

MIT-IIT Robotics Program

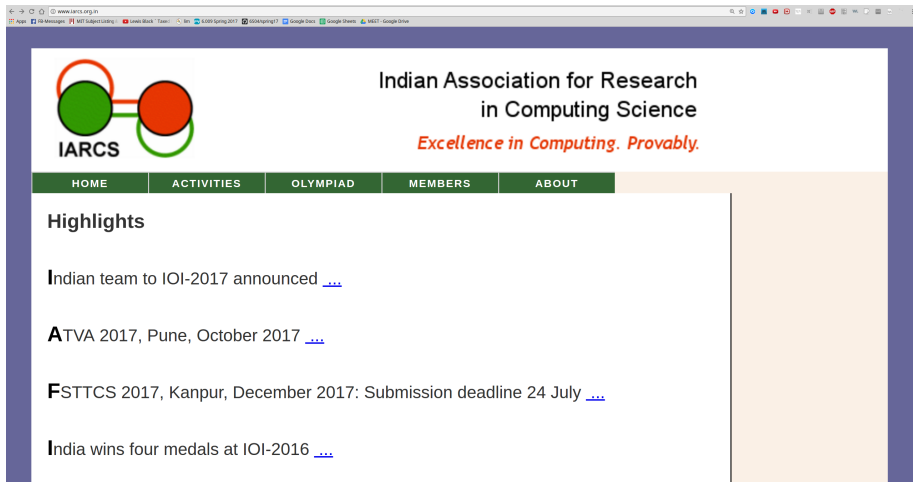
Logic Flow – Booleans, Logical & Relational Operators, Conditionals, While Loops

Amartya Shankha Biswas

June 5, 2017

Outline

- 1 Indian Computing Olympiad
- 2 Processing Graphics
 - Getting Started
 - Logic Flow
- 3 Functions
 - Definition
 - Examples
 - Defining Functions
- 4 More on Processing
 - Primitives
 - Coordinate System
 - Debugging
 - Interaction
 - Physics
 - Exercise



The screenshot shows a web browser window displaying the IARCS website. The browser's address bar shows 'www.iarcs.org.in'. The website has a purple header with the IARCS logo on the left, which consists of two overlapping circles (one green, one red) with lines connecting them. To the right of the logo, the text 'Indian Association for Research in Computing Science' is displayed in black, followed by the tagline 'Excellence in Computing. Provably.' in red. Below the header is a navigation bar with green buttons labeled 'HOME', 'ACTIVITIES', 'OLYMPIAD', 'MEMBERS', and 'ABOUT'. The main content area has a white background and features a 'Highlights' section with four entries, each followed by a blue link icon (three dots): 'Indian team to IOI-2017 announced', 'ATVA 2017, Pune, October 2017', 'FSTTCS 2017, Kanpur, December 2017: Submission deadline 24 July', and 'India wins four medals at IOI-2016'. The browser's taskbar at the bottom shows various open applications and the system clock indicating June 5, 2017, at 3:47.

www.iarcs.org.in

Indian Association for Research
in Computing Science

Excellence in Computing. Provably.

HOME ACTIVITIES OLYMPIAD MEMBERS ABOUT

Highlights

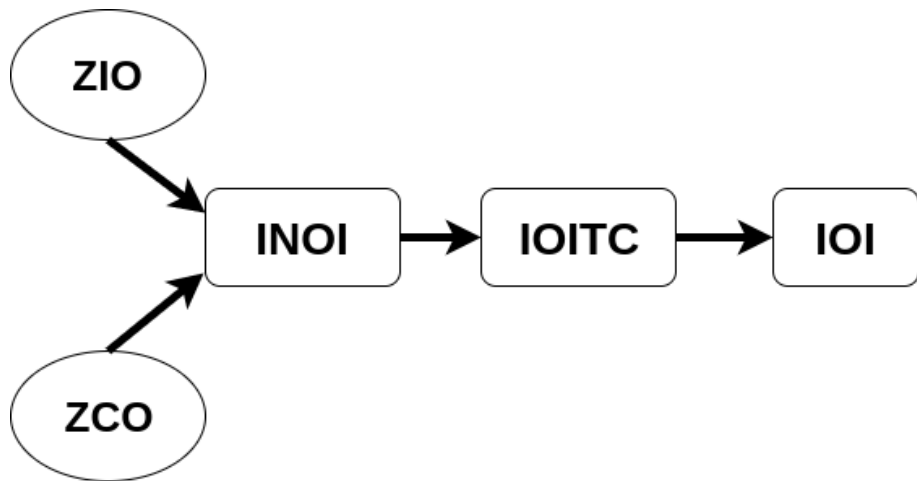
Indian team to IOI-2017 announced [...](#)

