**Programming for Artist 1**

**Functions:**

**Build in functions:**

void setup(){} //Will run only once.

// Is automatically called.

// set the size of your window here.

void draw(){} //executes after the setup.

// is automatically called.

//by default, the framerate is 60 but you can change that.

void mousePressed(){} //automatically called.

void keyPressed(){} //only called when mouse or key is pressed.

**Random function:**

Float rnd = random(0-50);

You can give it a minimum and maximum number. **The maximum number is never included**. So the max is actually 49.999 and not 50.

If you want so save a random value in an int you can use the random function inside the int function and use explicit conversion.

Int rnd = int(random(3.14,13));

**Create your own functions:**

Create the function:

void drawTree(){} //make the name clear.

//make sure it is not in another void function. So don’t put it in draw or setup.

Put whatever you want in this function. You must call the function for processing to read it and use it.

Call the function (in draw):

drawTree();

**With parameters:**

You can add variables that are specific to the function you made. When you call on this function you can add whatever value in those variables. This means u can for example draw the same thing in different sizes without having to go back in the code of the function.

First the function asks what it needs.

void drawPetals (int centerX, int centerY, int circleSize){}

anything between the brackets are called **parameters**

the programmer passes on the values he/she wants the parameters to have.

In draw:

drawPetals(200,150,30);

drawPetals(50,200,18);

**Variables:**

**Global:** variables on top of the script and you can use them everywhere in the code. (with some restrictions but those are not important now.

**Local:** variables that are local to the function. It only exist and works in the function is was created. This include normal variables and parameter variables.

**System Variables:**

width //width of the window

height //height of the window

mouseX //x position of your mouse in the window

mouseY // y position of your mouse in the window

**Create your own variable**

**Data(variable) types:**

int //used to store integers, without a decimal point. Can be positive or negative. The default value is 0

float //used to store floating-point number, with a decimal point. Can be positive and negative. Default value is 0.0

double // for floating-point numbers larger than those that can be stored in a float. Can be positive and negative. It uses up a lot of memory. We don’t really use it, the default is 0.0

char //used to store a single character. A char stores letter and symbols in the Unicode format. The ASCII table. You can put the char in an int and it will have the value that is associated with the letter on the ASCII table. Single quates and 1 character.

String //Used to store a sequence of characters. String greeting = “Hello World”;

use capital S and double quates.

color //used to store color values. Use the color function to fill the variable:

color name = color(RGB) or color(0/255) color name = #FFFF;

or u can use hexadecimal without having to use the function

Boolean //used to store values that are either true or false. Default = false

**declare a variable:**

type name = value; //the first time a variable gets a value it is called ***initialization.***

Int evilboy = 5;

**Explicit Data conversion**

Explicit conversion using a type

Int myNumber = int someVar;

Will only work if someVar is a char, float, double or int

Does not work for string.

**type Casting (use this one more it is better overall)**

type casting is explicit conversion using a function.

Int myNumber = int (someVar);

It will show up as blue and has brackets

Works with any primitive datatype

println ( “Hurray, my first line of code!”);

size(100,100); //sets the window size , width and height

Fullscreen(); // Sets the window size to full screen

background(0); // changed the color of the background 0=black 255=white

framerate(); //decides how many times something happens within 1 sec.

// do this in setup

// keep in mind that the framerate also means how many times a function is checked. For example if you press your mouse and it only checks 1 time a second, you will get a delayed response.

Drawing primitive shapes:

point(); //draws a point(x,y)

ellipse(); // draws an ellipse(x,y,w,h)

rect(); //draws a rectangle(x,y,w,h) you can add rounded corners by adding values, check reference

line(); //draws a line between 2 points (x1,y1,x2,y2)

triangle(); //draw a triangle with 3 points (x1,y1,x2,y2,x3,y3)

quad(); //draws a quad with 4 points (x1,y1,x2,y2,x3,y3,x4,y4

arc(); //(x,y,w,h,startAngle,stopAngle)

By default shapes are drawn from a certain point. For example, rect is drawn from the top left corner. You can use modes like rectMode to change that default to Center

rectMode(CENTER);

noFill(); //hides the fill of aa shape

fill(); //fill shapes with a color of choice

strokeWeight(); //changes the thickness of the lines

noStroke(); //removes the stroke/line altogether

stroke(); //unhides the stroke and gives the stroke a color of choice

color(); //0=black, 255 = white Anything in between = gray

//R,G,B 0-255 and hexadecimal #

//you can give it an extra value for alpha making it transparent. //From 0-255

//make sure to set the background in draw if you want transparency

//you can use tools color selector to pick any color you want

**Arithmetics:**

Remember that **/** , **\*** and **%** are calculated before + and -.

