Національний технічний університет України
«Київський політехнічний інститут»

Факультет Прикладної Математики
Кафедра Системного Програмування і Спеціалізованих Комп’ютерних Систем

ПРАКТИЧНА РОБОТА №2
з дисципліни
«Комп’ютерна Графіка»

ТЕМА: «Побудова кривої Безьє»

Група: КВ-11

Виконав: Брюханов О.

Оцінка: ___

Київ – 2023

Код програми

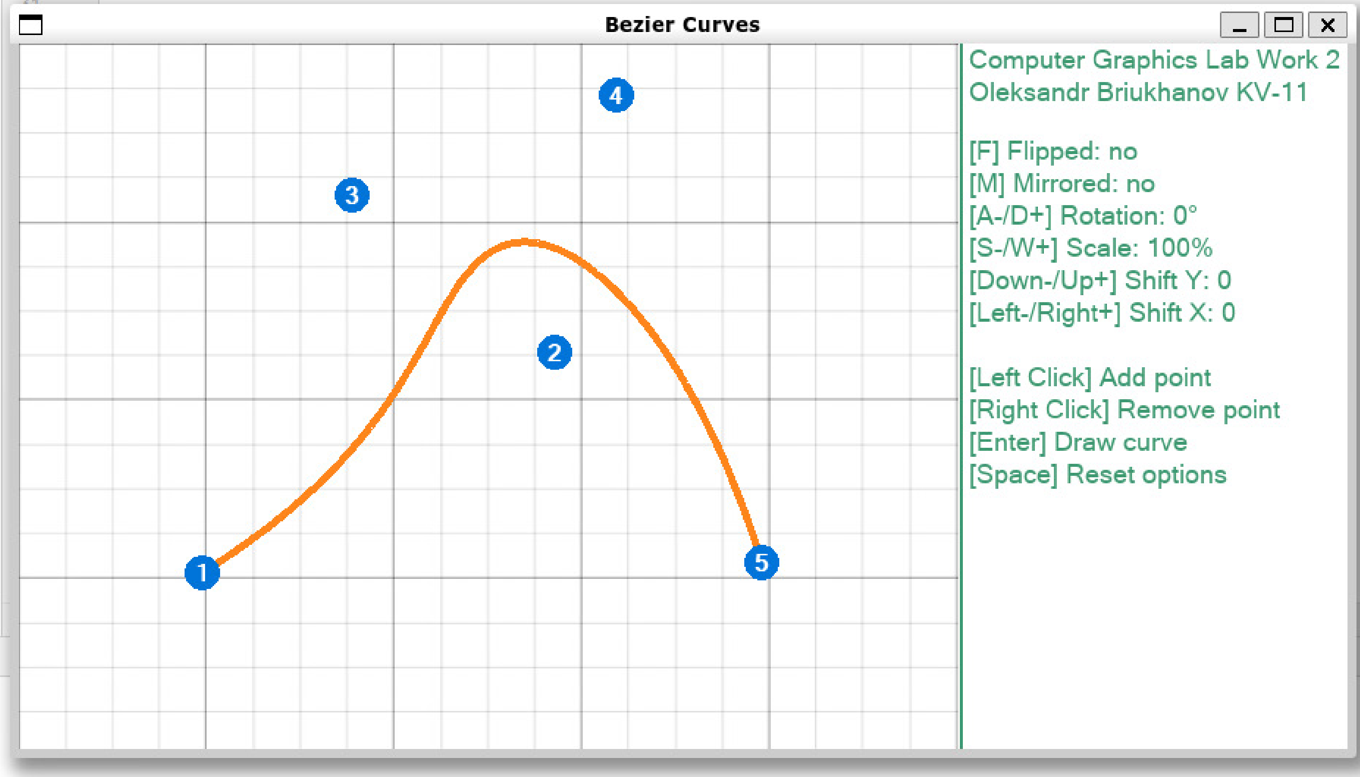
```ruby
CurveOptions            = Struct.new(:w, :h, :precision, :scale, :shift, :rotation,
:mirror, :flip)
def default_curve_opts = CurveOptions.new(640, 480, 100, 1, [0, 0], 0, false, false)

CurvePoint = Struct.new(:x, :y)

class BezierCurve
  # @param [Array<CurvePoint>] control_points
  def initialize(control_points)
    @control_points = control_points
  end

  # @param [CurveOptions] options
  # @return [Array<CurvePoint>]
  def get_points(options)
    points = Array.new
    dt = 1.0 / options.precision
    t = 0.0
    while true do
      point = CurvePoint.new(0, 0)

      # Calculate point
      @control_points.each_with_index do |cp, i|
        basis = beizer_basis(i, @control_points.length - 1, t)
        point.x += cp.x * basis
        point.y += cp.y * basis
      end

      # Apply transformations
      point.y = options.h - point.y if options.flip
      point.x = options.w - point.x if options.mirror

      if options.scale != 1
        point.x *= options.scale
        point.y *= options.scale
      end

      if options.shift != [0, 0]
        point.x += options.shift[0]
        point.y += options.shift[1]
      end

      if options.rotation != 0
        angle = options.rotation * (Math::PI / 180)
        point.x = point.x * Math::cos(angle) - point.y * Math::sin(angle)
        point.y = point.x * Math::sin(angle) + point.y * Math::cos(angle)
      end

      point.x = [options.w, ([0, point.x].max)].min
      point.y = [options.h, ([0, point.y].max)].min

      # Push point to array
      points.push(point)

      if t == 1.0
        break
      else
        t = [1.0, (t + dt)].min
      end
    end

    return points
  end

  protected

  def beizer_basis(i, n, t)
    comb = (1..n).to_a.combination(i).size
    comb * (t ** (n - i)) * (1 - t) ** i
  end
end


require 'ruby2d'
require_relative 'curve'

set width: 905, height: 480, background: 'white', title: 'Bezier Curves'

$texts  = []
$opts   = default_curve_opts
$lines  = []
$points = []
def draw_curve
  if $points.length < 4
    return
  end

  curve = BezierCurve.new($points.map{|p| CurvePoint.new(p[0][0], p[0][1])})
  data  = curve.get_points($opts)

  (1 ... data.length).each do |i|
    a = data[i - 1]
    b = data[i]

    $lines.push(Line.new(
      x1: a.x, y1: a.y,
      x2: b.x, y2: b.y,
      width: 5,
      color: 'orange',
      z: 10
    ))
  end
end

def erase_curve
  until $lines.length == 0
```

```
    line = $lines.pop
    line.remove
  end
end

def yes_no(x) = x ? 'yes' : 'no'

