Computer Graphics Project

Submitted To: Dr. Aloke Datta

Submitted By:

Shivesh Kaundinya	19ucs154
Aryan Dhuria	19ucs150
Shikhar Nigam	19ucs146

OBJECTIVE

The aim of this project is to create a digital cartoon character using the knowledge of computer graphics.

INTRODUCTION

In this project, we are going to create an image of Homura Akemi, a character from the popular anime Magic Girl Madoka. For this project, we will be using python and its Turtle module.

The Turtle module in Python provides a simple interface for drawing shapes on the screen. It has a robotic turtle which starts at (0, 0) on the screen. This turtle can be manipulated using trivial commands such as forward, backward, rotate, goto and many more. Hence, it can be used to create complex shapes and drawings quite easily.

Code: https://github.com/Shivesh-K/cg/blob/master/anime.py

SETUP

Imports

First up, we will be importing the necessary modules.

```
import turtle as te
from typing import Tuple
```

Global Declarations

Now, we need to declare a few global variables that we will require as well as configure the turtle drawing environment.

```
WriteSteps = 500        # Sampling times of Bezier function
Speed = 1
Width = 600             # Interface width
Height = 500            # Interface height
Xh, Yh = 0, 0           # Record the handle of the previous Bessel function
```

- WriteSteps is the number of time steps for Bezier curve
- Speed is the speed of turtle
- Width and Height are the dimensions of the turtle window
- Xh and Yh are used for smooth Bezier curve

```
te.tracer(10)
te.setup(Width, Height, 0, 0)
te.setworldcoordinates(0, Height, Width, 0)
te.pensize(1)
te.speed(Speed)
te.penup()
```

- te.tracer is used to set the turtle to draw on every 10th refresh of screen
- te.setup set the turtle window
- te.setworldcoordinates sets the actual drawing area. It is set such that top-left is the origin and right & down are positive x & y axes, respectively.
- te.pensize sets the thickness of stroke

DRAWING PRIMITIVES

For drawing our actual figure we first need a few primitive methods to draw portions of the image.

move_to

This function simply moves the turtle from current position to the given position without drawing the path.

```
def move_to(point):
    te.penup()
    te.goto(point)
```

bezier_point

This function takes a list of control points and the time step and returns the point on the curve at that time step. It uses recursion to continuously reduce the list of points until only one point remains.

```
def bezier_point(points, timestep, dimension=2):
    degree = len(points) - 1
    if(degree == 0):
        return points[0]
    else:
        new_points = []
        for idx in range(degree):
            temp = []
            for d in range(dimension):
                temp.append(points[idx][d] * (1.0 - timestep) +
                            points[idx + 1][d] * timestep)
            new_points.append(tuple(temp))
        return bezier_point(tuple(new_points), timestep, dimension)
```

bezier_curve

This function gets all the control points for the curve and then draws the curve. It iterates over the time steps and draws the point for each one.

```
def bezier_curve(points, dimension=None):
    if(dimension is None):
        dimension = len(points[0])

    move_to(points[0])
    te.pendown()
    for timestep in range(0, WriteSteps + 1):
        p = bezier_point(points, timestep / WriteSteps, dimension)
        te.goto(p)
    te.penup()
```

bezier_curve_through

The bezier_curve_through method draws a bezier curve through the given points along with the current position as the first point.

It also takes an optional parameter **relative** (default: false). If it is true, the points are taken as relative to the current position and, therefore, are first converted to absolute points by adding the current position to them. Otherwise, they remain the same.

Then, it inserts the current position to the list of points and calls the method for drawing the Bezier curve.

```
def bezier_curve_through(points, relative: bool = False):
    global Xh
    global Yh
    curr = te.position()
    points = list(points)
    if(relative):
        for i in range(len(points)):
            points[i] = tuple([c1 + c2 for c1, c2 in zip(points[i], curr)])
    points.insert(0, curr)
    bezier_curve(tuple(points))
    Xh = points[-1][0] - points[-2][0]
    Yh = points[-1][1] - points[-2][1]
```

smooth_bezier_curve

This function creates a bezier curve between the given points while smoothly connecting it to the last curve drawn. Similar to the previous method, it also takes an optional parameter **relative** which performs the same action as seen before.

```
def smooth_bezier_curve(points, relative: bool = False):
    global Xh
    global Yh
    points = list(points)
    points.insert(0, (Xh + te.position()[0], Yh + te.position()[1]))
    if(relative):
        points[0] = (points[0][0] - te.position()[0],
                     points[0][1] - te.position()[1])
    bezier_curve_through(tuple(points), relative)
```

