Variables & Datatypes
CSE 120 Spring 2017

Instructor: Justin Hsia
Teaching Assistants: Anupam Gupta, Braydon Hall, Eugene Oh, Savanna Yee
Administrivia

- Assignments:
  - Taijitu due today (4/5)
  - Reading Check 2 due tomorrow (4/6)
  - Custom Logo due Friday (4/7)

- No “big ideas” lecture this week
  - More time on programming
Lab: Custom Logo

drawing canvas: 400 x 220

colors: purple, gold, white (bg)
Lab: Custom Logo

```cpp
/* uw_logo.pde
    Created by Justin Hsia

    UW logo made out of rectangles in school colors.
*/

size(400,220);  // drawing canvas of 400x220
background(255); // white background

// The letter 'U' in purple
fill( 75, 47, 131); // purple fill
rect( 20, 20, 40, 180); // left side of U
rect( 65, 140, 40, 60); // middle base of U
rect(110, 20, 40, 180); // right side of U

// The letter 'W' in gold
fill(183, 165, 122); // gold fill
rect(160, 20, 40, 180); // left segment of W
rect(205, 140, 40, 60); // left base of W
rect(250, 100, 40, 90); // middle segment of W
rect(295, 140, 40, 60); // right base of W
rect(340, 20, 40, 180); // right segment of W
```
Drawing a Square

- [See Demo on Panopto]
Variables

- Piece of your program that holds the value of something
  - Every variable must be given a name and a datatype
- The values of these variables can change (i.e. vary) during the execution of your program
  - **Warning**: Not like a variable in Algebra (i.e. an unknown)

- **Assignment**: give a variable a specific value
  - *e.g.* \( x = 12; \)
- **Read**: use the current value of a variable
  - *e.g.* \( y = x + 1; \)
Datatypes

- **int**: integers
- **float**: decimal/real numbers
- **color**: a triple of numbers representing RGB
- **boolean**: true or false

Many more exist and can be found in the Processing Reference:
Declarations

- We **declare** a variable by telling Processing the variable’s datatype, followed by the variable’s name:

```plaintext
int x;
float half;
color yellow;
```

- You can also give a variable a starting value (**initialization**) in the same line as the declaration:

```plaintext
int x = 4;
float half = 0.5;
color yellow = color(255, 255, 0);
```
Drawing a Square with Variables

- [See Demo on Panopto]
Variable Rules & Guidelines

- Variables are case-sensitive
  - e.g. `leftside` is not the same as `leftSide`

- Variable names are meaningless to computers, but meaningful to humans
  - Choosing informative names improves readability and reduces confusion

- In this class, most of our variables will be declared and initialized at the very top of our programs
Variable Manipulation

- Executed sequentially, just like other statements

- For variable assignments, compute right-hand side \textit{first}, then store result in variable

- \textbf{Example: } \texttt{int x = 4;}

- \texttt{x = x + 1;}

  1) Read the current value of \( x \) (4) for the right-hand side

  2) Add 1 to the current value of \( x \)

  3) Store the result (5) back into \( x \)
Variable Practice

1) int x = 1;
   int y = 2;
   int z = 3;
   x = x + 1;
   y = y - 1;
   z = z + 2;

2) int x = 7;
   int y = 2;
   int z = 0;
   x = x + 3;
   y = y - 2;
   z = x + y;

3) int x = -1;
   int y = 0;
   int z = 5;
   x = x + z;
   y = y - x;
   z = x + z;

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
TMNT: Donatello

```cpp
donatello
define size(500,500);
define noStroke();
define background(255,245,220);

// Donatello
define fill(0,100,0); // dark green
define rect(230,182,40,15); // top of head
define fill(88,44,141); // purple
define rect(230,197,40,6); // bandana mask
define fill(0,100,0); // dark green
define rect(230,203,40,20); // bottom of head
define fill(219,136,0); // dark yellow
define rect(230,223,40,50); // shell
define fill(0,100,0); // dark green
define rect(230,273,40,45); // lower body
```
Donatello with a Variable

```java
int x_pos = 100; // x-position

size(500,500);
noStroke();
background(255,245,220);

// Donatello
fill(0,100,0);    // dark green
rect(x_pos,182,40,15); // top of head

fill(88,44,141); // purple
rect(x_pos,197,40,6); // bandana mask

fill(0,100,0); // dark green
rect(x_pos,203,40,20); // bottom of head

fill(219,136,0); // dark yellow
rect(x_pos,223,40,50); // shell

fill(0,100,0); // dark green
rect(x_pos,273,40,45); // lower body
```
Donatello with Motion

\[
\text{run setup( )}
\]

\[
\text{run draw( )}
\]
Stopping Motion

- Stop Donatello from running off the right side of the screen:
  \[ x_{\text{pos}} = \min(x_{\text{pos}} + 1, 460); \]

- Stop Donatello from running off the left side of the screen:
  \[ x_{\text{pos}} = \max(x_{\text{pos}} - 1, 0); \]
Falling Into Place

- Introduce variables for each body segment:

```java
int head_pos = 0;       // head position
float mask_pos = 15;    // mask position
int face_pos = 21;      // face position
float body_pos = 41;    // body position
int leg_pos = 91;       // leg position
```

- Update each variable at different speeds:

```java
head_pos = min(head_pos + 3, 364);
mask_pos = min(mask_pos + 3.5, 379);
face_pos = min(face_pos + 4, 385);
body_pos = min(body_pos + 4.5, 405);
leg_pos = min(leg_pos + 5, 455);
```
Falling Into Place

- Update y-positions of drawing based on new variables:

```cpp
// Donatello
fill(0,100,0);  // dark green
rect(x_pos,head_pos,40,15);  // top of head
fill(88,44,141);  // purple
rect(x_pos,mask_pos,40,6);  // bandana mask
fill(0,100,0);  // dark green
rect(x_pos,face_pos,40,20);  // bottom of face
fill(219,136,0);  // dark yellow
rect(x_pos,body_pos,40,50);  // shell
fill(0,100,0);  // dark green
rect(x_pos,leg_pos,40,45);  // lower body
```
Summary

- Variables are named quantities that can vary during the execution of a program.

- Datatypes specify different forms of data:
  - e.g. int, float, color, Boolean

- Variable *declarations* specify a variable datatype and name to the program:
  - Generally occurs at the top of the program.

- `min()` and `max()` functions can be used to limit or stop change in a variable value.