Student number	point total	req total	extra total	R1 Bezier (2p)	R2 B-spline (2p)	R3 Gen triangles (2p)	R4 new positions (2p)	R5 old positions (2p)	mod	notes	boundary handling (3p)	local coordinate frames (1p)	surfaces of revolution (3p)	gencyls (3p)	new subdiv schemes (?p)	bezier interp. camera path (4p)	other (put points here)	what other extras?
225157	0	0	0															
270034	4	4	0	2	2					R1, R2: you can make your code a lot more compact by using matrix-vector products and assigning vectors to matrix rows/cols								
293846	0	0	0															
295323	0	0	0															
345642	0	0	0															
348843	0	0	0															
349936	12.5	9.5	3	2	2	2	1.5	2		R4: tiny indexing error - ne should be neighborEdges[i][j] instead of ...[neighborTri][j]. Boundary handling: not working quite as expected with the patch object.	2	1						
350475	4	4	0	2	2													
352091	0	0	0															
353980	7.5	7.5	0	2	2	1.5	2			R1: no need to put basis polynomials (1, t, t*t, t*t*t) inside a matrix, just use the column as a vec4. R3: inconsistent winding order (some tris CW, others CCW). R4: works after fixing R3. Note: please remove .vs (hidden), build, and bin folders before submitting								
354439	11	10	1	2	2	2	2	2		Local coordinate frame: N, B missing		0					1	VCS (1p)
355593	0	0	0															
356026	13	10	3	2	2	2	2	2		Boundary handling for old vertices done in nonstandard way, but seems to work fine	3							
361749	0	0	0															
369181	8	7	1	2	2	1.5	1.5			R3: third pushed tri has opposite winding order to rest. After that, your commented R4 code is almost correct - you still need neighborEdges for getting v3.							1	VCS (1p)
372660	0	0	0															
387370	0	0	0															
425575	0	0	0															
425614	9.5	9.5	0	2	2	1.5	2	2		Record updated in Oodi (based on email thread). R3 question: the last two push_back'd triangles use clockwise vertex ordering, the rest are counter-clockwise. R4: using v3=0 on boundaries is a bit odd (using v3=v2 or skipping both looks better)								
426419	4	4	0	2	2													
427489	10	10	0	2	2	2	2	2										
428022	0	0	0															
429487	0	0	0															
430829	3	2	1	2						R1 (coreBezier): use indexed assignment instead of push_back, that fixes the lines							1	VCS (1p)

Student number	point total	req total	extra total	R1 Bezier (2p)	R2 B-spline (2p)	R3 Gen triangles (2p)	R4 new positions (2p)	R5 old positions (2p)	mod	notes	boundary handling (3p)	local coordinate frames (1p)	surfaces of revolution (3p)	gencyls (3p)	new subdiv schemes (?p)	bezier interp. camera path (4p)	other (put points here)	what other extras?
