

**MINI PROJECT
(2020-21)**

BIRDY

MID-TERM REPORT



Institute of Engineering & Technology

**Submitted by
Abhishek Singh
(181500028)**

Supervised By: -

Mr. Piyush Vashisth

Technical Trainer

Department of Computer Engineering & Applications

Contents

Abstract	3
1. Introduction	4
1.1 History	4
1.2 General Introduction to the topic	7
1.3 Gameplay	8
1.4 Hardware and Software Requirements	8
2. Objectives	9
3. Implementation Details	10
4. Progress till Date & The Remaining work	11
5. Some Screenshots	12
6. References	

Abstract

In this project I am going to build a 2d animated game called birdy which is a type of flappy bird game developed by Dong Nguyen which is a background scroll based game.

Introduction

1.1 History

The history of game making begins with the development of the first video games, although which video game is the first depends on the definition of *video game*. The first games created had little entertainment value, and their development focus was separate from user experience in fact, these games required mainframe computers to play them. *OXO*, written by Alexander S. Douglas in 1952, was the first computer game to use a digital display. In 1958, a game called *Tennis for Two*, which displayed its output on an oscilloscope, was made by Willy Higinbotham, a physicist working at the Brookhaven National Laboratory. In 1961, a mainframe computer game called *Spacewar!* was developed by a group of Massachusetts Institute of Technology students led by Steve Russell.

True commercial design and development of games began in the 1970s, when arcade video games and first-generation consoles were marketed. In 1971, *Computer Space* was the first commercially sold, coin-operated video game. It used a black-and-white television for its display, and the computer system was made of 74 series TTL chips. In 1972, the first home console system was released called Magnavox Odyssey, developed by Ralph H. Bity. The commercial success of *Pong* led other companies to develop *Pong* clones, spawning the video game industry.

Programmers worked a lot. That same year, Atari released *Pong*, an arcade game that increased video game popularity within the big companies to produce games for these devices. The industry did not see huge innovation in game design and a large number of consoles had very similar games. Many of these early games were often *Pong* clones. Some games were different, however, such as *Gun Fight*, which was significant for several reasons: an early 1975 on-foot, multi-directional shooter, which depicted game characters, game violence, and human-to-human combat. Tomohiro Nishikado's original version was based on discrete logic, which Dave Nutting adapted using the Intel 8080, making it the first video game to use a microprocessor. Console manufacturers soon started to produce consoles that were able to play independently developed games, and ran on microprocessors, marking the beginning of second-generation consoles, beginning with the release of the Fairchild Channel F in 1976.

The flood of *Pong* clones led to the video game crash of 1977, which eventually came to an end with the mainstream success of Taito's 1978 arcade shooter game *Space Invaders*, marking the beginning of the golden age of arcade video games and inspiring dozens of manufacturers to enter the market. Its creator Nishikado not only designed and programmed the game, but also did the artwork, engineered the arcade hardware, and put together a microcomputer from scratch. It was soon ported to the Atari 2600, becoming the first "killer app" and quadrupling the console's sales. At the same time, home computers appeared on the market, allowing individual programmers and hobbyists to develop games. This allowed hardware manufacturer and software manufacturers to act separately. A very large amount of games could be produced by single individuals, as games were easy to make because graphical and memory limitation did not allow for much content. Larger companies developed, who focused selected teams to work on a title. The developers of many early home video games, such as *Zork*, *Baseball*, *Air Warrior*, and *Adventure*, later transitioned their work as products of the early video game industry.

“ I wouldn't recommend [designing computer games] for someone
with a weak heart or a large appetite ”

—Jon Freeman, 1984

The industry expanded significantly at the time, with the arcade video game sector alone (representing the largest share of the gaming industry) generating higher revenues than both pop music and Hollywood films combined. The home video game industry, however, suffered major losses following the North American video game crash of 1983. In 1984 Jon Freeman warned in *Computer Gaming World*:

Q: Are computer games the way to fame and fortune?

A: No. Not unless your idea of fame is having your name recognized by one or two astute individuals at Origins ... I've been making a living (after a fashion) designing games for most of the last six years. I wouldn't recommend it for someone with a weak heart or a large appetite, though.

Chris Crawford and Don Daglow in 1987 similarly advised prospective designers to write games as a hobby first, and to not quit their existing jobs early. The home video game industry was revitalized soon after by the widespread success of the Nintendo Entertainment System.

By 1987 a video game required 12 months to develop and another six to plan marketing. Projects remained usually solo efforts, with single developers delivering finished games to their publishers. With the ever-increasing processing and graphical capabilities of arcade, console and computer products, along with an increase in user expectations, game design moved beyond the scope of a single developer to produce a marketable game in a reasonable time. This sparked the beginning of team-based development. In broad terms, during the 1980s,

pre-production involved sketches and test routines of the only developer. In the 1990s, pre-production consisted mostly of game art previews. In the early 2000s, pre-production usually produced a playable demo.