ATVA 2017, Pune, October 2017 [...](#)

FSTTCS 2017, Kanpur, December 2017: Submission deadline 24 July [...](#)

India wins four medals at IOI-2016 [...](#)

Levels



Outline

- 1 Indian Computing Olympiad
- 2 Processing Graphics
 - Getting Started
 - Logic Flow
- 3 Functions
 - Definition
 - Examples
 - Defining Functions
- 4 More on Processing
 - Primitives
 - Coordinate System
 - Debugging
 - Interaction
 - Physics
 - Exercise

Installing

```
fill(0); //Set fill color  
ellipse(a, b, c, d);
```

```
fill(0); //Set fill color  
ellipse(a, b, c, d);
```

a float: x-coordinate of ellipse
b float: y-coordinate of ellipse
c float: width of the ellipse
d float: height of the ellipse

Window Size

```
int displayWidth = 800;  
int displayHeight = 400;
```

Window Size

```
int displayWidth = 800;  
int displayHeight = 400;  
size(displayWidth, displayHeight);
```

Window Size

```
int displayWidth = 800;  
int displayHeight = 400;  
size(displayWidth, displayHeight);  
  
// The wrong way to specify  
// the middle of the screen  
ellipse(400, 200, 50, 50);
```

Window Size

```
int displayWidth = 800;
int displayHeight = 400;
size(displayWidth, displayHeight);

// The wrong way to specify
// the middle of the screen
ellipse(400, 200, 50, 50);

// Always the middle
// no matter how size() changes
ellipse(width/2, height/2, 50, 50);
```

Outline

- 1 Indian Computing Olympiad
- 2 Processing Graphics
 - Getting Started
 - Logic Flow
- 3 Functions
 - Definition
 - Examples
 - Defining Functions
- 4 More on Processing
 - Primitives
 - Coordinate System
 - Debugging
 - Interaction
 - Physics
 - Exercise



`run setup()`



Setup and Loop

```
int displayWidth = 800;  
int displayHeight = 400;
```

Setup and Loop

```
int displayWidth = 800;
int displayHeight = 400;

void setup () {
    size(displayWidth, displayHeight);
}
```


Setup and Loop

```
int displayWidth = 800;
int displayHeight = 400;

void setup () {
    size(displayWidth, displayHeight);
}

void draw () {
    background(255);
    fill(0);
    ellipse(width/2, height/2, 50, 50);
}
```

Outline

- 1 Indian Computing Olympiad
- 2 Processing Graphics
 - Getting Started
 - Logic Flow
- 3 **Functions**
 - **Definition**
 - Examples
 - Defining Functions
- 4 More on Processing
 - Primitives
 - Coordinate System
 - Debugging
 - Interaction
 - Physics
 - Exercise

What is a Function ?

A reusable block of code that performs a task.

What is a Function ?

A reusable block of code that performs a task.

