openprocessing.org/sketch/1068049

Breaking up lines

A line in a file may contain lots of individual chunks of data separated by whitespace. We'd like to break lines into words, just as we broke files into lines. Two techniques: (see example next slide)

```
.split()
splitTokens()
```

Turn a line of text into an array of "words" (any non-whitespace characters separated by whitespace).

(Note that join() can reassemble individual strings into a single result.)

Breaking up lines

```
function setup() {
  let s = "hello out there";
  print(s); // hello out there

  let myArr1 = [];
  myArr1 = s.split(" ");
  print(myArr1); // myarr1 is of length 3
                 // ["hello", "out", "there"]

  let myArr2 = [];
  myArr2 = splitTokens(s, " ");
  print(myArr2); // myarr2 is of length 3
                 // ["hello", "out", "there"]
}
```

Sometimes we need strings converted to numbers

Assume we have a shopping list and we want to calculate the total weight of the items in the list.

```
apple  2 kg
banana 5 kg
potato 4 kg
onion  2 kg
```

Total KG

```
let lines = [];
let words = [];
let totalKG = 0;

function preload() {
    lines = loadStrings("shoppinglist.txt");
}

function setup() {
    createCanvas(600, 600);
    textSize(24);
    background(220);

    for (let i = 0; i < lines.length; i++) {
        words = splitTokens(lines[i], " ");
        text(lines[i], 10, 30 + (i * 30));
        totalKG = totalKG + int(words[1]);
    }
    text("Total KG: " + totalKG, 10, 30 + (lines.length * 30));
}
```

apple 2 kg
banana 5 kg
potato 4 kg
onion 2 kg
Total KG: 13

https://openprocessing.org/sketch/1068060

SpeedReader Example

- Read in a text file.
- Make one big long list (array) of “words”
  - Words may contain punctuation in this example
- Display one word at a time

SpeedReader

```
let lines = [];
let words = [];
let index = 0;

function preload() {
  lines = loadStrings("marley.txt");
}

function setup() {
  createCanvas(400, 200);
  textSize(50);
  textAlign(CENTER);
  fill(255);
  let allLines = join( lines, " ");
  words = splitTokens(allLines, " ");
  frameRate(1);
}

function draw() {
  background( 80 );
  text(words[index], width/2, height/2);
  index = (index + 1) % words.length;
}
```

https://openprocessing.org/sketch/1068096

marley.txt

- Marley was dead: to begin with. There is no doubt whatever about that. The register of his burial was signed by the clergyman, the clerk, the undertaker, and the chief mourner. Scrooge signed it: and Scrooge's name was good upon 'Change, for anything he chose to put his hand to. Old Marley was as dead as a door-nail.Mind! I don't mean to say that I know, …….

Question about marley.txt

- How many times does the word “the” occur?
  - Get rid of or ignore punctuation

words = splitTokens(allLines, “ .:;<>?!@#$%^&*()");

- Capitalization does matter (“the” and “The” both count)

let wordInUpperCase = words[i].toUpperCase();

Count Occurrences of “The”

```
let lines = [];
let words = [];
function preload() {
   lines = loadStrings("marley.txt");
}
function setup() {
   createCanvas(400, 200);

   let allLines = join(lines, " ");
   words = splitTokens(allLines, " ");

   let count = 0;
   for (let i = 0; i < words.length; i++) {
      let wordInUpperCase = words[i].toUpperCase();
      if (wordInUpperCase === "THE") {
         count = count + 1;
      }
   }
   print("The number of occurrences of \"The\" is: " + count);
}
```

https://openprocessing.org/sketch/1068117

Writing text to a File

We know we can use print() to send any text to the console.

A similar mechanism puts text into the file.

```
saveStrings(list, textFilename);

save(list, textFilename);
```

Writing to a Text File

```
let words = 'apple bear cat dog';
let list = [];

function setup() {
    createCanvas(100, 100);
    background(200);
    text('click here to save', 10, 10, 70, 80);
    list = split(words, ' ');
}

function mousePressed() {
        save(list, 'nouns.txt');
}
```

https://openprocessing.org/sketch/1068192

Reasons to write text

Logging: Create a permanent record of the behaviour of the program to review later.

Persistence: Store information about the program's state in an external file so that the sketch can restart with that state later.

Workflow: create text output that can be read by another program for further processing.

Sprite Sheets data: Text file provides x and y values and more for sprite files.

Reading and writing images

Loading and Display an Image

```
let img;

function preload() {
  img = loadImage( "HockeyPlayer.jpg" );
}

function setup() {
  createCanvas( 800, 400 );
   img.resize(250, 0);
}

function draw() {
  background(255);
  imageMode(CORNER);
  image(img, mouseX, mouseY);
}
```

https://openprocessing.org/sketch/1068218

Image Loading Idiom

```
let img;

function preload() {

    img = loadImage("some_image.jpg");

}

function setup() {

   …

}

function draw() {

  image(img, 0, 0, width, height);

  …

}
```

Writing images

Take a screenshot.

```
save("filename.png");
```

Save the contents of the sketch window to an image with the given file name.

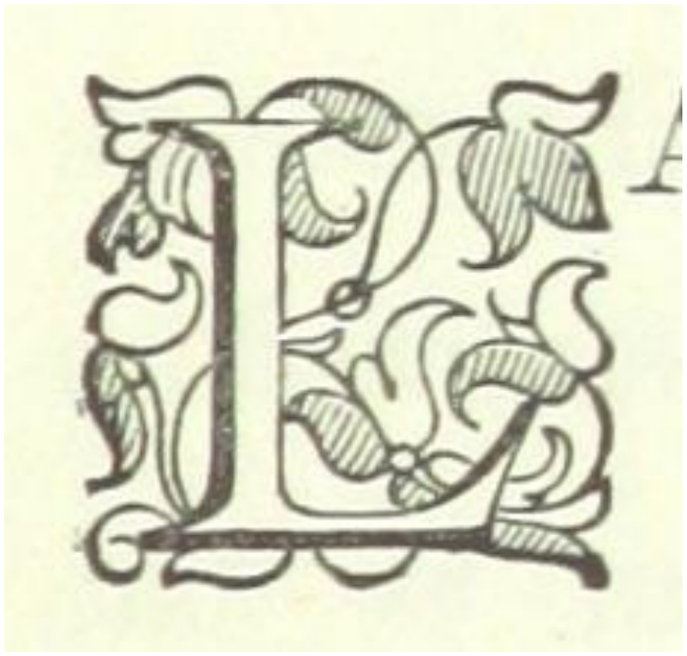
```
function keyPressed() {
  if (key === 's') {
    save("screen.png");
  }
}
```

Save an Image

```
let img;

function preload() {
  img = loadImage( "HockeyPlayer.jpg" );
}

function setup() {
  createCanvas( 800, 400 );
   img.resize(250, 0);
}

function draw() {
  background(200);
  imageMode(CORNER);
  image(img, mouseX, mouseY);
}

function keyPressed() {
  if (key === 's') {
    save("screen.png");
  }
}
```

https://openprocessing.org/sketch/1068269

Reading illustrations



Raster image: represented using a grid of pixels.