1.2 General Introduction to the topic

Flappy Bird is a [mobile game](#) developed by Vietnamese [video game artist](#) and [programmer](#) Dong Nguyen (Vietnamese: Nguyễn Hà Đông), under his game development company [dotGears](#).^[1] The game is a [side-scroller](#) where the player controls a bird, attempting to fly between columns of green pipes without hitting them. Nguyen created the game over the period of several days, using a bird protagonist that he had designed for a cancelled game in 2012.

The game was released in May 2013 but received a sudden rise in popularity in early 2014. *Flappy Bird* received poor reviews from some critics, who criticized its high level of difficulty, alleged plagiarism in graphics and game mechanics, while other reviewers found it addictive. At the end of January 2014, it was the most downloaded free game in the [App Store](#) for iOS. During this period, its developer said that *Flappy Bird* was earning \$50,000 a day from in-app advertisements as well as sales.

Flappy Bird was removed from both the [App Store](#) and [Google Play](#) by its creator on February 10, 2014. He claims that he felt guilt over what he considered to be its addictive nature and overuse. The game's popularity and sudden removal caused phones with it pre-installed to be put up for sale for high prices over the Internet.^{[2][3][4]} [Games similar](#) to *Flappy Bird* became popular on the iTunes App Store in the wake of its removal, and both Apple and Google have removed games from their app stores for being too similar to the original.

In August 2014, a revised version of *Flappy Bird*, called *Flappy Birds Family*, was released exclusively for the [Amazon Fire TV](#). Bay Tek Games also released a licensed coin-operated *Flappy Bird* arcade game.^[5]

1.3 Gameplay

Faby after passing the first pair of pipes

Flappy Bird is an arcade-style game in which the player controls the bird Faby, which moves persistently to the right. The player is tasked with navigating Faby through pairs of pipes that have equally sized gaps placed at random heights. Faby automatically descends and only ascends when the player taps the touchscreen. Each successful pass through a pair of pipes awards the player one point. Colliding with a pipe or the ground ends the gameplay. During the game over screen, the player is awarded a bronze medal if they reached ten or more points, a silver medal from twenty points, a gold medal from thirty points, and a platinum medal from forty points

1.4 Hardware Requirements

- Memory [4GB RAM (or higher)]
- Intel core i3 64-bit Processor (or higher)

1.4 Software requirements

- Love 2D
- Sublime text3(or any text editor that support language LUA)

Objective

The purpose of the project is to design and implement a 2-dimensional game written in LUA using gaming library of love 2d. The level will include everything that should be available in an arcade adventure game like the popular Nintendo classic Super Mario game. The game will be a single-player adventure game. The goals of this project is to create an easy to use, pick up and play game that could be played by all ages as long as they have a desktop computer or a laptop pc. The reason was as stated above that they are more gamers playing video games every day meaning a larger potential market.

Implementation Details

Part1: first is to download LOVE 2D application on which our game will be test after implementing the game of code in Sublime text 3 which is essentially most important part.

Part 2: we have to create two file called main and push which will be the most important part of the game.

Part 3: now add background,pipe,bird images in the game and implement the coding of bird(by which the player will move the bird).

Part 4: we create pipepair file so that we can see the pipe moving in pair from which our bird pass to advance.

Part 5: anti gravity update in which we see our bird jump when we press space key

Part 6: collision update in which game will over when bird will strike the ground or pipe.

Part 7: Pause and resume update.

Part 8: Audio update.