line_between

Given two points; source and destination, draw a line connecting source to destination.

```
def line_between(source, destination):
    move_to(source)
    te.pendown()
    te.goto(destination)
```

line_displace

Given certain displacement, draw a line from the current position to the displaced position.

```
def line_displace(displacement):
    line_between(te.position(), te.position() + displacement)
```

line_to

Given the destination point, draw a line from the current position to that destination.

```
def line_to(destination):
    line_between(te.position(), destination)
```

horizontal_to

Takes a destination x coordinate and draws a horizontal line from current position to destination point.

```
def horizontal_to(destination_x):
    line_between(te.position(), (destination_x, te.ycor()))
```

horizontal_displace

Takes a displacement in the x direction and draws the line.

```
def horizontal_displace(dx):
    line_between(te.position(), te.position() + (dx, 0))
```

vertical_displace

Takes a displacement in the y direction and draws the line.

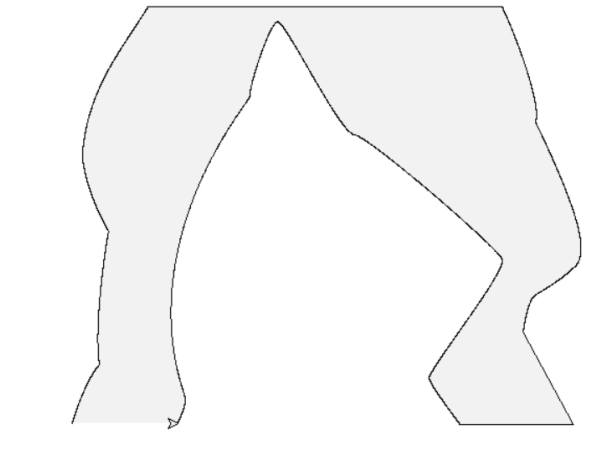
```
def vertical_displace(dy):
    line_between(te.position(), te.position() + (0, dy))
```

polyline

This method takes some points p_1, p_2, p_3,..., p_n as input and draws the lines p_1 to p_2, p_2 to p_3,..., p_{n-1} to p_n.

IMPLEMENTATION

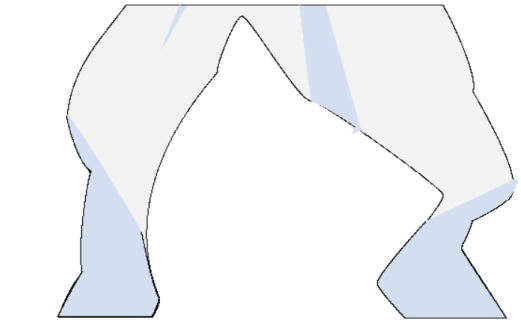
Now we can use the built up functions to draw the final image.



Below code is used to draw above curve through listed

1. Bezier_curve_through
2. Smooth_bezier_curve
3. Move_to
4. Line_displace
5. line_to

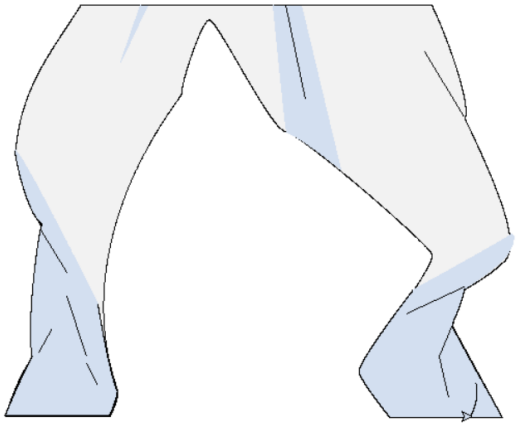
```
# Coat
te.color("black", "#F2F2F2")
move_to((61, 462))
te.begin_fill()
smooth_bezier_curve(((12, -41), (27, -58)), relative=True)
bezier_curve_through(((-6, -36), (6, -118), (9, -132)), relative=True)
bezier_curve_through(((-15, -27), (-23, -51), (-26, -74)), relative=True)
bezier_curve_through(((4, -66), (38, -105), (65, -149)), relative=True)
horizontal_to(486)
bezier_curve_through(((12, 24), (40, 99), (33, 114)), relative=True)
bezier_curve_through(((39, 82), (55, 129), (39, 144)), relative=True)
smooth_bezier_curve(((-31, 23), (-39, 28)), relative=True)
smooth_bezier_curve(((-12, 37), (-12, 37)), relative=True)
line_displace((50, 92))
horizontal_to(445)
smooth_bezier_curve(((-29, -38), (-31, -46)), relative=True)
smooth_bezier_curve(((78, -107), (72, -119)), relative=True)
smooth_bezier_curve(((355, 178), (340, 176)))
smooth_bezier_curve(((272, 63), (264, 64)))
smooth_bezier_curve(((-29, 67), (-27, 73)), relative=True)
smooth_bezier_curve(((99, 292), (174, 428), (173, 439)))
smooth_bezier_curve(((-8, 23), (-8, 23)), relative=True)
line_to((61, 462))
```