Vector illustration: represented using geometric paths.

Raster image: represented using a grid of pixels.
JPG, PNG, GIF, BMP, TIFF, …

Vector illustration: represented using geometric paths.
PDF, EPS, AI, SVG, …
Note that JavaScript p5 can load and display vector graphics, but it cannot save them.

Load a vector Image

```
let tiger;
function preload() {
  tiger = loadImage("tiger.svg");
}
function setup() {
  createCanvas(600, 600);
}
function draw() {
  background(220);
  image(tiger, 0, 0);
}
```

https://openprocessing.org/sketch/1068282

Moustachify Trump

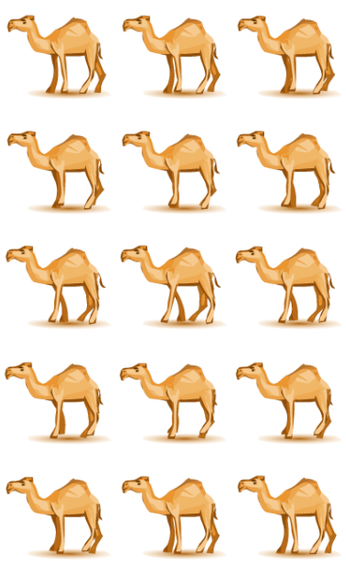
```
let face;
let stache;

function preload() {
  face = loadImage("Trump.jpg");
  stache = loadImage("stache.svg");
}
function setup() {
  createCanvas(face.width, face.height);
  noCursor();
}
function draw() {
  image(face, 0, 0);
  image(stache, mouseX, mouseY, stache.width / 2, stache.height / 2);
}

function keyPressed() {
  save("TrumpWithStache.png");
}
```

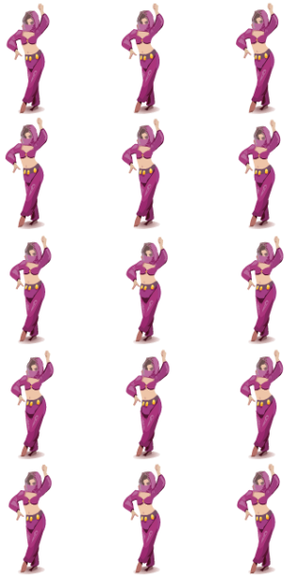
https://openprocessing.org/sketch/1068294

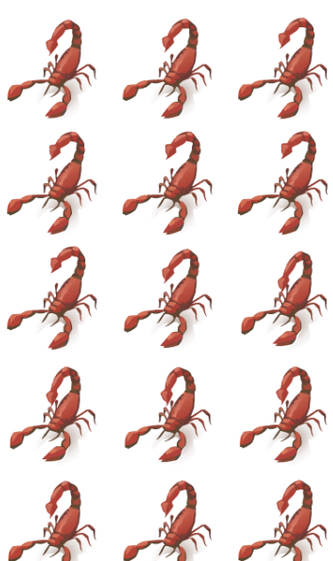
Using Sprite Files

Sprites

For CS106 we'll start with some sprites that are owned by uWaterloo
These 7 Sprites png files all have the exact same layout





