Math 155B - Spring 2020 - "Mini"-Midterm 3 - May 14, 2020 - 11:00am to 4:00pm sharp

Duration: 45 minutes

Instructions: Read completely before starting! You have 45 minutes once you start the midterm.

- Hand in your answers to the **four questions** as the answers to problems 1-4 on Gradescope.
- Hand in this cover page (or other statement of Academic Integrity along with the start and stop time) as the answer to problem 5 on Gradescope.
- You may (1) print out the quiz and write answers on the printed sheet, or (2) use a tablet to write on the PDF file, or (3) write answers on a blank sheet of paper (preferably white, unlined printer paper).
- You must SHOW WORK. Do not just upload unsupported answers.
- **BEFORE YOU START WORKING OR THINKING ABOUT THE PROBLEMS:** Write the start time in the space below.
- **WHEN YOU STOP:** Write the stop time in the space below. The total time should be at most 45 minutes. If more than 50 minutes, explain in the comments below.
- **AFTER YOU STOP:** Sign the Academic Integrity Acknowledgement below.
- Convert your written answers to a PDF file.
- **UPLOAD TO GRADESCOPE – IMMEDIATELY AFTER THE STOP TIME:** If there are problems uploading, please explain in the comment section. If you modify any answers after the "STOP TIME", that is, while preparing to upload, please explain in the comments below.

Academic Integrity Guidelines: **You must work this exam on your own. You may use the supplied "cheat sheet", but may not use any other notes, textbook, online resources, or resources of any kind. You may neither receive help nor provide help on this midterm.**

START TIME:

STOP TIME:

ACADEMIC INTEGRITY: I understood and abided by the academic integrity guidelines.

SIGNED: _____

Comments (optional):

Math 155B - Spring 2020 - Midterm #3 - May 14, 2020 - due 4:00pm sharp

1. A ***degree two*** B-spline curve $\mathbf{q}(u)$ has knot vector $0, 0, 0, 2, 2, 4, 6, 8, 8, 8,$ and control points $\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4, \mathbf{p}_5, \mathbf{p}_6$.

Express your answer below ***in terms of these six control points***.

a. What are $\mathbf{q}(2)$ and $\mathbf{q}(8)$ equal to?

b. What are the first and second control points of the degree one B-spline curve representing $\mathbf{q}'(u)$?

c. What is $\mathbf{q}'(0)$ equal to?

d. What is $\mathbf{q}'(2)$ equal to?

e. What is $\mathbf{q}'(1)$ equal to?

2. This question asks you to compare an Overhauser piecewise Bézier interpolating curve and a degree-three B-spline interpolating curve. They are both defined with the same list of control points $\mathbf{p}_0, \ldots, \mathbf{p}_n$. They are also both defined relative to the same values $u_0, \ldots, u_n$. so that, for many values of i, the curve interpolates p_i at $u = u_i$.

Your answers below should address qualitative aspects of the two kinds of curves. For example, properties that might be evident in applications or that a curve designer might notice. (Definitely, do not give complex mathematical formulas!)

(a) List some similarities the two kinds of curves have. That is, what properties do they have in common?

(b) In what ways are the two kinds of curves different?

(c) What advantages do Overhauser interpolating curves have over degree three B-Spline interpolating curves? (If any.)

(d) What advantages do degree three B-Spline interpolating curves have over Overhauser interpolating curves? (If any.)

3. (Algorithm for random permutation.) Give the pseudo-code for an algorithm that generates a random permutation. The input to the algorithm is an integer $N > 2$, and an preallocated integer array $A[]$. The algorithm should fill the array with the values of a random permutation. These values are to be stored in $A[0], A[1], \ldots, A[N-1]$.

You may assume you have available a function $randInt(k)$ that returns a random integer in $\{0, 1, \ldots, k\}$, or a function $rand()$ that returns a random floating point number in the range $[0, 1]$.

Your algorithm should be efficient, and run in time $O(N)$, namely, in linear time.

4. Answer the following "True/False" questions about a shader program with a vertex shader, a geometry shader and a fragment shader. It is assumed that the C++ program issues a `glDrawElements()` command with the option `GL_TRIANGLE_FAN`. The geometry shader is set up to take triangles as input (using `layout(triangles)in;`).

a. Each invocation of the geometry shader has access to all the vertices in the triangle fan.

b. The geometry shader can determine whether the draw command used `GL_TRIANGLE_FAN` instead of `GL_TRIANGLES` or `GL_TRIANGLE_STRIP`.

c. The geometry shader has access to all the vertex attributes of the vertices in the input triangle, even the ones that the vertex shader did not copy to "out" variables.

d. The geometry shader can read the values of uniform variables.

e. The geometry shader can change the values of uniform variables.

f. The geometry shader can tell whether the input triangle lies entirely within the viewable region.

g. The vertex shader can specify that some triangles should be sent directly to the fragment shader without being processed first by the geometry shader.

h. The fragment shader's "in" variables can be used to receive values directly from both the vertex shader and the geometry shader.