def describe_opts
  $texts[0].text = "[F] Flipped: " + yes_no($opts.flip)
  $texts[1].text = "[M] Mirrored: " + yes_no($opts.mirror)
  $texts[2].text = "[A-/D+] Rotation: " + $opts.rotation.to_s + "°"
  $texts[3].text = "[S-/W+] Scale: " + ($opts.scale * 100).to_int.to_s + "%"
  $texts[4].text = "[Down-/Up+] Shift Y: " + ($opts.shift[1] * -1).to_s
  $texts[5].text = "[Left-/Right+] Shift X: " + $opts.shift[0].to_s
end

def redraw_curve
  describe_opts
  erase_curve
  draw_curve
end

def add_point(x, y)
  circle = Circle.new(
    x: x, y: y,
    radius: 12,
    color: 'blue',
    z: 20
  )

  text = Text.new(
    ($points.length + 1).to_s,
    x: x - 5, y: y - 9,
    font: 'microsoftsansserif.ttf',
    size: 16,
    style: 'bold',
    color: 'white',
    z: 21
  )

  $points.push([[x, y], circle, text])
end

def remove_point
  if $points.length == 0
    return
  end

  last_point = $points.pop
  last_point[1].remove
  last_point[2].remove
end

def draw_text(text, x, y)
```

```ruby
    Text.new(
      text,
      x: x, y: y,
      font: 'microsoftsansserif.ttf',
      size: 18,
      color: 'olive',
      )
  end

  on :key_down do |event|
    case event.key
    when 'w'
      $opts.scale += 0.1
      redraw_curve
    when 's'
      $opts.scale -= 0.1
      redraw_curve
    when 'a'
      $opts.rotation -= 10
      if $opts.rotation < 0
        $opts.rotation += 360
      end
      redraw_curve
    when 'd'
      $opts.rotation += 10
      if $opts.rotation > 360
        $opts.rotation -= 360
      end
      redraw_curve
    when 'f'
      $opts.flip = !$opts.flip
      redraw_curve
    when 'm'
      $opts.mirror = !$opts.mirror
      redraw_curve
    when 'up'
      $opts.shift[1] -= 10
      redraw_curve
    when 'down'
      $opts.shift[1] += 10
      redraw_curve
    when 'left'
      $opts.shift[0] -= 10
      redraw_curve
    when 'right'
      $opts.shift[0] += 10
      redraw_curve
    when 'space'
      $opts = default_curve_opts
      redraw_curve
    when 'return'
      redraw_curve
    else
      puts event.key
    end
  end

  on :mouse_down do |event|
    if event.button == :left
      if event.x <= 640
        add_point(event.x, event.y)
      end
    else
      remove_point
    end
  end

  Image.new('squares.jpeg')

  Line.new(
    x1: 642, y1: 0,
    x2: 642, y2: 480,
    width: 2,
    color: 'olive'
  )

  draw_text("Computer Graphics Lab Work 2", 647, 0)
  draw_text("Oleksandr Briukhanov KV-11", 647, 22)
  draw_text("[Left Click] Add point", 647, 216)
  draw_text("[Right Click] Remove point", 647, 238)
  draw_text("[Enter] Draw curve", 647, 260)
  draw_text("[Space] Reset options", 647, 282)
  $texts = [
    draw_text("", 647, 62),
    draw_text("", 647, 84),
    draw_text("", 647, 106),
    draw_text("", 647, 128),
    draw_text("", 647, 150),
    draw_text("", 647, 172)
  ]

  describe_opts
  show
```