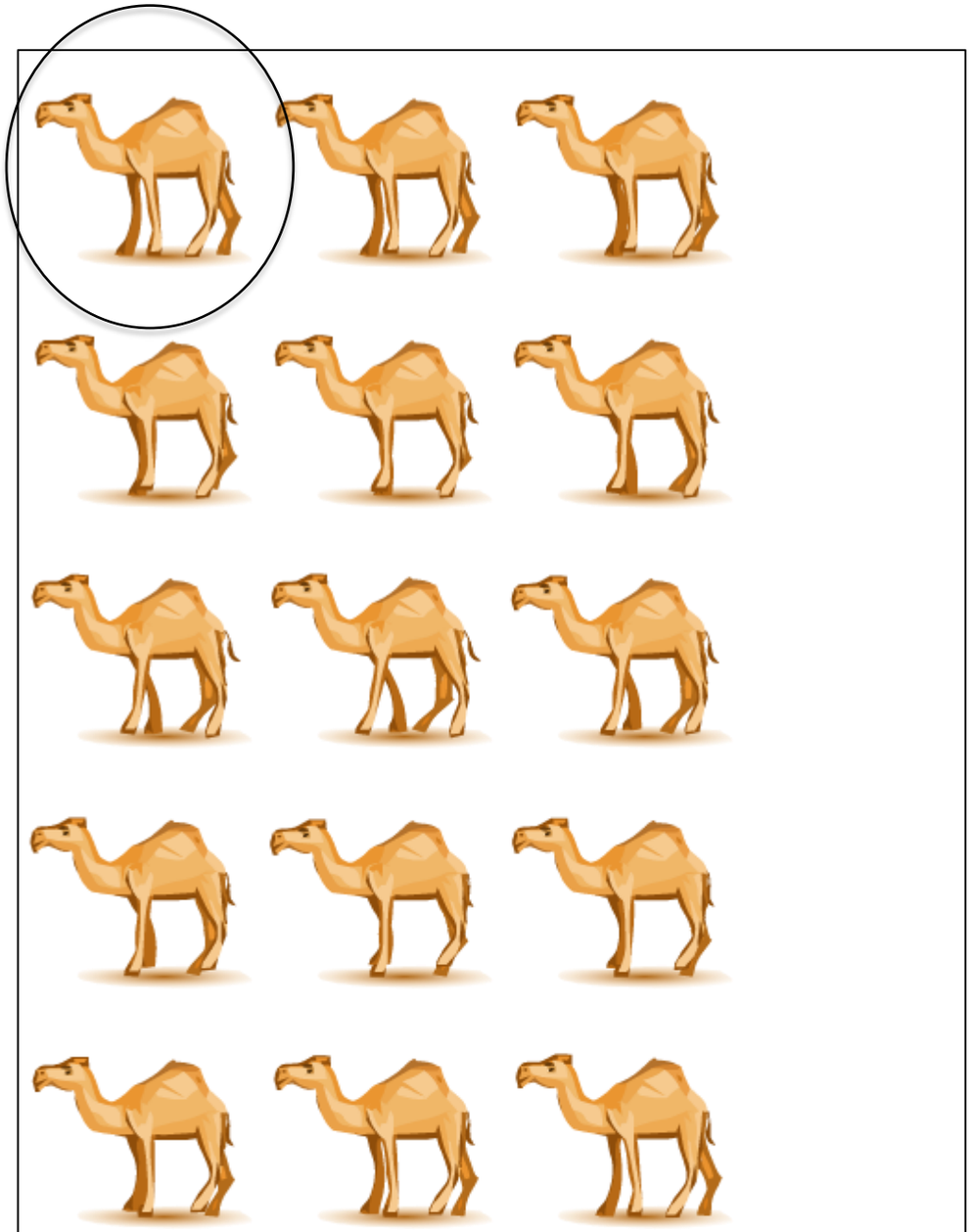


Text File Describing all Sprites on Previous Slide

```
1, 2, 2, 128, 128
2, 132, 2, 128, 128
3, 262, 2, 128, 128
4, 2, 132, 132, 128
5, 132, 132, 128, 128
6, 262, 132, 128, 128
7, 2, 262, 128, 128
8, 132, 262, 128, 128
9, 262, 262, 128, 128
10, 2, 392, 128, 128
11, 132, 392, 128, 128
12, 262, 392, 128, 128
13, 2, 522, 128, 128
14, 132, 522, 128, 128
15, 262, 522, 128, 128
```

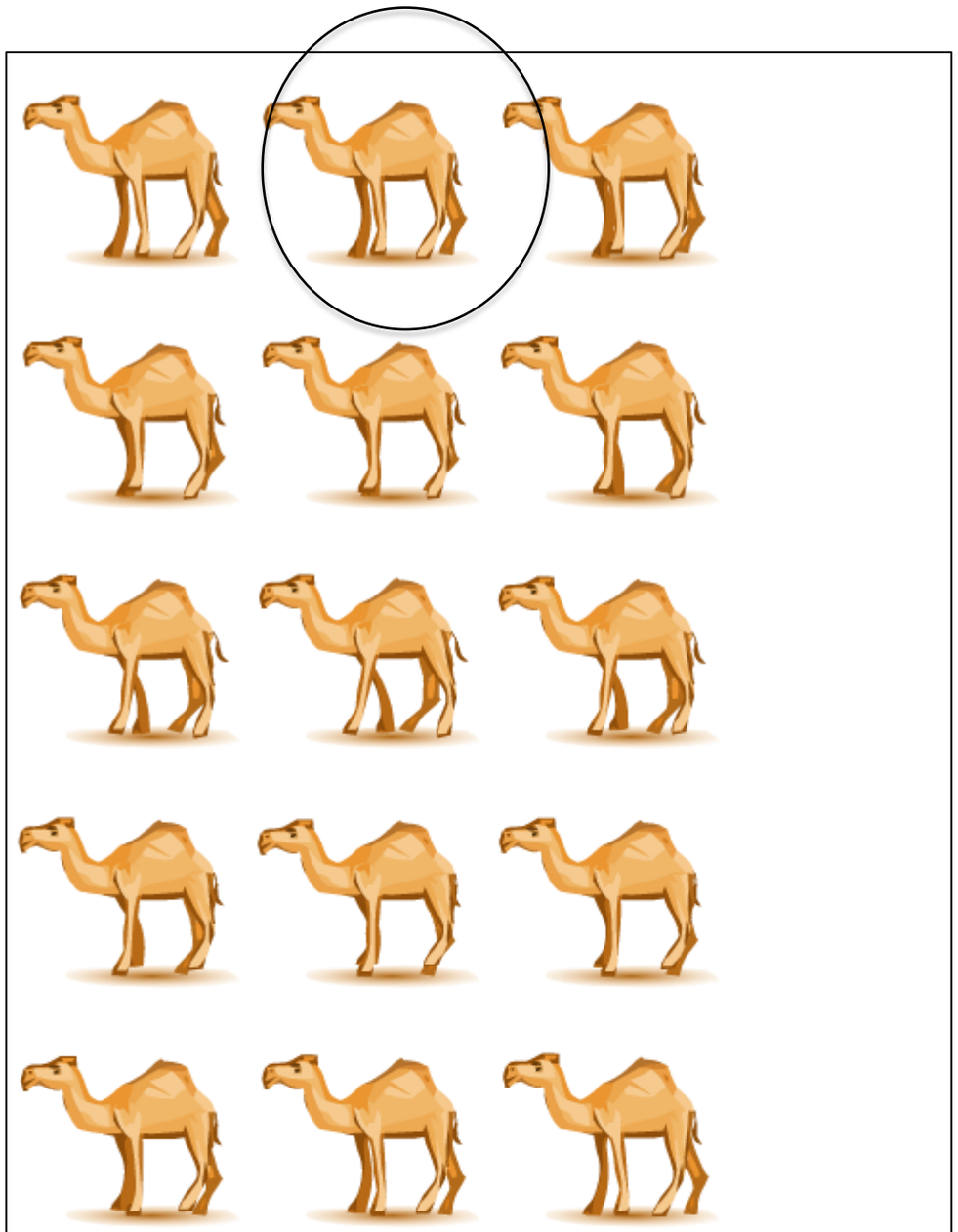
- Each row describes one image on the sprite
- There are 15 images and thus 15 rows
- Rows are numbered in column 1 but that doesn't matter to us. We can ignore the numbers 1-15 in column 1
- Columns 2 and 3 are the x and y locations of each graphic on the slide.
- Columns 3 and 4 are the width and height of each of the 15 graphics. All graphics on the sprite happen to be 128x128.

