LUDWIG-
MAXIMILIANS-
UNIVERSITÄT
MÜNCHEN

Fakultät für Betriebswirtschaft
Munich School of Management

Basics in Programming for MMT

Session 3 – Loops and Lists

www.mmt.bwl.lmu.de

Scope of the Session

1. Repetition

- Datatypes
- If and Else
- Colors
- Commands

2. Theory

- Loops
- For
- Strings and Characters
- Arrays
- Processing Variables

3. Next

- Session 4

4. Tutorial

- Reference
- Moving Circle
- Define arrays that store x and y coordinated of multiple circles
- Loops in Loops?
- What else?

Repetition

Repetition

Datatypes

- If we declare variables, we have to specify their types.
- Different datatypes require different space in the working memory.

Integer	`int i = 10;`
Float	`float f = 3.33;`
Boolean	`boolean b = false;`

Repetition

If and Else

- Based on a **condition**, we can execute specific code sections.
- `if` the condition is `true`, execute `{...}`. `else` execute `{***}`

```
void draw () {
    background(0);
    x = x+1;

    if (x>100) {...}
    else {***}

    rect(x,200,200,200);
}
```

Repetition

Colors

- Colors are either entered as gray values or RGB values.
- The number of arguments specifies the color type.
- Each color channel can take values from **0-255**.

Gray

```
background(0);
fill(123);
stroke(255);
```

RGB

```
background(255,0,0);
fill(0,255,0);
stroke(0,0,255);
```

Repetition

Commands

- The **name** of a command specifies what the computer should do.
- The **arguments** are values the command processes.
- Each command is ended with a **semicolon**.

`rect`	`(x,y,w,h)`	`;`
Name	Arguments	End

Theory

Theory

Loops (1/4)

- What do I have to do to draw three random rects?
- Duplicate commands!

```
float x;
float y;

void setup () {
    size(600,600);
}

void draw () {
    background(0);

    x = random(600);
    y = random(600);
    rect(x,y,200,200);
}
```

Theory

Loops (2/4)

- What do I have to do to draw three random rects?
- Duplicate commands!

```
float x;
float y;

void setup () {
    size(600,600);
}

void draw () {
    background(0);

    x = random(600);
    y = random(600);
    rect(x,y,200,200);

    x = random(600);
    y = random(600);
    rect(x,y,200,200);

}
```

Theory

Loops (3/4)

- What do I have to do to draw 100 random rects?
- Too much copy/paste …
- Reoccurring code is called redundant.
  We want to **avoid redundancy**.
- This is always the same.
  Can we **automate** the process?

```
float x;
float y;

void setup () {
    size(600,600);
}

void draw () {
    background(0);

    x = random(600);
    y = random(600);
    rect(x,y,200,200);

    x = random(600);
    y = random(600);
    rect(x,y,200,200);

}
```

Theory

Loops (4/4)

- `for` marks a loop which repeats the commands inside the `{}` as long as the statement defined in the `()` is `true`.
- The counter defined in the `()` is increased in the end of each loop by the calculation defined.

```
float x;
float y;

void setup () {
    size(600,600);
}

void draw () {
    background(0);

    for (int i=0; i<100; i=i+1) {
          x = random(600);
          y = random(600);
          rect(x,y,200,200);
    }
}
```

Theory

For

`for`	`(int i=0;`	`i<100;`	`i=i+1)`	`{...}`
	Start: Initial value of the counter	**End:** What is the maximum value of the counter?	**Steps:** How to increment after each loop	**Body:** Commands to be performed

Theory

Strings and Characters

- If we want to store symbols, words or sentences, we cannot use `int` of `float` variables.
- We use `char` for symbols such as letters, numbers and punctuation marks
- and `Strings` for whole words or sentences.
- You can combine `Strings` with the `+` operator.

```
char c1 = 'a';
char c2 = '.';
char c3 = '7';
char c4 = ' ';

String s1 = "hello";
String s2 = "world!";
String s3 = s1 + " " + s2;
```

Theory

Arrays (1/6)

- If we have many variables of the same type, we can store them in **container**.
- These containers are called **arrays** and can contain objects of a certain type.

```
int a;
int b;
int c;
int d;
int e;
int f;
...
```

Theory

Arrays (2/6)

Declare:

```
int [] a;
```

Init and assign:

```
void setup () {
    size(600,600);
    a = new int [3];}
    a[0] = 357;
    a[1] = 123;
    1[2] = 142;
}
```

Read:

```
void draw () {
    background(a[0],a[1],a[2]);
}
```

Theory

Arrays (3/6)

`int []`	`a`	`;`
Datatype: Array of integers	**Name**	End

Theory

Arrays (4/6)

```
a = new int [3] ;
```

Initialize:
Create `new` array that can contain 3 integer values.

Theory

Arrays (5/6)

```
a[1]          = 100 ;
```

Assign:
Assign a value to
the array entry with index 1.
First index is
always 0.

Theory

Arrays (6/6)

```
int x = a[0]          ;
```

Read:
To read a value from an array, you also have to specify the index in the `[]`.

Theory

Processing Variables

`width`	Width of sketch, defined in `size()`
`height`	Height of sketch, defined in `size()`
`mouseX`	Current mouse pointer position (x) according to the coordinate system in the sketch
`mouseY`	Current mouse pointer position (y) according to the coordinate system in the sketch

Next

Next

Session 4

- Define own commands
- void?

Tutorial

Tutorial

Reference

1. Go to: processing.org/reference

2. Find commands for getting mouse-position.

Tutorial

Moving Circle

1. Write a sketch where a circle follows the mouse pointer.
2. What else can you use the mouseX and mouseY values for?

Tutorial

Define arrays that store x and y Coordinates of multiple Circles

Move each circle independently. Use arrays and loops to minimize code.

Tutorial

Loops in Loops?

1. How to draw a chessboard pattern?
2. Try to nest loops to iterate over 2 dimensions

Tutorial

What else?

What else can we do? Be creative!