Результати роботи

```
end
```

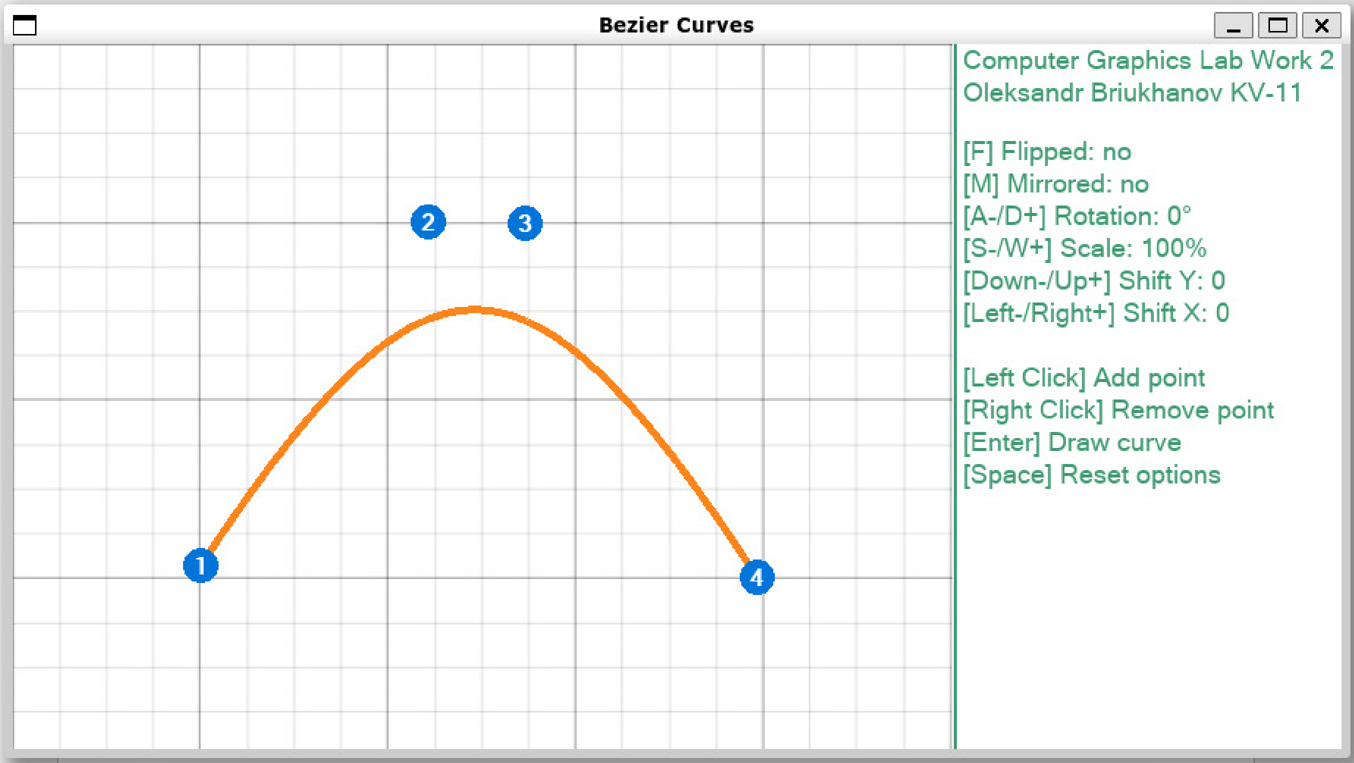


Bezier Curves

Computer Graphics Lab Work 2
Oleksandr Briukhanov KV-11

[F] Flipped: no
[M] Mirrored: no
[A-/D+] Rotation: 0°
[S-/W+] Scale: 100%
[Down-/Up+] Shift Y: 0
[Left-/Right+] Shift X: 0

[Left Click] Add point
[Right Click] Remove point
[Enter] Draw curve
[Space] Reset options