- written by you and used by someone else
- written by someone else and used by you

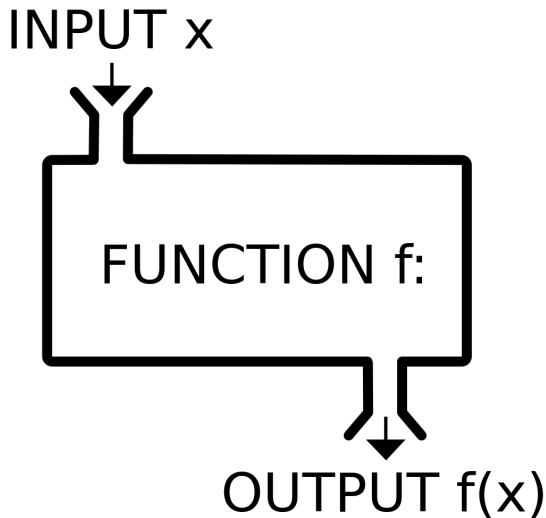
What is a Function ?

A reusable block of code that performs a task.

- written by you and used by someone else
- written by someone else and used by you

Don't need to know what the code looks like !

What is a Function ?



Outline

- 1 Indian Computing Olympiad
- 2 Processing Graphics
 - Getting Started
 - Logic Flow
- 3 **Functions**
 - Definition
 - **Examples**
 - Defining Functions
- 4 More on Processing
 - Primitives
 - Coordinate System
 - Debugging
 - Interaction
 - Physics
 - Exercise

You have already seen some functions.

- `main()`
- `sqrt(81)`
- `setup()`
- `draw()`
- `ellipse(50, 60, 10, 10)`
- `fill(255)`

The ellipse function.

Inputs to the ellipse() function

Coordinates of center. Size of ellipse.

The ellipse function.

Inputs to the ellipse() function

Coordinates of center. Size of ellipse.

What does the ellipse() function do ?

The ellipse function.

Inputs to the ellipse() function

Coordinates of center. Size of ellipse.

What does the ellipse() function do ?

Draws an ellipse on the screen, with specified parameters.

The ellipse function.

Inputs to the ellipse() function

Coordinates of center. Size of ellipse.

What does the ellipse() function do ?

Draws an ellipse on the screen, with specified parameters.

How ?

The ellipse function.

Inputs to the ellipse() function

Coordinates of center. Size of ellipse.

What does the ellipse() function do ?

Draws an ellipse on the screen, with specified parameters.

How ?

Who cares ? . . .

What do all functions have in common ?

- `main()`
- `sqrt(81)`
- `draw()`
- `ellipse(50, 60, 10, 10)`
- `fill(255)`

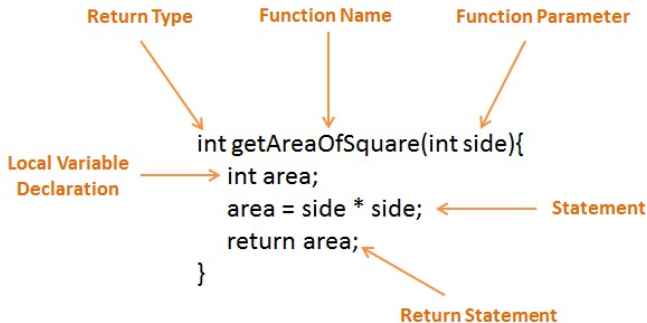
Parentheses ()

Outline

- 1 Indian Computing Olympiad
- 2 Processing Graphics
 - Getting Started
 - Logic Flow
- 3 **Functions**
 - Definition
 - Examples
 - **Defining Functions**
- 4 More on Processing
 - Primitives
 - Coordinate System
 - Debugging
 - Interaction
 - Physics
 - Exercise

The structure of a Function

Function Definition



techcrashcourse.com

How function works in C programming?

```
#include <stdio.h>

void functionName()
{
    ... ..
    ... ..
}

int main()
{
    ... ..
    ... ..

    functionName();

    ... ..
    ... ..
}
```

The diagram illustrates the control flow between a function and its caller. A horizontal arrow points from the closing brace of the `main()` function to the opening brace of the `functionName()` function, indicating the transfer of control to the function. A vertical arrow points from the closing brace of the `functionName()` function back to the line containing the `functionName();` call in the `main()` function, indicating the return of control to the caller.