Below code is used to draw above curve through listed functions

1. Bezier_curve_through
2. Smooth_bezier_curve
3. Pencolour
4. Polyline
5. Pencolor
6. Begin_fill
7. end_fill

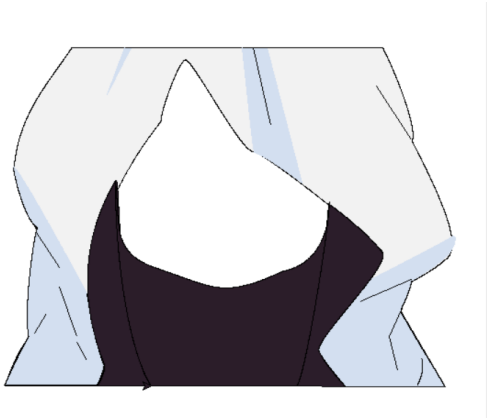
```
    move_to((60.5, 461.5))
    te.color("black", "#D3DFF0")
    te.begin_fill()
    bezier_curve_through(((0, 0), (17, -42), (27, -59)), relative=True)
    bezier_curve_through(((-6, -33), (6, -128), (10, -133)), relative=True)
    bezier_curve_through(
        ((-15, -10), (-27, -66), (-27.285, -75)), relative=True)
    te.pencolor("#D3DFF0")
    bezier_curve_through(((12.285, 11), (82.963, 156),
                          (82.963, 156)), relative=True)
    te.pencolor("black")
    smooth_bezier_curve(((12.322, 75), (19.322, 86)), relative=True)
    bezier_curve_through(((-1, 11), (-8, 25), (-8, 25)), relative=True)
    horizontal_to(60.5)
    te.end_fill()

    move_to((444.5, 464))
    te.begin_fill()
    bezier_curve_through(((0, 0), (-29, -36), (-31, -46)), relative=True)
    smooth_bezier_curve(((53.59, -82.337), (53.59, -82.337)), relative=True)
    te.pencolor("#D3DFF0")
    smooth_bezier_curve(((86.41, -47.663), (96.072, -54.85)), relative=True)
    smooth_bezier_curve(((563.5, 297.5), (570.5, 299.5), (518.5, 334)))
    te.pencolor("black")
    bezier_curve_through(((-2, 16), (-12, 33), (-12, 37)), relative=True)
    smooth_bezier_curve(((50, 92), (50, 93)), relative=True)
    horizontal_to(444.5)
    te.end_fill()

    move_to((195, 49))
    te.begin_fill()
    te.pencolor("#D3DFF0")
    polyline(((195, 49), (175.5, 106.5), (202.522, 49)))
    te.pencolor("black")
    horizontal_to(195)
    te.pencolor("#D3DFF0")
```



Below code is used to draw above curve through listed functions

1. Line_between
2. Polyline
3. Move_to
4. bezier_curve_through

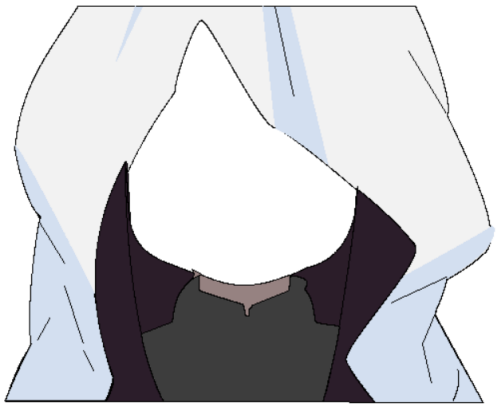
```
# Wrinkles
te.pencolor("black")
line_between((94.5, 397.5), (107.5, 373.5))
line_between((122.5, 317.5), (95.875, 274.699))
line_between((122.5, 341.5), (141.5, 402.5))
line_between((141.5, 409.5), (153.5, 431.5))
line_between((340.023, 49), (360.5, 144))
line_between((478.5, 95.5), (518.5, 161.5))
line_between((518.5, 332.5), (460.5, 359.5))
polyline(((506.5, 369.5), (493.5, 402.5), (502.5, 443.5)))
move_to((530, 429))
bezier_curve_through(((4, 16), (-5, 33), (-5, 33)), relative=True)
```