Sprite: camelAtlas.png



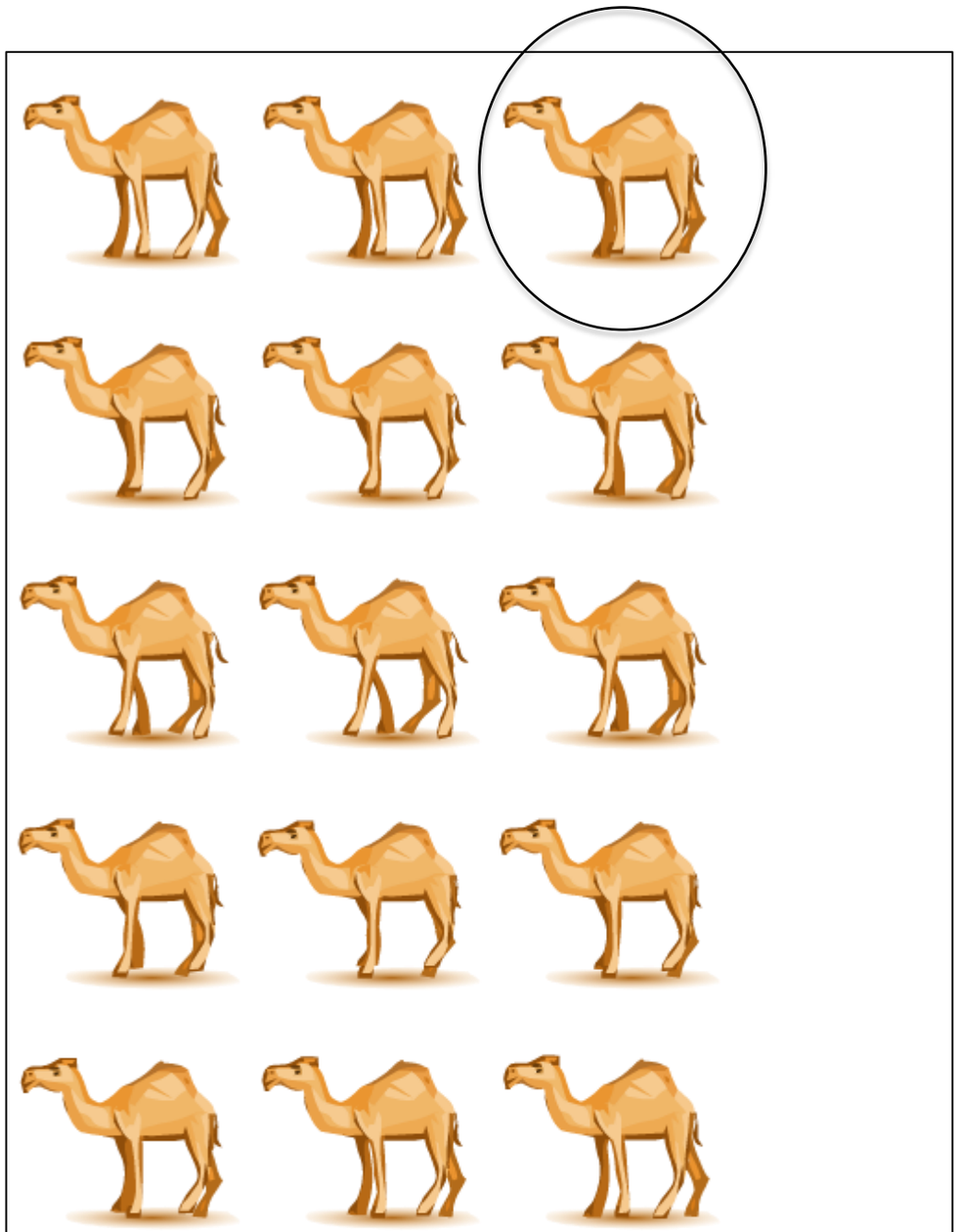
```
1, 2, 2, 128, 128
2, 132, 2, 128, 128
3, 262, 2, 128, 128
4, 2, 132, 132, 128
5, 132, 132, 128, 128
6, 262, 132, 128, 128
7, 2, 262, 128, 128
8, 132, 262, 128, 128
9, 262, 262, 128, 128
10, 2, 392, 128, 128
11, 132, 392, 128, 128
12, 262, 392, 128, 128
13, 2, 522, 128, 128
14, 132, 522, 128, 128
15, 262, 522, 128, 128
```

Sprite: camelAtlas.png



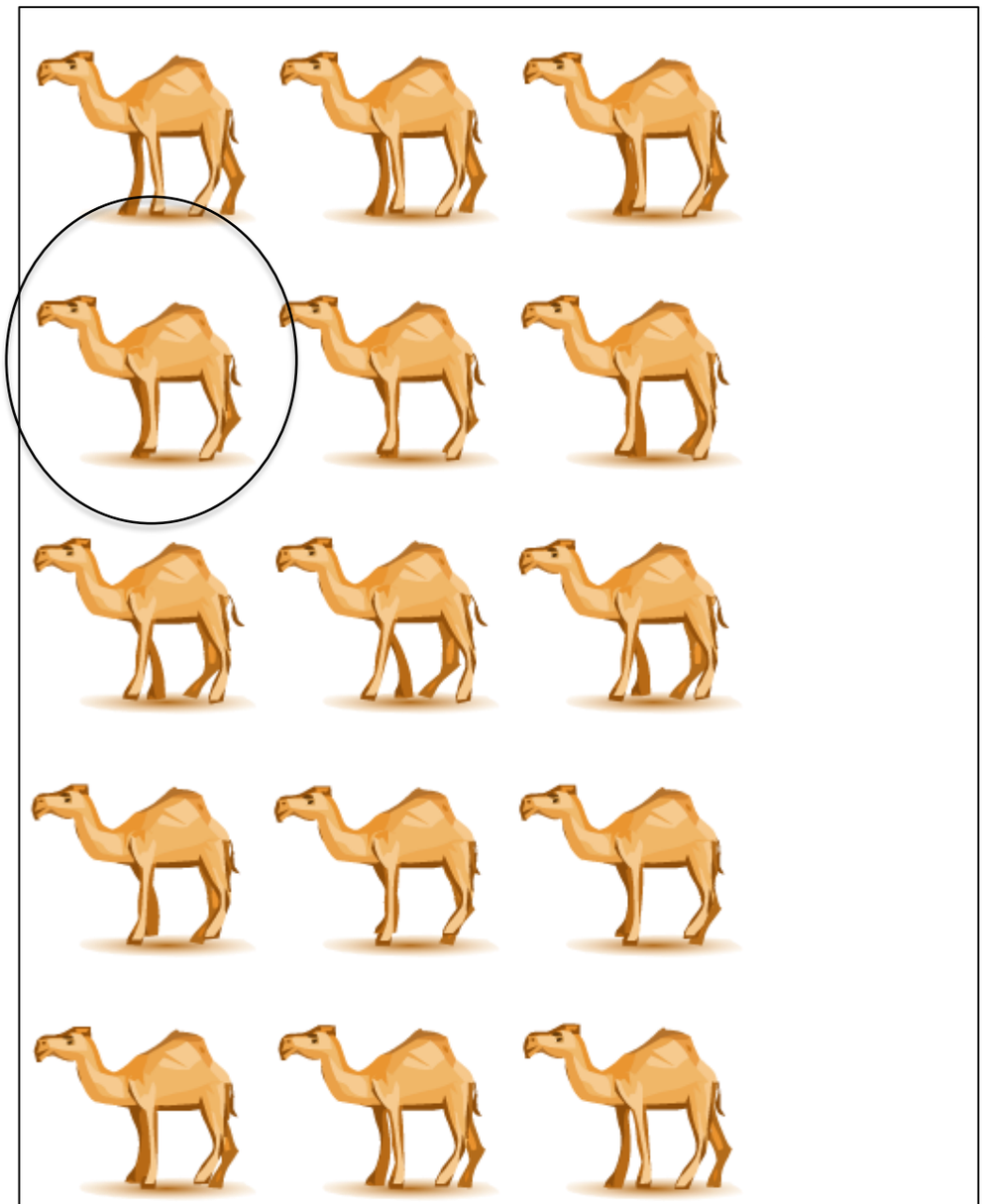
```
1, 2, 2, 128, 128
2, 132, 2, 128, 128
3, 262, 2, 128, 128
4, 2, 2, 132, 128
5, 132, 132, 128, 128
6, 262, 132, 128, 128
7, 2, 262, 128, 128
8, 132, 262, 128, 128
9, 262, 262, 128, 128
10, 2, 392, 128, 128
11, 132, 392, 128, 128
12, 262, 392, 128, 128
13, 2, 522, 128, 128
14, 132, 522, 128, 128
15, 262, 522, 128, 128
```