Input to a Function

How to pass arguments to a function?


```
#include <stdio.h>

int addNumbers(int a, int b);

int main()
{
    ... ..

    sum = addNumbers(n1, n2);
    ... ..
}

int addNumbers(int a, int b)
{
    ... ..
    ... ..
}
```

Two arrows originate from the arguments 'n1' and 'n2' in the function call 'sum = addNumbers(n1, n2);' within the main function. The arrow from 'n1' points down and to the left to the parameter 'a' in the function definition 'int addNumbers(int a, int b)'. The arrow from 'n2' points down and to the left to the parameter 'b' in the function definition.

Return Value (output) of a Function

Return statement of a Function

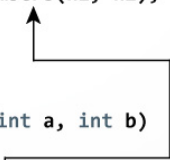
```
#include <stdio.h>

int addNumbers(int a, int b);

int main()
{
    ... ..
    sum = addNumbers(n1, n2);
    ... ..
}

int addNumbers(int a, int b)
{
    ... ..
    return result;
}
```

sum = result

A diagram consisting of two L-shaped lines. The first L-shape starts from the 'return result;' line in the addNumbers function, goes right and then up. The second L-shape starts from the 'sum = addNumbers(n1, n2);' line in the main function, goes left and then down. The two lines meet at a point, with an arrowhead pointing towards the main function, indicating the return of the value.

Exercise

Write a function `drawTarget()`, that takes x and y coordinate as input, and displays a target (concentric black and white circles) in that location.

Modify this function to take one integer N as input, and draw a target with N circles. This means that `drawTarget(5)` should display a target with 5 concentric circles.

Use this code to test.

```
void draw () {  
    if (mousePressed) {  
        drawTarget(mouseX, mouseY);  
        delay(200);  
    }  
}
```

Outline

- 1 Indian Computing Olympiad
- 2 Processing Graphics
 - Getting Started
 - Logic Flow
- 3 Functions
 - Definition
 - Examples
 - Defining Functions
- 4 More on Processing
 - Primitives
 - Coordinate System
 - Debugging
 - Interaction
 - Physics
 - Exercise

Window Size

```
int width = 800;  
int height = 400;
```

Window Size

```
int width = 800;  
int height = 400;  
size(width, height);
```

Window Size

```
int width = 800;  
int height = 400;  
size(width, height);  
  
// The wrong way to specify  
// the middle of the screen  
ellipse(400, 200, 50, 50);
```


Window Size

```
int width = 800;  
int height = 400;  
size(width, height);  
  
// The wrong way to specify  
// the middle of the screen  
ellipse(400, 200, 50, 50);  
  
// Always the middle  
// no matter how size() changes  
ellipse(width/2, height/2, 50, 50);
```

Rectangle

```
rect(a, b, c, d);  
rect(a, b, c, d, r);  
rect(a, b, c, d, tl, tr, br, bl);
```

Rectangle

```
rect(a, b, c, d);  
rect(a, b, c, d, r);  
rect(a, b, c, d, tl, tr, br, bl);
```

- a float: x-coordinate of rectangle
- b float: y-coordinate of rectangle
- c float: width of the rectangle
- d float: height of the rectangle

Rectangle

```
rect(a, b, c, d);  
rect(a, b, c, d, r);  
rect(a, b, c, d, tl, tr, br, bl);
```

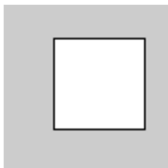
a float: x-coordinate of rectangle
b float: y-coordinate of rectangle
c float: width of the rectangle
d float: height of the rectangle
r float: radii for all four corners

Rectangle

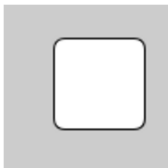
```
rect(a, b, c, d);  
rect(a, b, c, d, r);  
rect(a, b, c, d, tl, tr, br, bl);
```

a float: x-coordinate of rectangle
b float: y-coordinate of rectangle
c float: width of the rectangle
d float: height of the rectangle
r float: radii for all four corners
tl float: radius of top-left corner
tr float: radius of top-right corner
br float: radius of bottom-right corner
bl float: radius of bottom-left corner