Below code is used to draw above curve through listed functions

1. Smooth_bezier_curve
2. Bezier_curve_through
3. Begin_fill
4. Line_to
5. Horizontal_to
6. move_to

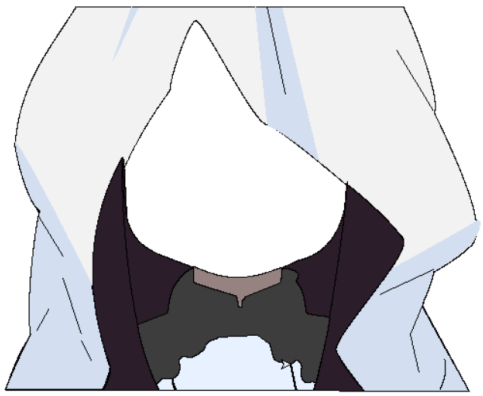
```
te.color("black", "#2b1d2a")
move_to((225, 462))
te.begin_fill()
horizontal_to(165)
smooth_bezier_curve(((9, -15), (8, -25)), relative=True)
bezier_curve_through(((-47, -126), (6, -212), (12, -225)), relative=True)
smooth_bezier_curve(((185, 305), (202, 428), (225, 462)))
line_to((225, 462))
te.end_fill()

move_to((390, 462))
te.begin_fill()
bezier_curve_through(
    ((10, -23), (34, -180), (35, -222)), relative=True)
bezier_curve_through(((7, 4), (54, 45), (61, 61)), relative=True)
smooth_bezier_curve(((-73, 101), (-72, 118)), relative=True)
bezier_curve_through(((5, 15), (31, 46), (31, 45)), relative=True)
line_to((390, 462))
te.end_fill()

"""
Layer 3
"""

# Inside of jacket
te.color("black", "#2b1d29")
move_to((225, 462))
te.begin_fill()
bezier_curve_through(((-28, -50), (-40, -166), (-40, -250)), relative=True)
bezier_curve_through(((6, 51), (-6, 87), (45, 106)), relative=True)
smooth_bezier_curve(((64, 27), (89, 24)), relative=True)
smooth_bezier_curve(((49, -18), (56, -20)), relative=True)
smooth_bezier_curve(((50, -10), (51, -85)), relative=True)
bezier_curve_through(((0, 29), (-25, 201), (-36, 225)), relative=True)
line_to((225, 462))
te.end_fill()
```



Below code is used to draw above curve through listed functions

1. Bezier_curve_through
2. Smooth_bezier_curve
3. Vertical_displace
4. Line_displace
5. Line_to
6. end_fill

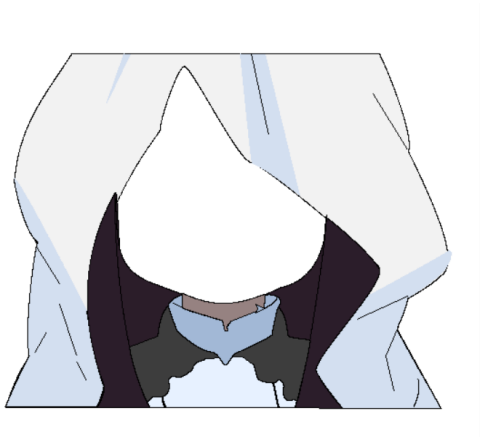
```
# Clothes
te.color("black", "#3D3D3D")
move_to((225, 462))
te.begin_fill()
bezier_curve_through(((-5, -5), (-22, -53), (-23, -70)), relative=True)
line_displace((32, -13))
bezier_curve_through(((3, -25), (6, -28), (12, -36)), relative=True)
smooth_bezier_curve(((13, -12), (16, -12)), relative=True)
vertical_displace(-2)
bezier_curve_through(((45, 20), (64, 14), (94, 1)), relative=True)
vertical_displace(2)
bezier_curve_through(((8, -2), (15, 2), (17, 4)), relative=True)
smooth_bezier_curve(((0, 6), (-2, 9)), relative=True)
bezier_curve_through(((10, 10), (10, 29), (11, 33)), relative=True)
smooth_bezier_curve(((23, 4), (25, 6)), relative=True)
smooth_bezier_curve(((-17, 83), (-17, 78)), relative=True)
line_to((225, 462))
te.end_fill()
```