Progress

1.) Part 1 is completed

- Love 2d installed
- Sublime text3 installed

2.) Part 2 is completed

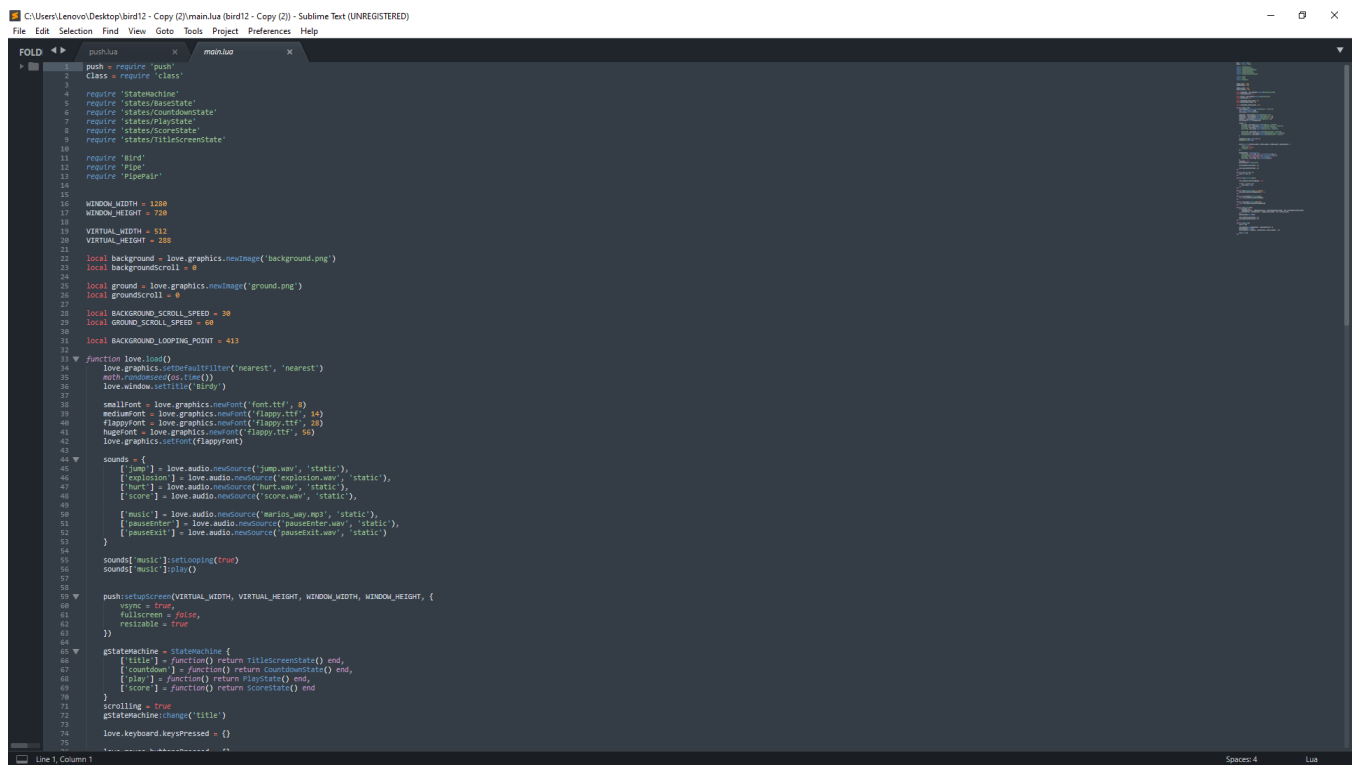
- Create a new project
- Create file called main and push

3.) Part 3 is completed

- Add background image(spites)
- Add Pipe image(spites)
- Add bird Image(spites)
- Implement code for bird to move



SCREENSHOTS



```
1  push = require 'push'
2  class = require 'class'
3
4  require 'statemachine'
5  require 'states/basestate'
6  require 'states/countdownstate'
7  require 'states/playstate'
8  require 'states/scorestate'
9  require 'states/titlescreenstate'
10
11 require 'bird'
12 require 'pipe'
13 require 'pipepair'
14
15 WINDOW_WIDTH = 1280
16 WINDOW_HEIGHT = 720
17 VIRTUAL_WIDTH = 512
18 VIRTUAL_HEIGHT = 288
19
20 local background = love.graphics.newImage('background.png')
21 local backgroundScroll = 0
22 local ground = love.graphics.newImage('ground.png')
23 local groundScroll = 0
24
25 local BACKGROUND_SCROLL_SPEED = 30
26 local GROUND_SCROLL_SPEED = 60
27
28 local BACKGROUND_LOOPING_POINT = 413
29
30 function love.load()
31     love.graphics.setDefaultFilter('nearest', 'nearest')
32     math.randomseed(os.time())
33     love.window.setTitle('Birdy')
34
35     smallFont = love.graphics.newFont('font.ttf', 8)
36     mediumFont = love.graphics.newFont('flappy.ttf', 16)
37     flappyFont = love.graphics.newFont('flappy.ttf', 32)
38     hugeFont = love.graphics.newFont('flappy.ttf', 64)
39     love.graphics.setFont(flappyFont)
40
41     sounds = {
42         ['jump'] = love.audio.newSource('jump.wav', 'static'),
43         ['explosion'] = love.audio.newSource('explosion.wav', 'static'),
44         ['hurt'] = love.audio.newSource('hurt.wav', 'static'),
45         ['score'] = love.audio.newSource('score.wav', 'static'),
46
47         ['music'] = love.audio.newSource('maria_way.mp3', 'static'),
48         ['pauseEnter'] = love.audio.newSource('pauseEnter.wav', 'static'),
49         ['pauseExit'] = love.audio.newSource('pauseExit.wav', 'static')
50     }
51
52     sounds['music']:setLooping(true)
53     sounds['music']:play()
54
55     push:setupScreen(VIRTUAL_WIDTH, VIRTUAL_HEIGHT, WINDOW_WIDTH, WINDOW_HEIGHT, {
56         vsync = true,
57         fullscreen = false,
58         resizable = true
59     })
