Name: **SID:**

Lab 7 - Part 1: Bézier curves

In this lab, we will render an approximation of a parametric curve known as the Bézier.

Consider the parametric equation of a segment between two **control points** P0 and P1:

(1) B(t) = (1 - t) * P0 + t * P1

For n control points, we can recursively apply Eq. 1 to consecutive control points until we are left with only B(t). For three control points:

(2) B(t) = **(1 - t)** * [**(1 - t)** * P0 + **t** * P1]
+ **t** * [**(1 - t)** * P1 + **t** * P2]

1. Given n control points, what is the degree of the polynomial equation for the Bezier curve?

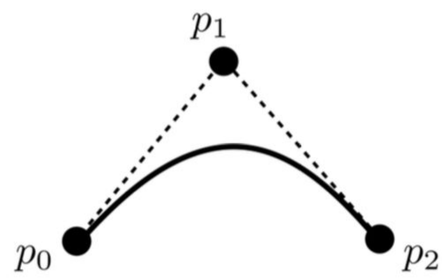
In general, B(t) for n points is given by:

$$B(t) = \sum_{i=0}^{n-1} \binom{n-1}{i} t^i (1-t)^{n-1-i} P_i$$

2. Since we may need the **factorial**, **combination** and **binomial** terms of B(t) in this lab, complete the code to for these functions below.

```
float factorial(int n) {




}
float combination(int n, int k) {


}

float binomial(int n, int k, float t) {


}
```

3. Ok, let's practice a problem.



Quadratic Bezier curve. This problem will guide you through deriving the quadratic Bezier blending functions.

Given three control points p_0, p_1, and p_2, a quadratic Bezier curve

$$f(t) = a_0 + a_1 t + a_2 t^2 \tag{1}$$

can be determined from the following conditions:

condition 1 $f(0) = p_0$

condition 2 $f(1) = p_2$

condition 3 $f'(0) = 2(p_1 - p_0)$

1. Fill in the right hand side of the equation below by differentiating equation (1).

$$f'(t) = \tag{2}$$

2. Use conditions 1-3 and equations (1) and (2) to fill in the following linear system:

$$\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} & & \\ & & \\ & & \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \\ p_2 \end{pmatrix}$$

3. Given that

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ a & b & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -a & -b & 1 \end{pmatrix}$$

fill in the following linear system:

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} & & \\ & & \\ & & \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \\ p_2 \end{pmatrix}$$

4. Use the above work to write down the quadratic Bezier blending functions $b_o(t)$, $b_1(t)$, $b_2(t)$, such that

$$f(t) = b_o(t)p_0 + b_1(t)p_1 + b_2(t)p_2.$$

(hint: recall that $f(t) = \mathbf{t}^T \mathbf{a}$, where $\mathbf{t} = (1, t, t^2)^T$ and $\mathbf{a} = (a_0, a_1, a_2)^T$.)

Extra space:

Lab 8 - Part 1: Coding

Download the skeleton code from iLearn and modify **main.cpp** as follows:

- ❏ **Define a global vector** to store the control points.
- ❏ **Push back the mouse click coordinates** into the vector in the function GL_mouse.
- ❏ Write the code for the **factorial, combination and binomial.**
- ❏ **Draw line segments between points along the Bezier curve** in GL_render().
    - ❏ You can use GL_LINE_STRIP to draw line segments between consecutive points.
    - ❏ You can iterate t between 0 and 1 in steps of 0.01.

Optional: Rather than using the general equation for the Bézier curve to write your program, can write a program where you recursively apply Eq. 1 to consecutive points until get B(t)?