Below code is used to draw above curve through listed functions

1. Vertical_displace
2. Bezier_curve_through
3. Smooth_bezier_curve
4. Horizontal_displace
5. Vertical_displace
6. Line_displace
7. begin_fill

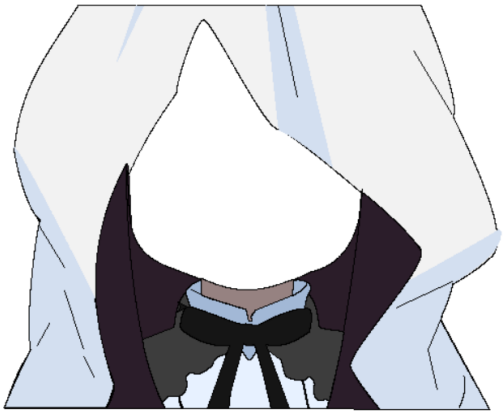
```
te.color("black", "#968281")
move_to((262, 329))
te.begin_fill()
vertical_displace(17)
bezier_curve_through(((1, 2), (44, 14), (45, 15)), relative=True)
smooth_bezier_curve(((3, 12), (3, 12)), relative=True)
horizontal_displace(3)
vertical_displace(-5)
bezier_curve_through(((1, -3), (4, -6), (5, -7)), relative=True)
line_displace((36, -14))
bezier_curve_through(((1, -1), (3, -16), (2, -17)), relative=True)
smooth_bezier_curve(((318, 348), (296, 344), (262, 329)))
te.end_fill()
```



Below code is used to draw above curve through listed functions

1. Bezier_curve_through
2. Smooth_bezier_curve
3. Line_displace
4. End_fill
5. Line_to
6. move_to

```
te.color("black", "#E7F1FF")
move_to((225, 462))
te.begin_fill()
line_displace((-3, - 5))
bezier_curve_through(((0, -2), (4, -4), (5, -6)), relative=True)
smooth_bezier_curve(((16, 3), (19, -8)), relative=True)
smooth_bezier_curve(((0, -7), (0, -11)), relative=True)
smooth_bezier_curve(((5, -8), (9, -5)), relative=True)
smooth_bezier_curve(((19, -8), (19, -11)), relative=True)
smooth_bezier_curve(((6, -7), (6, -7)), relative=True)
smooth_bezier_curve(((7, -2), (9, -4)), relative=True)
line_displace((41, -2))
line_displace((12, 9))
smooth_bezier_curve(((3, 15), (7, 18)), relative=True)
smooth_bezier_curve(((15, 4), (17, 4)), relative=True)
smooth_bezier_curve(((4, -4), (6, -4)), relative=True)
smooth_bezier_curve(((6, 4), (5, 9)), relative=True)
smooth_bezier_curve(((0, 9), (0, 9)), relative=True)
smooth_bezier_curve(((1, 7), (7, 6)), relative=True)
smooth_bezier_curve(((8, 0), (8, 0)), relative=True)
line_displace((-2, 8))
line_to((225, 462))
te.end_fill()

te.pensize(2)
move_to((240, 450))
smooth_bezier_curve(((0, 9), (3, 12)), relative=True)
move_to((372, 462))
bezier_curve_through(((-2, -4), (-5, -29), (-7, -28)), relative=True)
te.pensize(1)
```