60
61     gstatemachine = statemachine {
62         ['title'] = function() return titlescreenstate() end,
63         ['countdown'] = function() return countdownstate() end,
64         ['play'] = function() return playstate() end,
65         ['score'] = function() return scorestate() end
66     }
67     scrolling = true
68     gstatemachine:change('title')
69     love.keyboard.keypressed = {}
70
71     -- Love's math library is great! ;)
72 end
```

```
FOLD 1
main.lua  x  main.lua  x
55 sounds["music"] = looping(true)
56 sounds["music"] = play()
57
58
59 push:setupScreen(VIRTUAL_WIDTH, VIRTUAL_HEIGHT, WINDOW_WIDTH, WINDOW_HEIGHT, {
60     vmem = true,
61     fullscreen = false,
62     resizable = true
63 })
64
65 gStateMachine = stateMachine {
66     ['title'] = function() return titleScreenState() end,
67     ['countdown'] = function() return countdownState() end,
68     ['play'] = function() return playState() end,
69     ['score'] = function() return scoreState() end
70 }
71 scrolling = true
72 gStateMachine:change('title')
73 love.keyboard.keypressed = {}
74 love.mouse.buttonspressed = {}
75
76 end
77
78 function love.resize(w, h)
79     push:resize(w, h)
80 end
81
82 function love.keypressed(key)
83     love.keyboard.keypressed[key] = true
84
85     if key == "escape" then
86         love.event.quit()
87     end
88 end
89
90 function love.mousepressed(x, y, button)
91     love.mouse.buttonspressed[button] = true
92 end
93
94 function love.keyboard.waspressed(key)
95     return love.keyboard.keypressed[key]
96 end
97
98 function love.mouse.waspressed(button)
99     return love.mouse.buttonspressed[button]
100 end
101
102 function love.update(dt)
103     if scrolling then
104         backgroundScroll = (backgroundScroll + BACKGROUND_SCROLL_SPEED * dt) % BACKGROUND_LOOPING_POINT
105         groundScroll = (groundScroll + groundScrollSpeed * dt) % VIRTUAL_WIDTH
106     end
107     gStateMachine:update(dt)
108
109     love.keyboard.keypressed = {}
110     love.mouse.buttonspressed = {}
111 end
112
113 function love.draw()
114     push:resize()
115
116     love.graphics.draw(background, backgroundScroll, 0)
117     gStateMachine:render()
118     love.graphics.draw(ground, groundScroll, VIRTUAL_HEIGHT - 16)
119
120     push:finish()
121 end
```