457598	9.5	9.5	0	2	2	1.5	2	2		R3: your middle to push_back'd triangles have the wrong winding order - this explains the bug you reported								
460297	0	0	0															
464772	0	0	0															
46477D	0	0	0															
46596K	0	0	0															
474199	11	10	1	2	2	2	2	2				1						
474322	13	10	3	2	2	2	2	2		R5, extra: logic seems more complicated than it needs to be, but works fine	3							
474458	7	7	0	2	2	1.5	1.5	0		R3: last two pushed tris are CW instead of CCW. R4: boundary check wrong (>0 should be >=0)								
474898	0	0	0															
475389	0	0	0															
475813	0	0	0															
475910	0	0	0															
477329	4.5	4	0.5	2	2					R2: No need to invert matrices by hand, just call invert(). Local frame extra: continuity not properly handled (using Binit)		0.5						
477811	13	10	3	2	2	2	2	2		Debug crash: you're using nt to index before checking if it's -1	3							
478328	8	8	0	2	2	2	2											
478470	0	0	0															
478687	0	0	0															
479505	11	10	1	2	2	2	2	2									1	Subdiv colors (1p)
479589	10	9	1	2	2	2	2	1		No need to invert matrices by hand, just call invert(). R4: setting v3 to index 0 on boundary a bit strange, using origin as pos or skipping v2&3 looks better. R5: looks a bit off, crashes if mesh has boundaries		1						
479741	15	10	5	2	2	2	2	2			3	1					1	Subdiv colors (1p)
480086	0	0	0															
480248	0	0	0															
480714	10	10	0	2	2	2	2	2		R4: always using globally first vertex on boundary is a bit strange, causes position shift (-0 p)								
480798	0	0	0															
481577	10.5	10	0.5	2	2	2	2	2		Local frame: Binit used all the time, not just for first point. Also not updated based on previous segment. R4: using v2 as v3 on boundary better than using index 0		0.5						
493840	11.5	10	1.5	2	2	2	2	2		R1: would be easier to use an explicit Bezier basis matrix (like the one you invert in R2). Local frame: always using same binormal (Binit), not interatively updated, also no continuity across segments		0.5					1	VCS (1p)

Student number	point total	req total	extra total	R1 Bezier (2p)	R2 B-spline (2p)	R3 Gen triangles (2p)	R4 new positions (2p)	R5 old positions (2p)	mod	notes	boundary handling (3p)	local coordinate frames (1p)	surfaces of revolution (3p)	gencyls (3p)	new subdiv schemes (?p)	bezier interp. camera path (4p)	other (put points here)	what other extras?
506300	10	10	0	2	2	2	2	2		R2: you can just call invert() instead of inverting matrices by hand. R4, R5: you can simplify your code by using neighborEdges								
508285	7	7	0	2	2	2	1			R4: you need neighborEdges for getting v3								
51620U	0	0	0															
524926	7	7	0	2	2	2	1			R4: buggy, you need to read from neighborEdges as well (or use nv + 1 instead of nv + 2)								
525417	13	10	3	2	2	2	2	2			3							
525491	0	0	0															
525941	0	0	0															
526050	10	10	0	2	2	2	2	2										
526319	6	6	0	2	2	2												
526775	0	0	0															
527143	0	0	0															
527389	0	0	0															
528867	0	0	0															
528883	10	10	0	2	2	2	2	2		Your submission was flagged by Windows Defender as containing a trojan. Checked with VirusTotal, it should be clean. Still might want to check your PC.								
529196	0	0	0															
529303	0	0	0															
530185	0	0	0															
530363	0	0	0															
530619	0	0	0															
530648	0	0	0															
530868	11	10	1	2	2	2	2	2		Local frame: the discontinuity of closed loops has to be handled explicitly (by propagating angle error over curve as post process)		1						
530981	11	10	1	2	2	2	2	2									1	Subdiv colors
540094	0	0	0															
540654	0	0	0															
544566	0	0	0															
549040	0	0	0															
549163	11.5	10	1.5	2	2	2	2	2		Local frames: not orthogonal, no continuity across segments. Initial binormal can be arbitrarily chosen, but has to be updated along curve. Normal is the vector orthogonal to T and B. Boundary handling: you need one CW and one CCW walk in case of a boundary (old vertices)	1.5	0						
55055P	17.5	10	7.5	2	2	2	2	2		Local coord frames: you are not propagating the binormal over segments and instead starting each with the static (0,0,1). This causes the flipping.	3	0.5	3				1	Subdivision coloring (1p)
552794	0	0	0															
552969	0	0	0															
554598	10	10	0	2	2	2	2	2										

Student number	point total	req total	extra total	R1 Bezier (2p)	R2 B-spline (2p)	R3 Gen triangles (2p)	R4 new positions (2p)	R5 old positions (2p)	mod	notes	boundary handling (3p)	local coordinate frames (1p)	surfaces of revolution (3p)	gencyls (3p)	new subdiv schemes (?p)	bezier interp. camera path (4p)	other (put points here)	what other extras?