Below code is used to draw above curve through listed functions

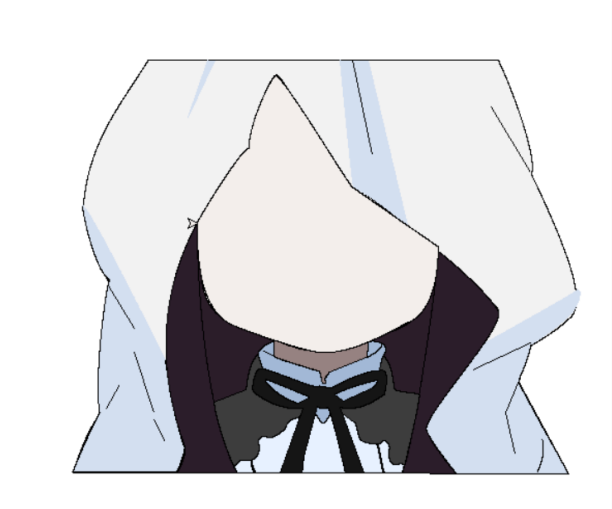
1. Smooth_bezier_curve
2. Line_displace
3. Bezier_curve_through
4. Begin_fill
5. Move_to
6. End_fill
7. horizontal_displace

```
te.color("black", "#A2B8D6")
move_to((262, 331))
te.begin_fill()
bezier_curve_through(((0, 8), (-1, 13), (0, 15)), relative=True)
smooth_bezier_curve(((43, 14), (45, 15)), relative=True)
line_displace((3, 12))
horizontal_displace(3)
smooth_bezier_curve(((-1, -3), (0, -5)), relative=True)
line_displace((5, -7))
line_displace((36, -14))
bezier_curve_through(((1, -1), (2, -12), (2, -15)), relative=True)
smooth_bezier_curve(((25, -2), (15, 13)), relative=True)
bezier_curve_through(((-2, 4), (-7, 29), (-7, 32)), relative=True)
smooth_bezier_curve(((-35, 19), (-41, 22)), relative=True)
smooth_bezier_curve(((-9, 14), (-12, 14)), relative=True)
smooth_bezier_curve(((-7, -12), (-14, -15)), relative=True)
bezier_curve_through(((-19, -2), (-41, -25), (-41, -25)), relative=True)
smooth_bezier_curve(((-10, -26), (-10, -30)), relative=True)
smooth_bezier_curve(((255, 332), (262, 331)))
te.end_fill()

move_to((262, 346))
line_displace((-12, -6))
move_to((369, 333))
bezier_curve_through(((2, 4), (-6, 10), (-15, 14)), relative=True)
```

Below code is used to draw above curve through listed functions

1. Bezier_curve_through
2. Line_displace
3. Horizontal_displace
4. Smooth_bezier_through
5. begin_fill

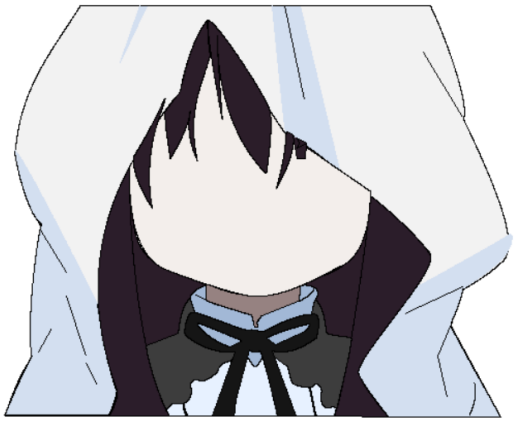
```
te.color("black", "#151515")
move_to((247, 358))
te.begin_fill()
bezier_curve_through(((-5, 3), (-8, 20), (-6, 23)), relative=True)
bezier_curve_through(((25, 21), (50, 17), (50, 17)), relative=True)
line_displace((-23, 64))
horizontal_displace(22)
smooth_bezier_curve(((1, -13), (2, -16)), relative=True)
line_displace((13, -50))
bezier_curve_through(((2, 2), (7, 3), (10, 1)), relative=True)
smooth_bezier_curve(((18, 65), (18, 65)), relative=True)
horizontal_displace(19)
line_displace((-24, -65))
bezier_curve_through(((21, 5), (39, -10), (44, -13)), relative=True)
bezier_curve_through(((5, -20), (1, -21), (0, -24)), relative=True)
bezier_curve_through(((-18, -2), (-49, 15), (-52, 17)), relative=True)
smooth_bezier_curve(((-11, -3), (-15, -1)), relative=True)
smooth_bezier_curve(((252, 356), (247, 358)))
te.end_fill()
```



Below code is used to draw above curve through listed functions

1. Bezier_curve_through
2. Smooth_bezier_curve
3. End_fill
4. Move_to
5. Begin_fill
6. Line_displace

```
te.color("black", "#A2B8D6")
move_to((297, 387))
te.begin_fill()
line_displace((-11, 6))
bezier_curve_through(((-1, 0), (-20, -7), (-30, -19)), relative=True)
smooth_bezier_curve(((259, 373), (297, 385), (297, 387)))
te.end_fill()

move_to((323, 384))
te.begin_fill()
line_displace((8, 7))
line_displace((30, -14))
bezier_curve_through(((1, -1), (5, -6), (4, -7)), relative=True)
smooth_bezier_curve(((329, 379), (323, 384)))
te.end_fill()
```