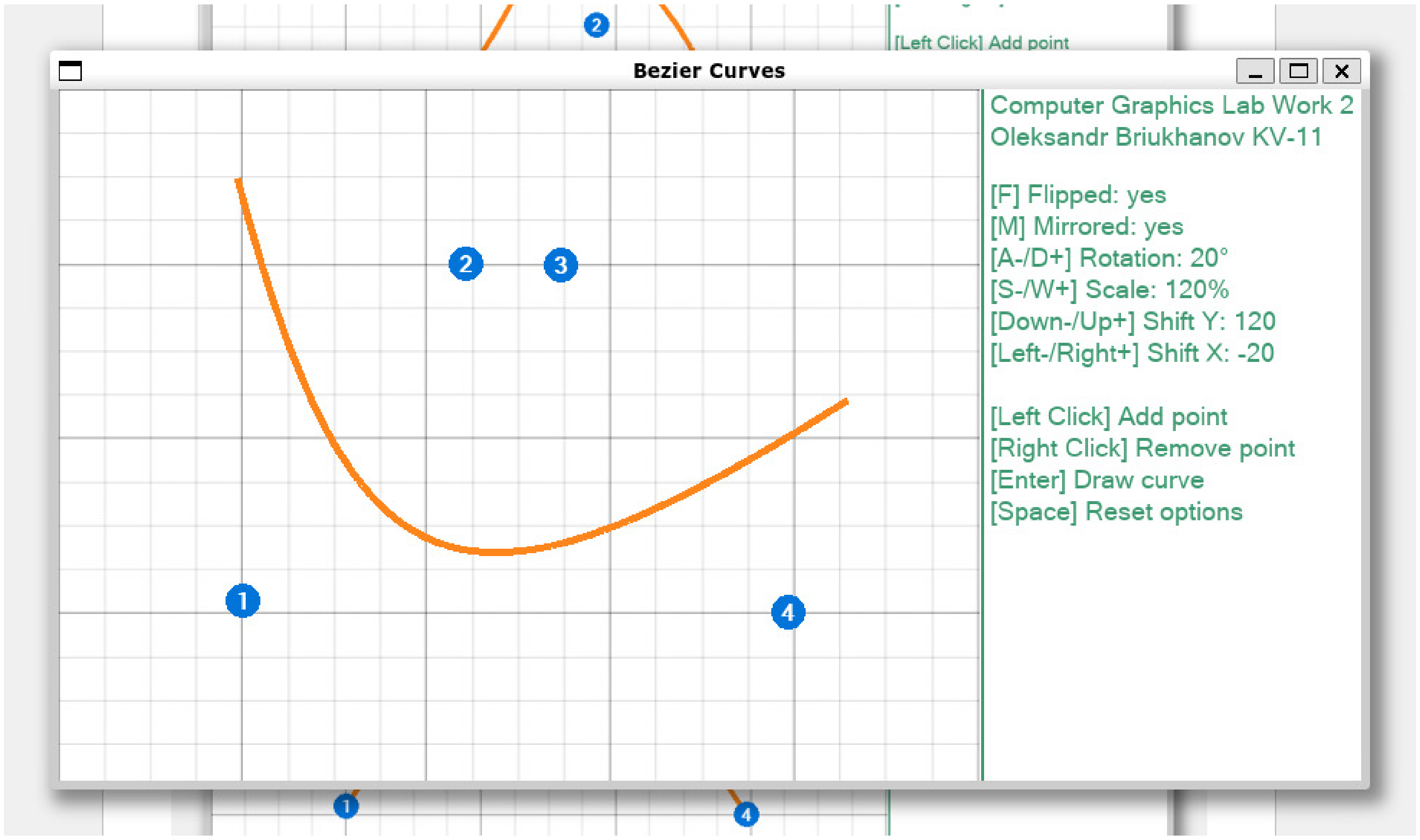


Bezier Curves

Computer Graphics Lab Work 2
Oleksandr Briukhanov KV-11

[F] Flipped: no
[M] Mirrored: no
[A-/D+] Rotation: 0°
[S-/W+] Scale: 100%
[Down-/Up+] Shift Y: 0
[Left-/Right+] Shift X: 0

[Left Click] Add point
[Right Click] Remove point
[Enter] Draw curve
[Space] Reset options



Computer Graphics Lab Work 2
Oleksandr Briukhanov KV-11

[F] Flipped: yes
[M] Mirrored: yes
[A-/D+] Rotation: 20°
[S-/W+] Scale: 120%
[Down-/Up+] Shift Y: 120
[Left-/Right+] Shift X: -20

[Left Click] Add point
[Right Click] Remove point
[Enter] Draw curve
[Space] Reset options