Sprite: camelAtlas.png



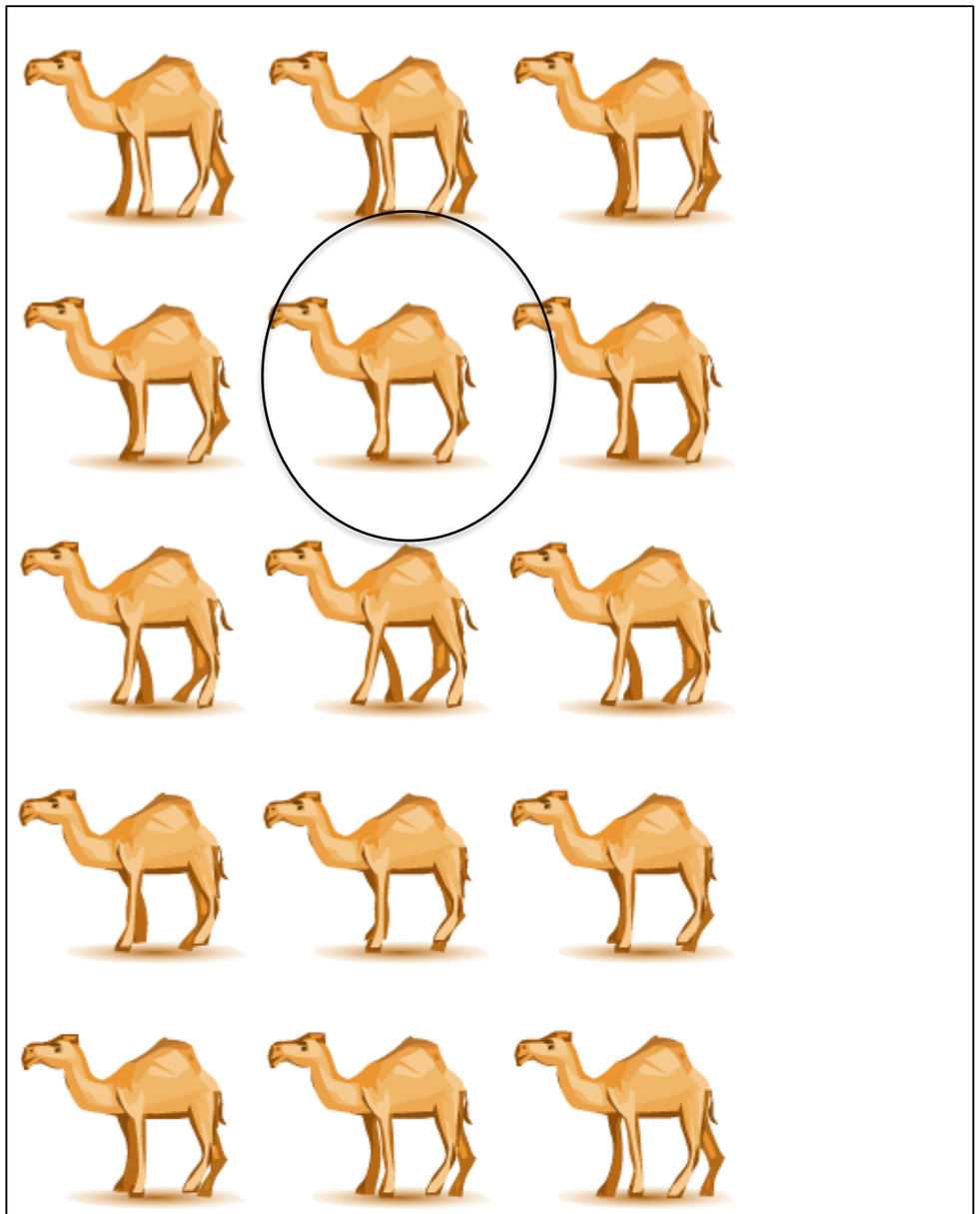
```
1, 2, 2, 128, 128
2, 132, 2, 128, 128
3, 262, 2, 128, 128
4, 2, 132, 132, 128
5, 132, 132, 128, 128
6, 262, 132, 128, 128
7, 2, 262, 128, 128
8, 132, 262, 128, 128
9, 262, 262, 128, 128
10, 2, 392, 128, 128
11, 132, 392, 128, 128
12, 262, 392, 128, 128
13, 2, 522, 128, 128
14, 132, 522, 128, 128
15, 262, 522, 128, 128
```