C:\Users\Lenovo\Desktop\bird12 - Copy (2)\push.lua (bird12 - Copy (2)) - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

```
FOLD 1 push.lua x main.lua x
1 local push = {
2
3     defaults = {
4         fullscreen = false,
5         resizable = false,
6         pixelperfect = false,
7         highdpi = true,
8         canvas = true
9     }
10
11 }
12
13 setmetatable(push, push)
14
15 function push.applySettings(settings)
16     for k, v in pairs(settings) do
17         self["." .. k] = v
18     end
19 end
20
21 function push.resetSettings() return self.applySettings(self.defaults) end
22
23 function push.setupScreen(WIDTH, HEIGHT, RWIDTH, RHEIGHT, settings)
24     settings = settings or {}
25
26     self._WIDTH, self._HEIGHT = WIDTH, HEIGHT
27     self._RWIDTH, self._RHEIGHT = RWIDTH, RHEIGHT
28
29     self.applySettings(self.defaults)
30     self.applySettings(settings)
31
32     love.window.setMode(self._RWIDTH, self._RHEIGHT, {
33         fullscreen = self._fullscreen,
34         resizable = self._resizable,
35         highdpi = self._highdpi
36     })
37
38     self:initialize()
39
40     if self._canvas then
41         self:setupCanvas({ name = "default" })
42     end
43
44     self._borderColor = { 0, 0, 0 }
45
46     self._drawFunctions = {
47         ["start"] = self.start,
48         ["end"] = self.finish
49     }
50
51     return self
52 end
53
54 function push.setupCanvas(canvases)
55     table.insert(canvases, { name = "_render" })
56
57     self._canvas = true
58     self._canvases = {}
59
60     for i = 1, #canvases do
61         self._canvases[i] = {
62             name = canvases[i].name,
63             shader = canvases[i].shader,
64             canvas = love.graphics.newCanvas(self._WIDTH, self._HEIGHT)
65         }
66     end
67
68     return self
69 end
70
71 function push.setCanvas(name)
72     if not self._canvas then return true end
73     return love.graphics.setCanvas(self:getCanvasTable(name).canvas)
74 end
75
76 function push:getCanvasTable(name)
77     for i = 1, #self._canvases do
78         if self._canvases[i].name == name then
79             return self._canvases[i]
80         end
81     end
82
83     if not shader then
84         self:_getCanvasTable("_render").shader = name
85     else
86         self:_getCanvasTable(name).shader = shader
87     end
88
89     return self
90 end
91
92 function push:initialize()
93     self._SCALE = self._highdpi and love.window.getDPIScale() or 1
94
95     self._SCALE = {
96         x = self._RWIDTH/self._WIDTH * self._SCALE,
97         y = self._RHEIGHT/self._HEIGHT * self._SCALE
98     }
99
100     if self._stretched then
101         self:_offset = { x = 0, y = 0 }
102     else
103         local scale = math.min(self._SCALE.x, self._SCALE.y)
104         if self._pixelperfect then scale = math.floor(scale) end
105
106         self:_offset = { x = (self._SCALE.x - scale) * (self._WIDTH/2), y = (self._SCALE.y - scale) * (self._HEIGHT/2) }
107         self._SCALE.x, self._SCALE.y = scale, scale --apply same scale to x and y
108     end
109
110     self._WIDTH = self._RWIDTH * self._SCALE - self:_offset.x * 2
111     self._HEIGHT = self._RHEIGHT * self._SCALE - self:_offset.y * 2
112
113     --[[ DEPRECATED ]]--
114     function push:poll(operation, shader)
115         if operation == "start" then
116             self:finish()
117         elseif operation == "finish" or operation == "end" then
118             self:finish(shader)
119         end
120     end
121
122     function push:start()
123         if self._canvas then
124             love.graphics.push()
125             love.graphics.setCanvas(self._canvases[1].canvas)
126         else
127             love.graphics.translate(self:_offset.x, self:_offset.y)
128             love.graphics.setScissor(self:_offset.x, self:_offset.y, self._WIDTH*self._SCALE.x, self._HEIGHT*self._SCALE.y)
129             love.graphics.push()
130             love.graphics.scale(self._SCALE.x, self._SCALE.y)
131         end
132     end
133
134     function push:finish(shader)
135         love.graphics.setBackgroundColor(unpack(self._borderColor))
136         if self._canvas then
```

