Gliwice, 15.01.2023r.

Fundamentals and implementations of modeling and simulations.

Documentation of laboratory task no 7.

Title: Bezier curves

Author: Artur Łukaszek

Field of studies: Informatics (sem.V)

Project Objective:

Goal of the task is to create some nice, fancy, funny, interesting shapes with the aid of Bezier curves.

Description:

A **Bézier curve** is a parametric curve used in computer graphics and related fields. A set of discrete "control points" defines a smooth, continuous curve by means of a formula. Usually the curve is intended to approximate a real-world shape that otherwise has no mathematical representation or whose representation is unknown or too complicated. The Bézier curve is named after French engineer Pierre Bézier (1910–1999), who used it in the 1960s for designing curves for the bodywork of Renault cars. Other uses include the design of computer fonts and animation. Bézier curves can be combined to form a Bézier spline, or generalized to higher dimensions to form Bézier surfaces.

In vector graphics, Bézier curves are used to model smooth curves that can be scaled indefinitely. "Paths", as they are commonly referred to in image manipulation programs, are combinations of linked Bézier curves. Paths are not bound by the limits of rasterized images and are intuitive to modify.

Bézier curves are also used in the time domain, particularly in animation, user interface design and smoothing cursor trajectory in eye gaze controlled interfaces. For example, a Bézier curve can be used to specify the velocity over time of an object such as an icon moving from A to B, rather than simply moving at a fixed number of pixels per step. When animators or interface designers talk about the "physics" or "feel" of an operation, they may be referring to the particular Bézier curve used to control the velocity over time of the move in question.

A Bezier curve is a parametric curve defined on plane by $n+1$ control points $\mathbf{P}_i = (x_i, y_i), i = 0, \ldots, n$. The point $\mathbf{C}(t)$ on curve is determined by the formula

$$\mathbf{C}(t) = \sum_{i=0}^{n} \mathbf{P}_i \cdot B_{i,n}(t),$$

where $t \in [0, 1]$ is a parameter and $B_{i,n}(t)$ is a Bernstein polynomial:

$$B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}, \quad i = 0, ..., n.$$

Operators helpful in creating the various shapes with the aid of Bezier curves:

– rotation about angle ϕ (multiplication by the rotation matrix $\begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}$ transforms point (x, y) into a new point (x', y'), rotated about angle ϕ);

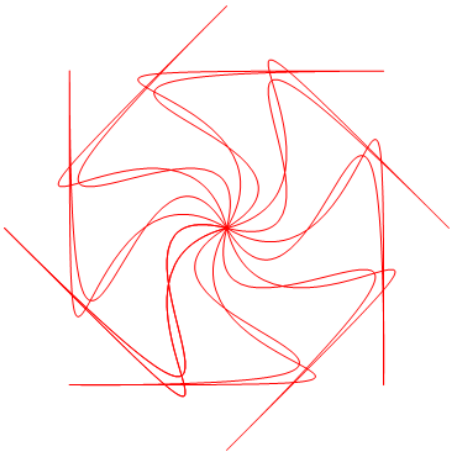
– reflection about X axis (multiplication by the reflection matrix $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ transforms point (x, y) into a new point $(x, -y)$);

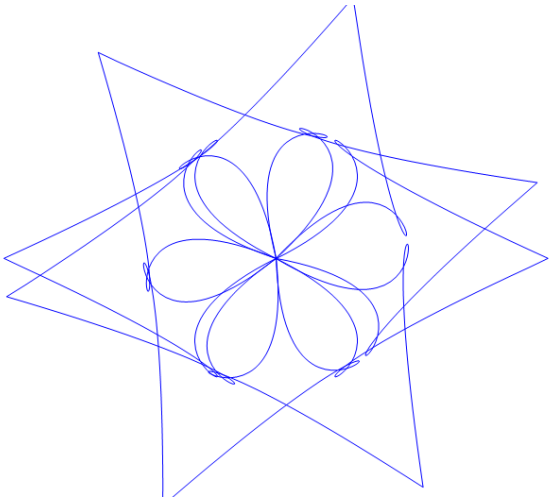
– reflection about Y axis (multiplication by the reflection matrix $\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$ transforms point (x, y) into a new point $(-x, y)$);

– translation about vector $(2, 3)$ (addition $(x+2, y+3)$ transforms point (x, y) into a new translated point (x', y')).

```
rotation[points_, angle_] := Module[{p = points, a = angle}, Return[p.{{Cos[a], -Sin[a]}, {Sin[a], Cos[a]}}]]
                             moduł                           zwróć  cosinus  sinus  sinus  cosinus

reflection[points_, axis_] := Module[{p = points, ax = axis},
                              moduł

  If[ax == "X", Return[p.{{1, 0}, {0, -1}}], If[ax == "Y", Return[p.{{-1, 0}, {0, 1}}], Return[p]]]
  operator warun…  zwróć                     operator warun…  zwróć                     zwróć

translation[points_, vector_] := Module[{p = points, v = vector, l = Length[points]},
                                 moduł                                 długość

  For[i = 1, i ≤ l, i++, p[[i, 1]] = p[[i, 1]] + v[[1]]; p[[i, 2]] = p[[i, 2]] + v[[2]]];
  dla

  Return[p]]
  zwróć
```

Shapes created using bezier curves:





Enclosures:

File with the program(Łukaszek_Artur_proj_7.nb)