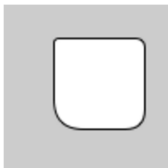
rect() Examples



```
rect(30, 20, 55, 55);
```



```
rect(30, 20, 55, 55, 7);
```

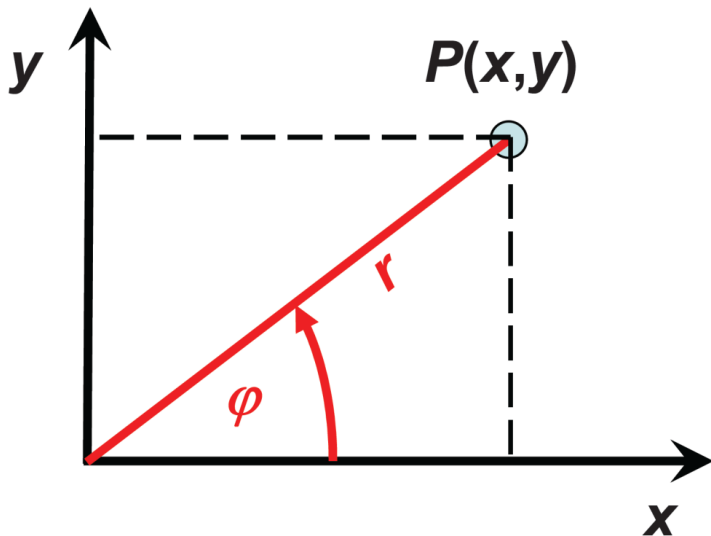


```
rect(30, 20, 55, 55, 3, 6, 12, 18);
```

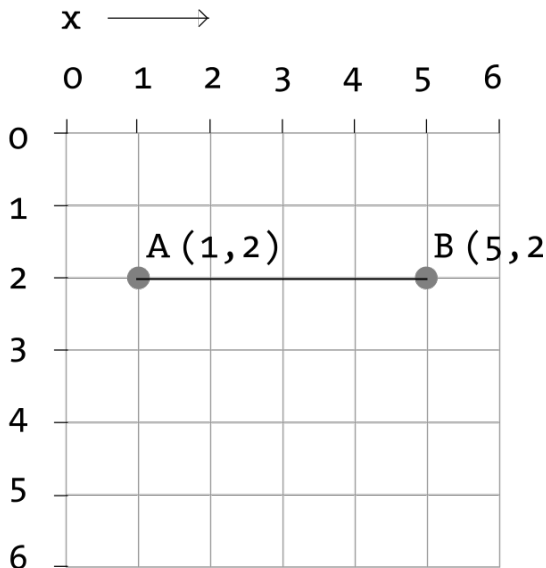
Outline

- 1 Indian Computing Olympiad
- 2 Processing Graphics
 - Getting Started
 - Logic Flow
- 3 Functions
 - Definition
 - Examples
 - Defining Functions
- 4 More on Processing
 - Primitives
 - **Coordinate System**
 - Debugging
 - Interaction
 - Physics
 - Exercise