C:\Users\Lenovo\Desktop\bird12 - Copy (2)\push.lua (bird12 - Copy (2)) - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

```
FOLD 1 push.lua x main.lua x
62 self._canvases[i] = {
63     name = canvases[i].name,
64     shader = canvases[i].shader,
65     canvas = love.graphics.newCanvas(self._WIDTH, self._HEIGHT)
66 }
67 end
68
69 return self
70 end
71
72 function push.setCanvas(name)
73     if not self._canvas then return true end
74     return love.graphics.setCanvas(self:getCanvasTable(name).canvas)
75 end
76
77 function push:getCanvasTable(name)
78     for i = 1, #self._canvases do
79         if self._canvases[i].name == name then
80             return self._canvases[i]
81         end
82     end
83
84     if not shader then
85         self:_getCanvasTable("_render").shader = name
86     else
87         self:_getCanvasTable(name).shader = shader
88     end
89
90     return self
91 end
92
93 function push:initialize()
94     self._SCALE = self._highdpi and love.window.getDPIScale() or 1
95
96     self._SCALE = {
97         x = self._RWIDTH/self._WIDTH * self._SCALE,
98         y = self._RHEIGHT/self._HEIGHT * self._SCALE
99     }
100
101     if self._stretched then
102         self:_offset = { x = 0, y = 0 }
103     else
104         local scale = math.min(self._SCALE.x, self._SCALE.y)
105         if self._pixelperfect then scale = math.floor(scale) end
106
107         self:_offset = { x = (self._SCALE.x - scale) * (self._WIDTH/2), y = (self._SCALE.y - scale) * (self._HEIGHT/2) }
108         self._SCALE.x, self._SCALE.y = scale, scale --apply same scale to x and y
109     end
110
111     self._WIDTH = self._RWIDTH * self._SCALE - self:_offset.x * 2
112     self._HEIGHT = self._RHEIGHT * self._SCALE - self:_offset.y * 2
113
114     --[[ DEPRECATED ]]--
115     function push:poll(operation, shader)
116         if operation == "start" then
117             self:finish()
118         elseif operation == "finish" or operation == "end" then
119             self:finish(shader)
120         end
121     end
122
123     function push:start()
124         if self._canvas then
125             love.graphics.push()
126             love.graphics.setCanvas(self._canvases[1].canvas)
127         else
128             love.graphics.translate(self:_offset.x, self:_offset.y)
129             love.graphics.setScissor(self:_offset.x, self:_offset.y, self._WIDTH*self._SCALE.x, self._HEIGHT*self._SCALE.y)
130             love.graphics.push()
131             love.graphics.scale(self._SCALE.x, self._SCALE.y)
132         end
133     end
134
135     function push:finish(shader)
136         love.graphics.setBackgroundColor(unpack(self._borderColor))
137         if self._canvas then
```

```
C:\Users\Lenovo\Desktop\bird12 - Copy (2)\push.lua (bird12 - Copy (2)) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

FOLD 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 1040 1041 1042 1043 1044 1045 1046 1047 1048 1049 1050 1051 1052 1053 1054 1055 1056 1057 1058 1059 1060 1061 1062 1063 1064 1065 1066 1067 1068 1069 1070 1071 1072 1073 1074 1075 1076 1077 1078 1079 1080 1081 1082 1083 1084 1085 1086 1087 1088 1089 1090 1091 1092 1093 1094 1095 1096 1097 1098 1099 1100 1101 1102 1103 1104 1105 1106 1107 1108 1109 1110 1111 1112 1113 1114 1115 1116 1117 1118 1119 1120 1121 1122 1123 1124 1125 1126 1127 1128 1129 1130 1131 1132 1133 1134 1135 1136 1137 1138 1139 1140 1141 1142 1143 1144 1145 1146 1147 1148 1149 1150 1151 1152 1153 1154 1155 1156 1157 1158 1159 1160 1161 1162 1163 1164 1165 1166 1167 1168 1169 1170 1171 1172 1173 1174 1175 1176 1177 1178 1179 1180 1181 1182 1183 1184 1185 1186 1187 1188 1189 1190 1191 1192 1193 1194 1195 1196 1197 1198 1199 1200 1201 1202 1203 1204 1205 1206 1207 1208 1209 1210 1211 1212 1213 1214 1215 1216 1217 1218 1219 1220 1221 1222 1223 1224 1225 1226 1227 1228 1229 1230 1231 1232 1233 1234 1235 1236 1237 1238 1239 1240 1241 1242 1243 1244 1245 1246 1247 1248 1249 1250 1251 1252 1253 1254 1255 1256 1257 1258 1259 1260 1261 1262 1263 1264 1265 1266 1267 1268 1269 1270 1271 1272 1273 1274 1275 1276 1277 1278 1279 1280 1281 1282 1283 1284 1285 1286 1287 1288 1289 1290 1291 1292 1293 1294 1295 1296 1297 1298 1299 1300 1301 1302 1303 1304 1305 1306 1307 1308 1309 1310 1311 1312 1313 1314 1315 1316 1317 1318 1319 1320 1321 1322 1323 1324 1325 1326 1327 1328 1329 1330 1331 1332 1333 1334 1335 1336 1337 1338 1339 1340 1341 1342 1343 1344 1345 1346 1347 1348 1349 1350 1351 1352 1353 1354 1355 1356 1357 1358 1359 1360 1361 1362 1363 1364 1365 1366 1367 1368 1369 1370 1371 1372 1373 1374 1375 1376 1377 1378 1379 1380 1381 1382 1383 1384 1385 1386 1387 1388 1389 1390 1391 1392 