Below code is used to draw above curve through listed functions

1. Bezier_curve_through
2. Smooth_bezier_curve
3. End_fill
4. Begin_fill
5. move_to

```
te.color("black", "#F3EEEB")
move_to((185, 212))
te.begin_fill()
bezier_curve_through(((4, -9), (46, -77), (52, -75)), relative=True)
bezier_curve_through(((-2, -17), (19, -68), (27, -73)), relative=True)
bezier_curve_through(((16, 15), (71, 108), (76, 112)), relative=True)
smooth_bezier_curve(((76, 53), (86, 60)), relative=True)
bezier_curve_through(((0, 65), (-27, 75), (-31, 76)), relative=True)
bezier_curve_through(((-50, 28), (-70, 30), (-85, 30)), relative=True)
smooth_bezier_curve(((-77, -22), (-86, -26)), relative=True)
smooth_bezier_curve(((180, 302), (186, 228), (185, 212)))
te.end_fill()
```

Below code is used to draw above curve through listed functions

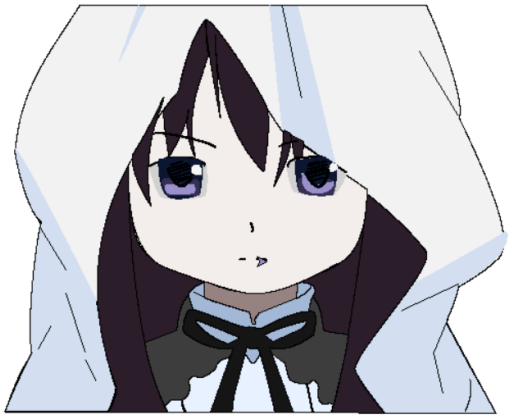
1. Bezier_curve_through
2. Smooth_bezier_curve
3. Line_to
4. Begin_fill
5. End_fill
6. move_to

```
te.color("black", "#2B1D29")
move_to((189, 202))
te.begin_fill()
bezier_curve_through(((-1, 22), (19, 51), (19, 51)), relative=True)
smooth_bezier_curve(((-10, -42), (7, -92)), relative=True)
smooth_bezier_curve(((212, 168), (196, 189), (189, 202)))
te.end_fill()

move_to((221, 155))
te.begin_fill()
bezier_curve_through(((-2, 6), (5, 48), (5, 48)), relative=True)
smooth_bezier_curve(((18, -28), (20, -48)), relative=True)
bezier_curve_through(((-5, 24), (4, 43), (7, 50)), relative=True)
bezier_curve_through(((-10, -49), (3, -72), (13, -106)), relative=True)
bezier_curve_through(((-2, -7), (-3, -32), (-3, -35)), relative=True)
bezier_curve_through(((-17, 18), (-27, 71), (-27, 71)), relative=True)
line_to((221, 155))
te.end_fill()

move_to((264, 64))
te.begin_fill()
bezier_curve_through(((-4, 5), (14, 100), (14, 100)), relative=True)
smooth_bezier_curve(((-6, -79), (-5, -85)), relative=True)
bezier_curve_through(((0, 98), (49, 139), (49, 139)), relative=True)
smooth_bezier_curve(((8, -50), (3, -65)), relative=True)
smooth_bezier_curve(((272, 64), (264, 64)))
te.end_fill()

move_to((342, 176))
te.begin_fill()
bezier_curve_through(((-1, 27), (-10, 57), (-10, 57)), relative=True)
smooth_bezier_curve(((20, -33), (17, -54)), relative=True)
line_to((342, 176))
te.end_fill()
```

Below code is used to draw above curve through listed functions

1. Line_displace
2. Bezier_curve_through
3. Smooth_bezier_curve
4. End_fill
5. Move_to
6. Pencolor
7. Pensize

```
te.color("black", "#D1D1D1")
te.pensize(2)
move_to((206, 212))
te.begin_fill()
line_displace((15, -7))
bezier_curve_through(((4, -1), (26, -2), (30, 0)), relative=True)
smooth_bezier_curve(((10, 3), (12, 7)), relative=True)
te.pencolor("#D1D1D1")
te.pensize(1)
smooth_bezier_curve(((2, 27), (-1, 30)), relative=True)
smooth_bezier_curve(((-39, 5), (-44, 1)), relative=True)
smooth_bezier_curve(((206, 212), (206, 212)))
te.end_fill()

move_to((384, 204))
te.begin_fill()
te.pencolor("black")
te.pensize(2)
bezier_curve_through(((-3, -1), (-18, -1), (-28, 1)), relative=True)
smooth_bezier_curve(((-9, 6), (-10, 9)), relative=True)
te.pencolor("#D1D1D1")
te.pensize(1)
smooth_bezier_curve(((3, 18), (6, 23)), relative=True)
smooth_bezier_curve(((38, 6), (40, 4)), relative=True)
smooth_bezier_curve(((10, -9), (13, -22)), relative=True)
te.pencolor("black")
te.pensize(2)
line_to((384, 204))
te.end_fill()
```