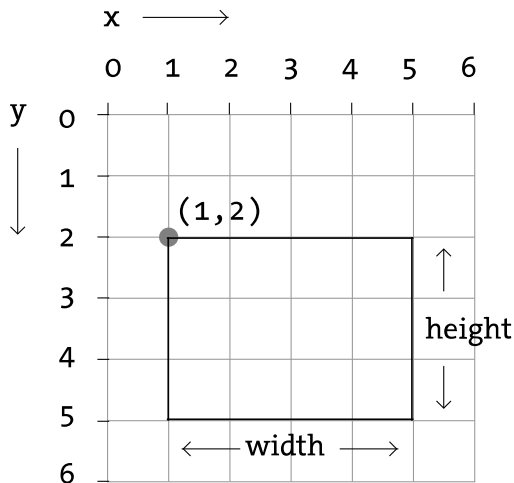
Normal Coordinate System



Processing Coordinate System



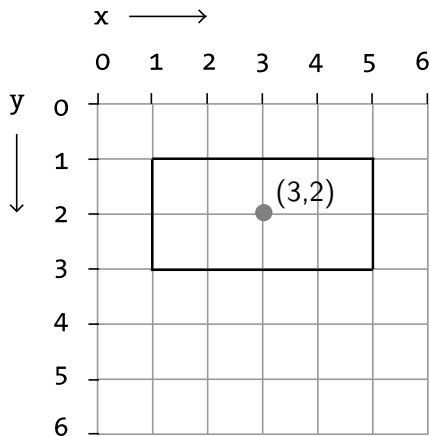
Drawing a Rectangle



`rect(x, y, width, height);`

Example: `rect(1,2,4,3);`

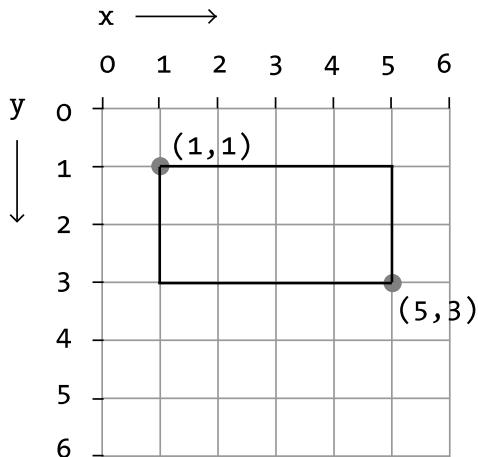
CENTER Rectangle



```
rectMode ( CENTER );  
rect( x , y , width , height );
```

Example: `rectMode (CENTER);`
`rect (3 , 2 , 4 , 2);`

CORNERS Rectangle



```
rectMode ( CORNERS );  
rect(x1 , y1 , x2 , y2 );
```

Example: `rectMode (CORNERS);`
`rect (1, 1, 5, 3) ;`

Outline

- 1 Indian Computing Olympiad
- 2 Processing Graphics
 - Getting Started
 - Logic Flow
- 3 Functions
 - Definition
 - Examples
 - Defining Functions
- 4 More on Processing
 - Primitives
 - Coordinate System
 - **Debugging**
 - Interaction
 - Physics
 - Exercise

- `print()` statement prints all items separated by spaces

```
print(item1, item2, . . . );
```

- `print()` statement prints all items separated by spaces

```
print(item1, item2, . . . );
```

- `println()` is the same, but prints a new line at the end

```
println(item1, item2, . . . );
```

Outline

- 1 Indian Computing Olympiad
- 2 Processing Graphics
 - Getting Started
 - Logic Flow
- 3 Functions
 - Definition
 - Examples
 - Defining Functions
- 4 More on Processing
 - Primitives
 - Coordinate System
 - Debugging
 - **Interaction**
 - Physics
 - Exercise

- Mouse position – Global Variables

`mouseX, mouseY`

`ellipse(mouseX, mouseY, 2*R, 2*R)`

Mouse Handling

- Mouse position – Global Variables

`mouseX, mouseY`

`ellipse(mouseX, mouseY, 2*R, 2*R)`

- Detect Mouse Click

```
if (mousePressed) {  
    fill(255); // White  
}  
else {  
    fill(0); // Black  
}
```

Keyboard Handling

```
char LEFT = 'a', RIGHT = 'd', UP = 'w';  
boolean left, right, up;
```

Keyboard Handling

```
char LEFT = 'a', RIGHT = 'd', UP = 'w';
boolean left, right, up;
void keyPressed() {
    if (key == LEFT)        left = true;
    if (key == RIGHT)       right = true;
    if (key == UP)          up = true;
}
```

Keyboard Handling

```
char LEFT = 'a', RIGHT = 'd', UP = 'w';
boolean left, right, up;

void keyPressed() {
    if (key == LEFT)        left = true;
    if (key == RIGHT)       right = true;
    if (key == UP)          up = true;
}

void keyReleased() {
    if (key == LEFT)        left = false;
    if (key == RIGHT)       right = false;
    if (key == UP)          up = false;
}
```