Example: 2+2\*3

**DO!:** 2\*3 = 6 6+2=8

**DON’T!:** 2+2=4 4\*3 = 12 **THIS IS INCORECT**

Int addition = number1 + number2;

Int subtraction = number1 – number2;

Int multiplication = number1 \* number2;

Int division = number 1 / number2;

Int remainder = number1 % number2;

**Compound assignment operator:**

posX = posX + 4; 🡪 posX += 4; //increase by 4

posX = posX – 4; 🡪 posX -= 4; //decrease by 4

posX = posX \*4; 🡪 posX \*= 4; //multiple by 4

posX = posX / 4; 🡪 posX /= 4; //divide by 4

posX = posX % 4; 🡪 posX %= 4; //remainder

posX ++; //increase by exactly 1

posX --; //decrease by exactly 1

When you calculate with ints and the answer is a float number it will cut of all decimal number. It won’t round it out it will just cut them off.

When calculating with floats and ints together it is a little tricky. when u store a value in a float at least 1 of the numbers used when calculating has to be a float. If you use 2 ints and save it in a float. It will technically not raise an error but you won’t get the decimal numbers it will say .0 at the end.

You will get an error when you try to save float into an int. you cannot calculate with floats and try and save it into an int. int/int will always be an int and trying to save it into a float will give problems. You can use data conversion to help with these kind of problems.

**animation:**

when u want to animate something make 2 functions

make a drawing function and an update function. The update function is used to actually animate.

**Modulo operator:**

You use a modulo% to reset an animation.

Modulos calculate the remainder of division

Int remainder= 5 % 4;

5 cupcakes for 4 toddlers. Give each 1 a cupcake. You have 1 cupcake left.

5%4 = 1

Int remainder = 3%4;

3 cupcakes for 4 toddlers. We don’t have enough to give every toddler a cupcake so we are going to keep all 3