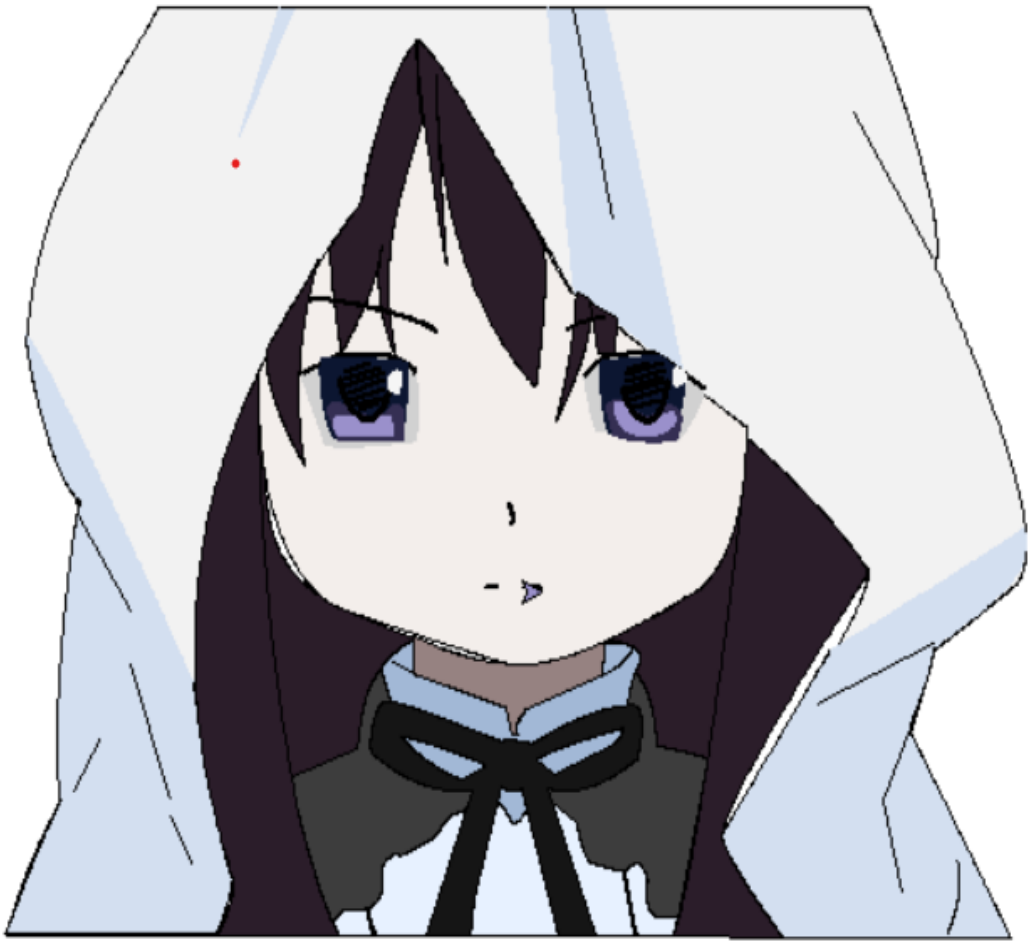


Below code is used to draw above curve through listed functions

1. Move_to
2. Line_between
3. Bezier_curve_through
4. Smooth_bezier_curve
5. pencolor

```
te.pencolor("black")
move_to((309, 270))
bezier_curve_through(((0, 0), (4, 7), (1, 9)), relative=True)
line_between((296.5, 307.5), (303.5, 307.5))
move_to((315, 307))
smooth_bezier_curve(((10, -1), (10, 2)), relative=True)
```

FINAL RESULT



CONCLUSION

Through this project, we learned about various computer graphics concepts and how to implement them. We learned about Bezier curves & line drawing algorithms as well as about drawing in Python using the Turtle module.