563068	3	3	0	1	2					R1 (evalBezier): no loop over control points, only draws first segment. Local frame: using the problematic second derivative normal instead of the better formulation (see handout appendix). Basis continuity also not ensured across segments.		0						
576149	0	0	0															
585716	0	0	0															
586333	7	7	0	2	2	2	1			R4: logic for v3 is wrong; you can make use of neighborTris[i][j] and neighborEdges[i][j] directly instead of looping. Also, v3 == 0 is a valid configuration and doesn't indicate a boundary								
586702	4	4	0	2	2	0				Your R2 is doing what was intended!								
586980	4.5	4.5	0	2	2	0.5				R3: logic OK, but indexing of vertices is way off								
587316	0	0	0															
587471	19.5	10	9.5	2	2	2	2	2		Gencyls: 'weirder' continuity needed for full points	3	1	3	2.5				
588289	11.5	9.5	2	2	2	2	1.5	2		R4: nEdge: index with i, not v0. Boundary: while loop logic too complex	1						1	Subdiv colors (1p)
589291	0	0	0															
589343	0	0	0															
589848	9.5	9.5	0	2	2	2	1.5	2		R4: almost correct, but you forgot to read v2 from indices (like v0, v1). R5: works (miraculously) even though R4 was wrong.								
590921	0	0	0															
591904	5	4	1	2	2					Local frame: finite diff. derivative is OK, but you can also differentiate the Bezier matrix (B) or the basis functions (1, t, t^2, t^3) analythically		1						
591946	9.5	9.5	0	2	2	2	2	1.5		R5: inside while loop, you're assigning to 'Vec3i triangle' instead of the outer 'int triangle'. After that the implementation almost works (slight diff. to reference still for ico)								
592929	11	10	1	2	2	2	2	2		R5 looks good. It also almost implements boundary handling (double while loop)	1							
593274	10.5	9.5	1	2	2	2	2	1.5		R5: weighs something wrong (looks different from reference on ico), but almost there. Creating your own utilities, such as alternative Mat4 constructors, is totally OK							1	VCS (1p)
593847	0	0	0															
594435	0	0	0															
595926	11	10	1	2	2	2	2	2		For closed loops (sponza, gencyls), continuity at the seam has to be explicitly enforced (by distributing angle error across curve)		1						
595997	0	0	0															

Student number	point total	req total	extra total	R1 Bezier (2p)	R2 B-spline (2p)	R3 Gen triangles (2p)	R4 new positions (2p)	R5 old positions (2p)	mod	notes	boundary handling (3p)	local coordinate frames (1p)	surfaces of revolution (3p)	gencyls (3p)	new subdiv schemes (?p)	bezier interp. camera path (4p)	other (put points here)	what other extras?
596747	0	0	0															
597429	13	10	3	2	2	2	2	2			3							
597623	8	8	0	2	2	1.5	2	0.5		R3: the order in which you push the new indices into new_indices is inconsistent with the indexing scheme. This causes problems in R4, which in itself is correct. R5: reasonable attempt. Instead of if-elses, would have been more elegant to use the modulo operator.								