Keyboard Handling

```
if (left) {  
    // Move Left . . .  
}  
if (right) {  
    // Move Right . . .  
}  
if (up) {  
    // Move Up . . .  
}
```

Outline

- 1 Indian Computing Olympiad
- 2 Processing Graphics
 - Getting Started
 - Logic Flow
- 3 Functions
 - Definition
 - Examples
 - Defining Functions
- 4 More on Processing
 - Primitives
 - Coordinate System
 - Debugging
 - Interaction
 - **Physics**
 - Exercise

Define Position and Velocity

```
float ballX = width/2, ballY = height/2;  
float ballVx = 2, ballVy = 3;
```


Define Position and Velocity

```
float ballX = width/2, ballY = height/2;  
float ballVx = 2, ballVy = 3;  
  
void draw() {  
    ellipse(ballX, ballY, 2*R, 2*R);  
    updateBallPosition();  
}
```

Define Position and Velocity

```
float ballX = width/2, ballY = height/2;
float ballVx = 2, ballVy = 3;

void draw() {
    ellipse(ballX, ballY, 2*R, 2*R);
    updateBallPosition();
}

void updateBallPosition() {
    ballX += ballVx;
    ballY += ballVy;
}
```

Gravity

```
float gravity = 1;
```

Gravity

```
float gravity = 1;

void draw() {
    ellipse(ballX, ballY, 2*R, 2*R);
    updateBallVelocity();
    updateBallPosition();
}
```

Gravity

```
float gravity = 1;

void draw() {
    ellipse(ballX, ballY, 2*R, 2*R);
    updateBallVelocity();
    updateBallPosition();
}

void updateBallVelocity() {
    ballVy += gravity;
}
```

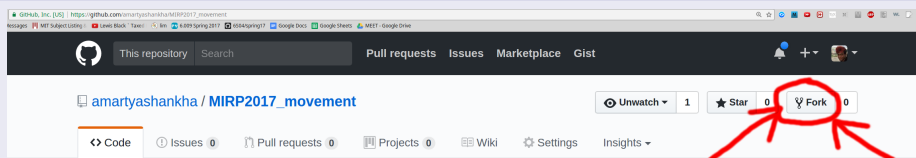
Outline

- 1 Indian Computing Olympiad
- 2 Processing Graphics
 - Getting Started
 - Logic Flow
- 3 Functions
 - Definition
 - Examples
 - Defining Functions
- 4 More on Processing
 - Primitives
 - Coordinate System
 - Debugging
 - Interaction
 - Physics
 - Exercise

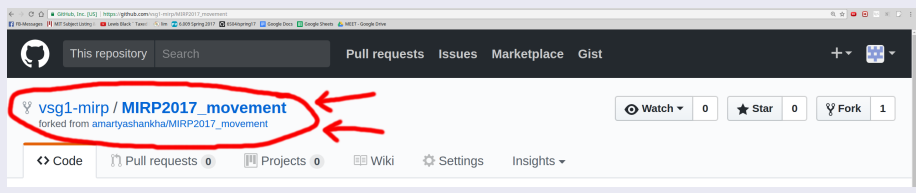
Forking

Go to https://github.com/amartyashankha/MIRP2017_movement

Click on Fork



The title should change



Cloning

Clone the repo

The screenshot shows the GitHub interface for the repository 'vsg1-mirp / MIRP2017_movement', which is forked from 'amartyashankha/MIRP2017_movement'. The repository has 0 Watchers, 0 Stars, and 1 Fork. The main content area shows the 'First Processing lab for MIRP 2017' with an 'Edit' button. Below this, it indicates 1 commit, 1 branch, 0 releases, and 1 contributor. A file list shows 'movement' (first commit) and 'README.md' (first commit). A 'Clone or download' button is visible, and a modal window is open showing the 'Clone with HTTPS' option, providing the URL 'https://github.com/vsg1-mirp/MIRP2017_movement.git'.

- Open Processing
- File → Open
- Navigate to the files inside Movement

Exercise

Resolve collisions with other walls. Move ball using WASD.

Move ball using WASD.