Sprite: camelAtlas.png



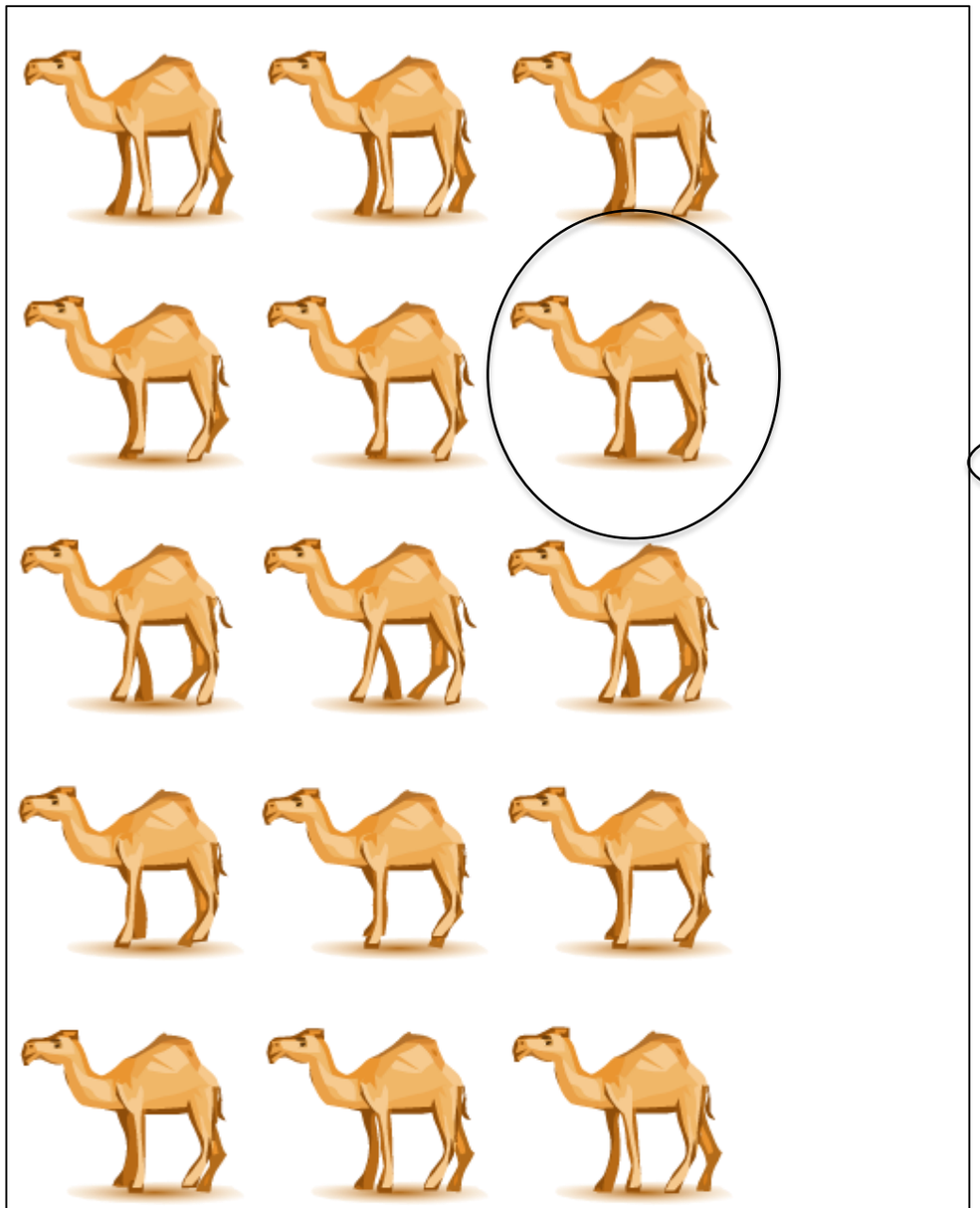
```
1, 2, 2, 128, 128
2, 132, 2, 128, 128
3, 262, 2, 128, 128
4, 2, 132, 132, 128
5, 132, 132, 128, 128
6, 262, 132, 128, 128
7, 2, 262, 128, 128
8, 132, 262, 128, 128
9, 262, 262, 128, 128
10, 2, 392, 128, 128
11, 132, 392, 128, 128
12, 262, 392, 128, 128
13, 2, 522, 128, 128
14, 132, 522, 128, 128
15, 262, 522, 128, 128
```

Sprite: camelAtlas.png



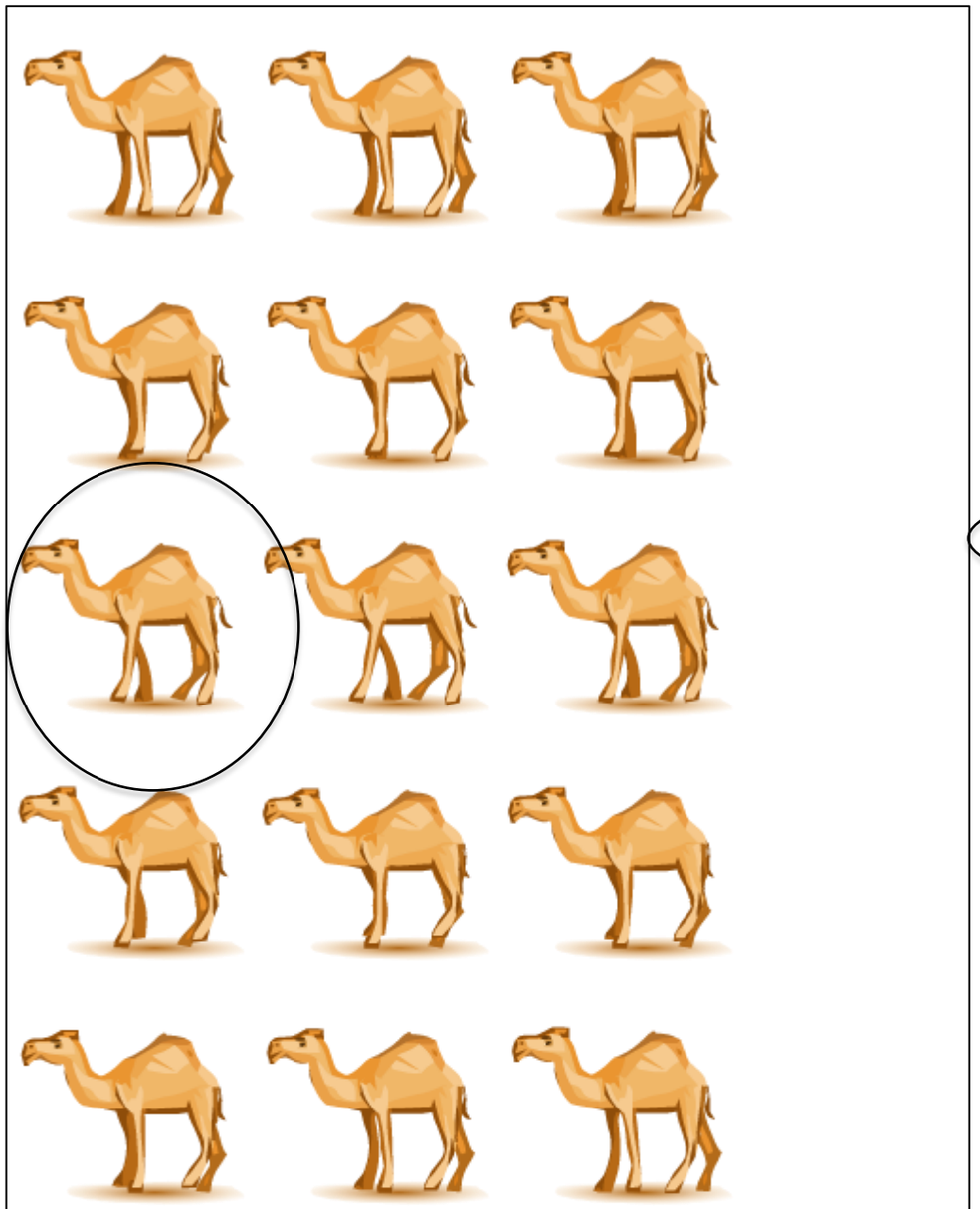
```
1, 2, 2, 128, 128
2, 132, 2, 128, 128
3, 262, 2, 128, 128
4, 2, 132, 132, 128
5, 132, 132, 128, 128
6, 262, 132, 128, 128
7, 2, 262, 128, 128
8, 132, 262, 128, 128
9, 262, 262, 128, 128
10, 2, 392, 128, 128
11, 132, 392, 128, 128
12, 262, 392, 128, 128
13, 2, 522, 128, 128
14, 132, 522, 128, 128
15, 262, 522, 128, 128
```