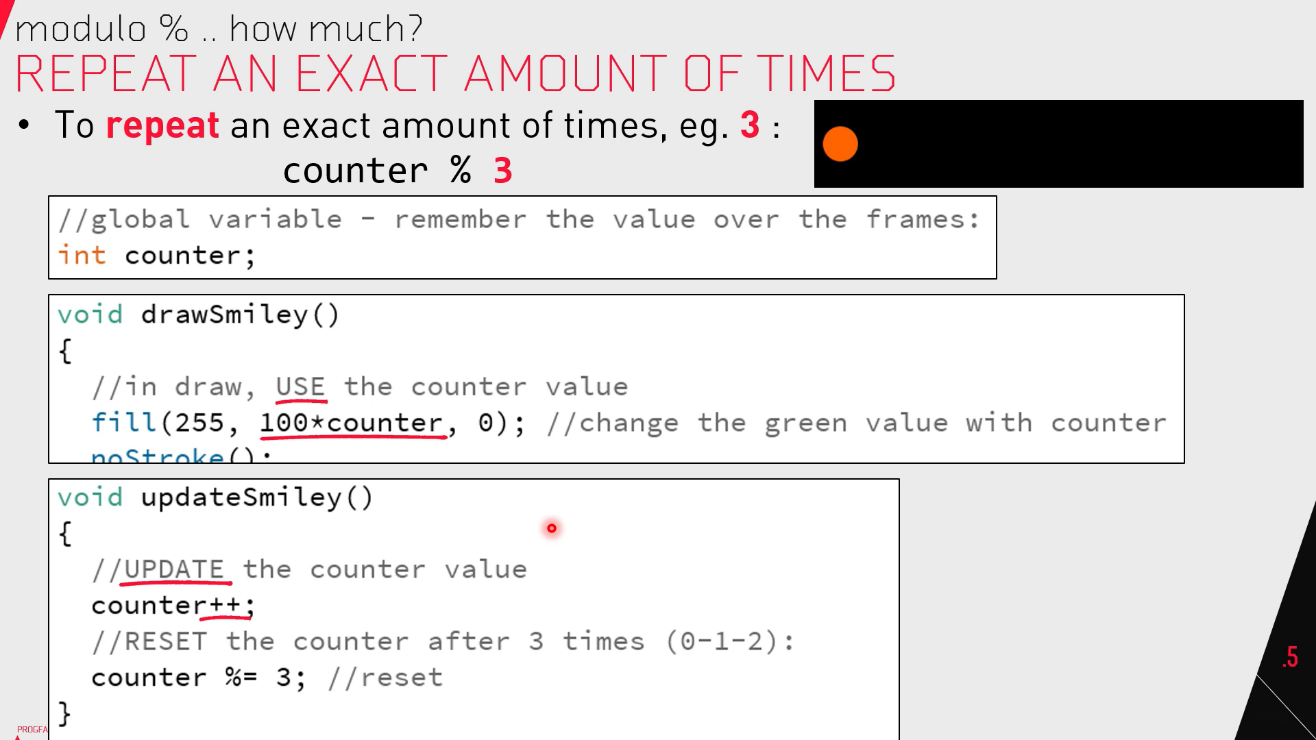
3%4=3

4%4=0

8%4=0

Resetting /using the modulo





**Images:**

Make sure you add the images u need in the data folder.

Hit ctrl K to open your file location. In the sketch location need to be a folder called data. Make sure its lower case.

Another way is to just drag the images into the sketch. It will create the data folder automatically and put the images in there.

**Load the images:**

When naming images it is highly recommended to start the name with img for image.

Create the image variable. Using the PImage type

PImage imgleaf

Loaf the images in setup. Don’t do it in draw or it is going to give a big delay.

ImgLeaf = loadImage(“bananaleaf.png”);

You can also add a url in the brackets but it is not advised since it cost a lot of memory.

You loaded in the image but it won’t show until you draw the image.

**Draw the image:**

Image(imgLeaf,x,y); or image(imgLeaf,x,y, width,height);

Instead of using the width and height in the image function you can resize the image.

imgLeaf.resize(150,0);

this will now have resized the image so that when you draw it it will be that size. if you make 1 of the values 0 it will scale proportionately to the other value given.

you can ask for the width and height of an image to use it for code.

imgLeaf.width; and imgLeaf.height;

you can also only draw part of an image.

Create an image variable and give it a name. put the image u want a part of into that variable .

Pimage imgPart = imgLeaf.get(x,y,w,h);

**Fonts & Text:**

If you have a font file add it to the data folder as explained by images.

You can create a font using tools and then create font.

Choose a size for the font and give the file a name. click ok and it will generate a .vlw font file for you.

If you then go to the data folder it will be in there.

Create the PFont variable.

PFont mainFont;

Do this next step in setup not in draw!

**Loading a font:**

Use a vlw file that is in ur data folder

maineFont = loadFont(“Ariel.vlw”);

this is the same way you load an image. You don’t have to give a font size since you already gave it a size when you created the font. Notice how it says .vlw at the end.

**Creating a font:**

Use a font that is already on your computer.

mainFont = createFont(“Ariel”,18);

18 is the size you want the font to be.

**Use the font**

Set the font:

textFont(MainFont);

use the font:

text(“Hello world”,x,y);