1393 1394 1395 1396 1397 1398 1399 1400 1401 1402 1403 1404 1405 1406 1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422 1423 1424 1425 1426 1427 1428 1429 1430 1431 1432 1433 1434 1435 1436 1437 1438 1439 1440 1441 1442 1443 1444 1445 1446 1447 1448 1449 1450 1451 1452 1453 1454 1455 1456 1457 1458 1459 1460 1461 1462 1463 1464 1465 1466 1467 1468 1469 1470 1471 1472 1473 1474 1475 1476 1477 1478 1479 1480 1481 1482 1483 1484 1485 1486 1487 1488 1489 1490 1491 1492 1493 1494 1495 1496 1497 1498 1499 1500 1501 1502 1503 1504 1505 1506 1507 1508 1509 1510 1511 1512 1513 1514 1515 1516 1517 1518 1519 1520 1521 1522 1523 1524 1525 1526 1527 1528 1529 1530 1531 1532 1533 1534 1535 1536 1537 1538 1539 1540 1541 1542 1543 1544 1545 1546 1547 1548 1549 1550 1551 1552 1553 1554 1555 1556 1557 1558 1559 1560 1561 1562 1563 1564 1565 1566 1567 1568 1569 1570 1571 1572 1573 1574 1575 1576 1577 1578 1579 1580 1581 1582 1583 1584 1585 1586 1587 1588 1589 1590 1591 1592 1593 1594 1595 1596 1597 1598 1599 1600 1601 1602 1603 1604 1605 1606 1607 1608 1609 1610 1611 1612 1613 1614 1615 1616 1617 1618 1619 1620 1621 1622 1623 1624 1625 1626 1627 1628 1629 1630 1631 1632 1633 1634 1635 1636 1637 1638 1639 1640 1641 1642 1643 1644 1645 1646 1647 1648 1649 1650 1651 1652 1653 1654 1655 1656 1657 1658 1659 1660 1661 1662 1663 1664 1665 1666 1667 1668 1669 1670 1671 1672 1673 1674 1675 1676 1677 1678 1679 1680 1681 1682 1683 1684 1685 1686 1687 1688 1689 1690 1691 1692 1693 1694 1695 1696 1697 1698 1699 1700 1701 1702 1703 1704 1705 1706 1707 1708 1709 1710 1711 1712 1713 1714 1715 1716 1717 1718 1719 1720 1721 1722 1723 1724 1725 1726 1727 1728 1729 1730 1731 1732 1733 1734 1735 1736 1737 1738 1739 1740 1741 1742 1743 1744 1745 1746 1747 1748 1749 1750 1751 1752 1753 1754 1755 1756 1757 1758 1759 1760 1761 1762 1763 1764 1765 1766 1767 1768 1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1786 1787 1788 1789 1790 1791 1792 1793 1794 1795 1796 1797 1798 1799 1800 1801 1802 1803 1804 1805 1806 1807 1808 1809 1810 1811 1812 1813 1814 1815 1816 1817 1818 1819 1820 1821 1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840 1841 1842 1843 1844 1845 1846 1847 1848 1849 1850 1851 1852 1853 1854 1855 1856 1857 1858 1859 1860 1861 1862 1863 1864 1865 1866 1867 1868 1869 1870 1871 1872 1873 1874 1875 1876 1877 1878 1879 1880 1881 1882 1883 1884 1885 1886 1887 1888 1889 1890 1891 1892 1893 1894 1895 1896 1897 1898 1899 1900 1901 1902 1903 1904 1905 1906 1907 1908 1909 1910 1911 1912 1913 1914 1915 1916 1917 1918 1919 1920 1921 1922 1923 1924 1925 1926 1927 1928 1929 1930 1931 1932 1933 1934 1935 1936 1937 1938 1939 1940 1941 1942 1943 1944 1945 1946 1947 1948 1949 1950 1951 1952 1953 1954 1955 1956 1957 1958 1959 1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 1970 1971 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024 2025 2026 2027 2028 2029 2030 2031 2032 2033 2034 2035 2036 2037 2038 2039 2040 2041 2042 2043 2044 2045 2046 2047 2048 2049 2050 2051 2052 2053 2054 2055 2056 2057 2058 2059 2060 2061 2062 2063 2064 2065 2066 2067 2068 2069 2070 2071 2072 2073 2074 2075 2076 2077 2078 2079 2080 2081 2082 2083 2084 2085 2086 2087 2088 2089 2090 2091 2092 2093 2094 2095 2096 2097 2098 2099 2100 2101 2102 2103 2104 2105 2106 2107 2108 2109 2110 2111 2112 2113 2114 2115 2116 2117 2118 2119 2120 2121 2122 2123 2124 2125 2126 2127 2128 2129 2130 2131 2132 2133 2134 2135 2136 2137 2138 2139 2140 2141 2142 2143 2144 2145 2146 2147 2148 2149 2150 2151 2152 2153 2154 2155 2156 2157 2158 2159 2160 2161 2162 2163 2164 2165 2166 2167 2168 2169 2170 2171 2172 2173 2174 2175 2176 2177 2178 2179 2180 2181 2182 2183 2184 2185 2186 2187 2188 2189 2190 2191 2192 2193 2194 2195 2196 2197 2198 2199 2200 2201 2202 2203 2204 2205 2206 2207 2208 2209 2210 2211 2212 2213 2214 2215 2216 2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231 2232 2233 2234 2235 2236 2237 2238 2239 2240 2241 2242 2243 2244 2245 2246 2247 2248 2249 2250 2251 2252 2253 2254 2255 2256 2257 2258 2259 2260 2261 2262 2263 2264 2265 2266 2267 2268 2269 2270 2271 2272 2273 2274 2275 2276 2277 2278 2279 2280 2281 2282 2283 2284 2285 2286 2287 2288 2289 2290 2291 2292 2293 2294 2295 2296 2297 2298 2299 2300 2301 2302 2303 2304 2305 2306 2307 2308 2309 2310 2311 2312 2313 2314 2315 2316 2317 2318 2319 2320 2321 2322 2323 2324 2325 2326 2327 2328 2329 2330 2331 2332 2333 2334 2335 2336 2337 2338 2339 2340 2341 2342 2343 2344 2345 2346 2347 2348 2349 2350 2351 2352 2353 2354 2355 2356 2357 2358 2359 2360 2361 2362 2363 2364 2365 2366 2367 2368 2369 2370 2371 2372 2373 2374 2375 2376 2377 2378 2379 2380 2381 2382 2383 2384 2385 2386 2387 