Sprite: camelAtlas.png



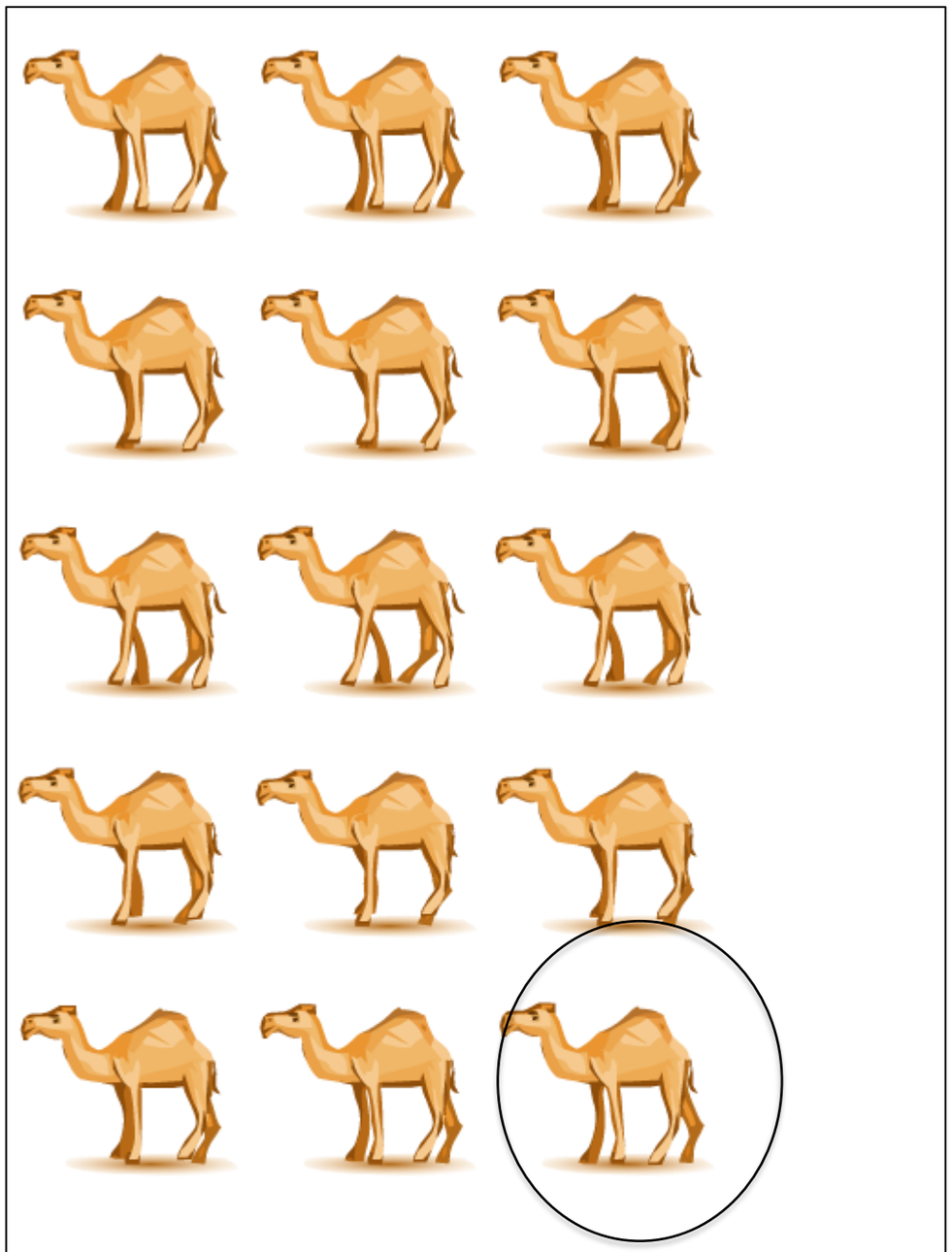
```
1, 2, 2, 128, 128
2, 132, 2, 128, 128
3, 262, 2, 128, 128
4, 2, 132, 132, 128
5, 132, 132, 128, 128
6, 262, 132, 128, 128
7, 2, 262, 128, 128
8, 132, 262, 128, 128
9, 262, 262, 128, 128
10, 2, 392, 128, 128
11, 132, 392, 128, 128
12, 262, 392, 128, 128
13, 2, 522, 128, 128
14, 132, 522, 128, 128
15, 262, 522, 128, 128
```

Sprite: camelAtlas.png



```
1, 2, 2, 128, 128
2, 132, 2, 128, 128
3, 262, 2, 128, 128
4, 2, 132, 132, 128
5, 132, 132, 128, 128
6, 262, 132, 128, 128
7, 2, 262, 128, 128
8, 132, 262, 128, 128
9, 262, 262, 128, 128
10, 2, 392, 128, 128
11, 132, 392, 128, 128
12, 262, 392, 128, 128
13, 2, 522, 128, 128
14, 132, 522, 128, 128
15, 262, 522, 128, 128
```