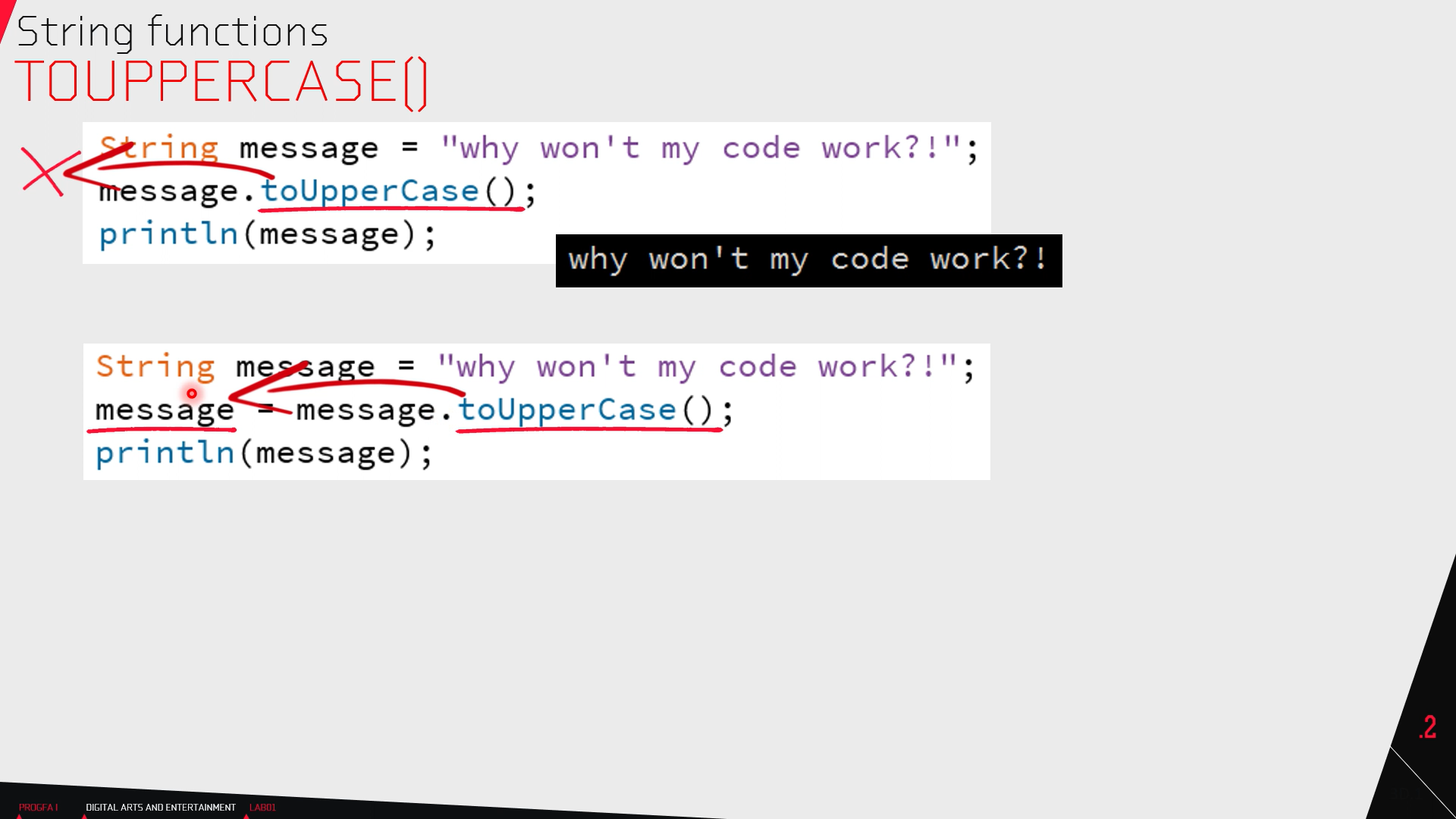
Keep text alignment in mind. To change the alignment you do not use MODE u use ALLIGN

textAlign(CENTER,TOP); //(Horizont,Vertical)

if you draw a text to your screen, draw the background so it doesn’t end up being pixelated.

**String Functions:**

When working with String functions you have to catch the code. You have to change the existing message into the changed message.



Message.toUpperCase(); //converts the string to upper case

Message.toLowerCase(); //converts the string to lower case

Name.substring(); //3 print the string from index 3

//0,3 print the index from 0 to 3 of the string

Char character = name.charAt(0); //gives back the char at index 0 of the string

Int index = name.indexOf(“Dae”); //gives the index number where he finds Dae in the String.

int numChars = name.lenght(); //gives the number of characters in the String.

you can link functions together.called chaning

Forexample:

String part = name.substring(0,3).toUpperCase();

is the same as:

String part = name.substring(0,3);

Part = part.toUpperCase();

**Use key input to build String:**

Make a global String variable. It needs to be global because you use it in draw and keypressed.

String needs to have something or you will get a nul pointer exception. You can put an empty string in there.

String input (“”);

In draw:

Text(input,10,height/2);

In keypressed:

//use += because we want to add to the existing string.

Input+= key;

//now we can type but you run into trouble with shift. This is however not discussed in week 4.