2388 2389 2390 2391 2392 2393 2394 2395 2396 2397 2398 2399 2400 2401 2402 2403 2404 2405 2406 2407 2408 2409 2410 2411 2412 2413 2414 2415 2416 2417 2418 2419 2420 2421 2422 2423 2424 2425 2426 2427 2428 2429 2430 2431 2432 2433 2434 2435 2436 2437 2438 2439 2440 2441 2442 2443 2444 2445 2446 2447 2448 2449 2450 2451 2452 2453 2454 2455 2456 2457 2458 2459 2460 2461 2462 2463 2464 2465 2466 2467 2468 2469 2470 2471 2472 2473 2474 2475 2476 2477 2478 2479 2480 2481 2482 2483 2484 2485 2486 2487 2488 2489 2490 2491 2492 2493 2494 2495 2496 2497 2498 2499 2500 2501 2502 2503 2504 2505 2506 2507 2508 2509 2510 2511 2512 2513 2514 2515 2516 2517 2518 2519 2520 2521 2522 2523 2524 2525 2526 2527 2528 2529 2530 2531 2532 2533 2534 2535 2536 2537 2538 2539 2540 2541 2542 2543 2544 2545 2546 2547 2548 2549 2550 2551 2552 2553 2554 2555 2556 2557 2558 2559 2560 2561 2562 2563 2564 2565 2566 2567 2568 2569 2570 2571 2572 2573 2574 2575 2576 2577 2578 2579 2580 2581 2582 2583 2584 2585 2586 2587 2588 2589 2590 2591 2592 2593 2594 2595 2596 2597 2598 2599 2600 2601 2602 2603 2604 2605 2606 2607 2608 2609 2610 2611 2612 2613 2614 2615 2616 2617 2618 2619 2620 2621 2622 2623 2624 2625 2626 2627 2628 2629 2630 2631 2632 2633 2634 2635 2636 2637 2638 2639 2640 2641 2642 2643 2644 2645 2646 2647 2648 2649 2650 2651 2652 2653 2654 2655 2656 2657 2658 2659 2660 2661 2662 2663 2664 2665 2666 2667 2668 2669 2670 2671 2672 2673 2674 2675 2676 2677 2678 2679 2680 2681 2682 2
```

```
C:\Users\Lenovo\Desktop\bird12 - Copy (2)\class.lua (bird12 - Copy (2)) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

FOLD 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62
push.lua x class.lua x
1 local function include_helper(to, from, seen)
2   if from == nil then
3     return to
4   elseif type(from) == 'table' then
5     return from
6   elseif seen[from] then
7     return seen[from]
8   end
9
10  seen[from] = to
11  for k,v in pairs(from) do
12    k = include_helper(k, v, seen)
13    if to[k] == nil then
14      to[k] = include_helper({}, v, seen)
15    end
16  end
17  return to
18 end
19
20 local function include(class, other)
21   return include_helper(class, other, {})
22 end
23
24
25 local function clone(other)
26   return setmetatable(include({}, other), getmetatable(other))
27 end
28
29 local function new(class)
30   -- main
31   class = class or {}
32   local inc = class.__includes or {}
33   if getmetatable(inc) then inc = {inc} end
34   for _, other in ipairs(inc) do
35     if type(other) == 'string' then
36       other = _G[other]
37     end
38     include(class, other)
39   end
40
41   class.__index = class
42   class.__init = class.__init or class.__init or function() end
43   class.__include = class.__include or include
44   class.__clone = class.__clone or clone
45
46   return setmetatable(class, {__call = function(c, ...)
47     local o = setmetatable({}, c)
48     o:__init(...)
49     return o
50   end})
51
52
53
54
55
56 if class_common == false and not common then
57   common = {}
58   function common.class(name, prototype, parent)
59     return new({__includes = {prototype, parent}})
60   end
61   function common.instance(class, ...)
62     return class(...)
63   end
64 end
65
66 return setmetatable({new = new, include = include, clone = clone},
67   {__call = function(...) return new(...) end})
68 end
```

```
C:\Users\Lenovo\Desktop\bird12 - Copy (2)\Bird.lua (bird12 - Copy (2)) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

FOLD 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
push.lua x Bird.lua x
1 bird = class{}
2
3 local GRAVITY = 25
4
5 function bird:Init()
6   self.image = love.graphics.newImage(bird.png)
7   self.x = VIRTUAL_WIDTH / 2 - 8
8   self.y = VIRTUAL_HEIGHT / 2 - 8
9
10  self.width = self.image:getWidth()
11  self.height = self.image:getHeight()
12
13  self.dy = 0
14 end
15
16 function bird:collides(pipe)
17   if (self.x + 2 < (self.width - 4) <= pipe.x and self.x + 2 <= pipe.x + PIPE_WIDTH then
18     if (self.y + 2) < (self.height - 4) <= pipe.y and self.y + 2 <= pipe.y + PIPE_HEIGHT then
19       return true
20     end
21   end
22   return false
23 end
24
25 function bird:update(dt)
26   self.dy = self.dy + GRAVITY * dt
27
28   if love.keyboard.wasPressed('space') or love.mouse.wasPressed(1) then
29     self.dy = -8
30     sounds.jump:play()
31   end
32   self.y = self.y + self.dy
33 end
34
35 function bird:render()
36   love.graphics.draw(self.image, self.x, self.y)
37 end
```


References

<https://www.lua.org/>

<https://www.tutorialspoint.com/lua/index.htm>

<https://love2d.org/>

<https://www.sublimetext.com/3>

<https://www.wikipedia.org/>