Sprite: camelAtlas.png



```
1, 2, 2, 128, 128
2, 132, 2, 128, 128
3, 262, 2, 128, 128
4, 2, 132, 132, 128
5, 132, 132, 128, 128
6, 262, 132, 128, 128
7, 2, 262, 128, 128
8, 132, 262, 128, 128
9, 262, 262, 128, 128
10, 2, 392, 128, 128
11, 132, 392, 128, 128
12, 262, 392, 128, 128
13, 2, 522, 128, 128
14, 132, 522, 128, 128
15, 262, 522, 128, 128
```

Sprite: chestAtlas.png

Described by the same data file as camelAtlas.png.

```
1, 2, 2, 128, 128
2, 132, 2, 128, 128
3, 262, 2, 128, 128
4, 2, 132, 132, 128
5, 132, 132, 128, 128
6, 262, 132, 128, 128
7, 2, 262, 128, 128
8, 132, 262, 128, 128
9, 262, 262, 128, 128
10, 2, 392, 128, 128
11, 132, 392, 128, 128
12, 262, 392, 128, 128
13, 2, 522, 128, 128
14, 132, 522, 128, 128
15, 262, 522, 128, 128
```

Sprite: chestAtlas.png (slide 1 of 2)

```
let spriteSheet;
let spriteDataFile = [];
let fifteenImages = [];
let lines = [];
let index = 0;
let row = [];


function preload() {
    spriteSheet = loadImage("chestAtlas.png");
    spriteDataFile = loadStrings("commonData15Graphics.txt");
}

function setup() {
    createCanvas(windowWidth, windowHeight);
    background(100);
    initialization();
    frameRate(1);
}
```

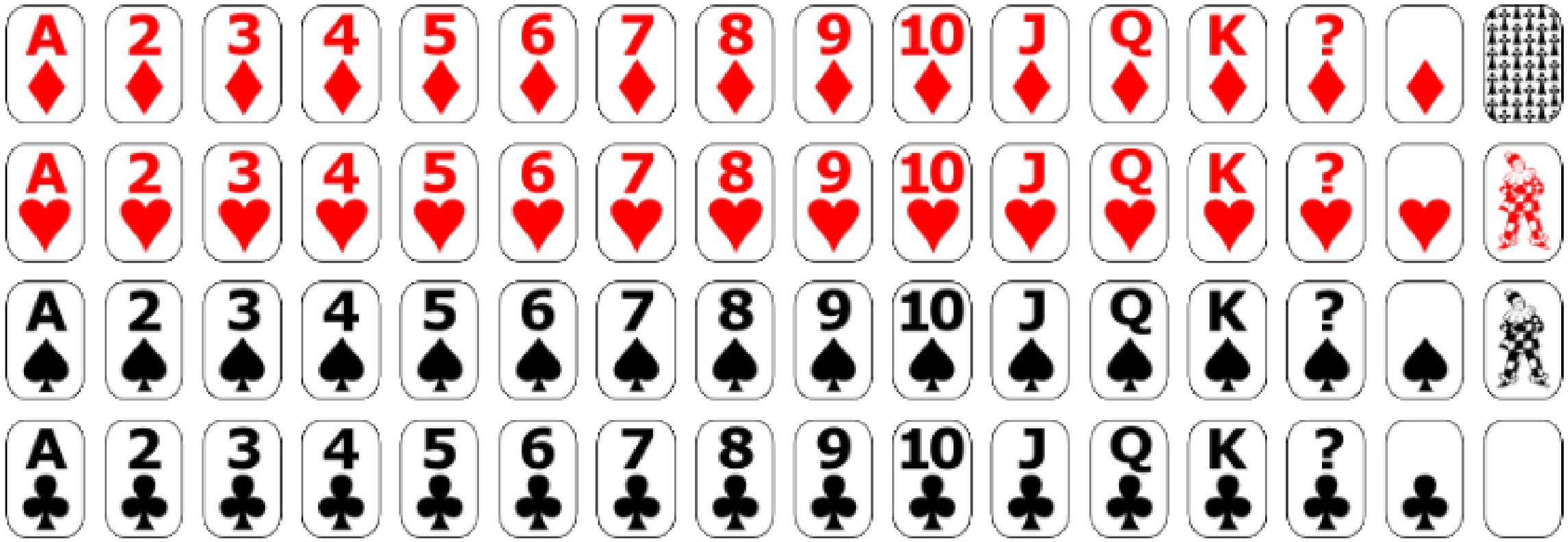
Sprite: lampAtlas.png (slide 2 of 2)

```
function draw() {
    background(220);
    print(index);
    image(fifteenImages[index], 0, 0);
    text(index, 62, 62);
    index = index + 1;
    if (index === fifteenImages.length)
        index = 0;
}
```

```
function initialization() {
    textSize(36);
    for (let i = 0; i < spriteDataFile.length; i++) {
        row = spriteDataFile[i].split(",");
        fifteenImages[i] = spriteSheet.get(row[1], row[2], row[3], row[4]);
    }
}
```

https://openprocessing.org/sketch/1071241

Sprite: Playing Cards



Text File with Data about the Sprite file for the Deck of Cards

cardData.txt

1,0,0,51.25,70.25

2,51.25,0,51.25,70.25

3,102.5,0,51.25,70.25

4,153.75,0,51.25,70.25

5,205,0,51.25,70.25

6,256.25,0,51.25,70.25

7,307.5,0,51.25,70.25

8,358.75,0,51.25,70.25

9,410,0,51.25,70.25

10,461.25,0,51.25,70.25

11,512.5,0,51.25,70.25

12,563.75,0,51.25,70.25

13,615,0,51.25,70.25

There are 52 rows

Each Row Contains:

- Sequential Number (1-52)
- X position on the sprite
- Y position on the sprite
- Width on the sprite
- Height on the sprite

Load and Display the Cards (slide 1 of 2)

```
let cardDeck;
let cardData;
let cardImages = [];
let lines = [];
let index = 0;
let row = [];

function preload() {
   cardDeck = loadImage("cards.png");
   cardData = loadStrings("cardData.txt");
}

function setup() {
   createCanvas(windowWidth, windowHeight);
   background(100);
   initialization();
   frameRate(5);
}
```

Load and Display the Cards (slide 2 of 2)

```
function draw() {
   background(220);
   image(cardImages[index], 0, 0);
   index = (index + 1) % (cardData.length - 1);
   ellipse(mouseX, mouseY, 20, 20);
}

function initialization() {
   for (let i = 0; i < cardData.length - 1; i++) {
      row = cardData[i].split(",");
      cardImages[i] = cardDeck.get(row[1], row[2], row[3], row[4]);
   }
}
```

Sprites

Here are a few more sprites. They don't have corresponding data files.
A data file for each would need to be a slight modification of the sprite data files for camel, lamp, etc.