597937	23.5	10	13.5	2	2	2	2	2		Adaptive step size: reasonable implementation, but has some issues like you state	3	1	3	2.5			4	Adaptive step size (3p), subdiv coloring (1p)
602851	0	0	0															
602893	10	10	0	2	2	2	2	2										
603096	14	10	4	2	2	2	2	2			3	1						
603245	12.5	10	2.5	2	2	2	2	2		Local frame: B not accumulated, not continuous over segments (by updating Binit). Boundary: like you say, you need a second while loop in the other direction	1	0.5					1	Subdiv colors (1p)
604095	56.5	10	46.5	2	2	2	2	2		Gcyl: 'weirder' discontinuity. Catmull-Clark: does not behave like Blender's implementation. Isosurface: lack of proper shading makes it hard to see the shape. Adaptive step size: UI sliders don't really affect the end result	3	1	3	2.5	3	4	30	VCS (1p), Subdiv colors (1p), Catmull-Clark (3p), Catmull-Rom (3p), Isosurface (4p), Other primitive curves (3p), Curvature visualization (3p), Adaptive step size (3p), Piecewise Spline surfaces (3p), Curve scaling (4p), curve editor (5p)
604273	5	5	0	1	1	1.5	1	0.5		R1: last control point shared, use +3 in for loop. R1, R2: look wrong, but core idea is correct. R3: second and fourth tri have wrong winding order. R4: not quite correct. R5: crashes or hangs. Already got VCS points last round, despite note								
606064	9.5	8.5	1	2	2	1.5	2	1		R3: middle two push_back'd tris have wrong winding order. R5: good attempt							1	VCS (1p)
608949	0	0	0															
609155	0	0	0															
609249	0	0	0															
612472	14.5	10	4.5	2	2	2	2	2		Local frame: the discontinuity you noted must be fixed for full points	3	0.5					1	Subdiv colors (1p)
612498	0	0	0															
612870	9	8	1	2	2	2	2	0									1	VCS (1p)
614577	0	0	0															
614580	0	0	0															
621308	8.5	7	1.5	2	2	2	1			R4: for v3, you need to read from neighborEdges: indices[nbTri][(nbEdge + 2) % 3]. Subdiv coloring: not very informative		1					0.5	Subdiv coloring (0.5p)
628835	10	10	0	2	2	2	2	2		The kind of boundary checking you do is exactly what was asked for the requirements!								
63036R	0	0	0															

Student number	point total	req total	extra total	R1 Bezier (2p)	R2 B-spline (2p)	R3 Gen triangles (2p)	R4 new positions (2p)	R5 old positions (2p)	mod	notes	boundary handling (3p)	local coordinate frames (1p)	surfaces of revolution (3p)	gencyls (3p)	new subdiv schemes (?p)	bezier interp. camera path (4p)	other (put points here)	what other extras?
641922	0	0	0															
646804	10	10	0	2	2	2	2	2										
647764	8	8	0	2	2	2	2			Already got VCS points last round, no need to include git log								
648530	0	0	0															
648860	0	0	0															
650191	10	10	0	2	2	2	2	2										
650227	0	0	0															
650405	0	0	0															
650560	10	10	0	2	2	2	2	2										
650942	10	10	0	2	2	2	2	2										
651527	4.5	4	0.5	2	2	0	0	0		Local coord frames: discontinuos over segments. You are using the last control point as the binormal for i>0?? No need to compute the matrix--vector products by hand! Just use the built-in operators.		0.5						
651585	8	8	0	2	2	2	2	0		Sorry, R5 too far from anything working								
651637	9.5	9.5	0	2	2	2	2	1.5		R5: looks like you were only missing brackets in one of your if-statements. Hence you were hitting break immediately in the first iteration.								
651789	0	0	0															
652102	11	10	1	2	2	2	2	2				1						
652131	4	4	0	2	2	0	0	0										
652335	14.5	10	4.5	2	2	2	2	2		Local coord frames: discontinuities between segments as seen in campath.		0.5	3				1	Subdivision coloring (1p)
652649	10	10	0	2	2	2	2	2										
652898	0	0	0															
652937	10	10	0	2	2	2	2	2										
653127	9.5	9.5	0	2	2	2	2	1.5		R5: old vertices shift slightly to the side (see icosahedron) - one vertex is probably missed or counted twice								
653596	0	0	0															
653693	10	10	0	2	2	2	2	2		Subdivision is somewhat slow. You don't really need to store the neigbors of the even vertices in R5. Accumulating the positions and keeping track of the number of neighbors is enough.								
653871	0	0	0															
653907	10	10	0	2	2	2	2	2										
653910	10	10	0	2	2	2	2	2										
654595	25.5	10	15.5	2	2	2	2	2		Gencyls: 'weirder' discontinuity	3	1	3	2.5			6	Subdiv colors (1p), Catmull-Rom (1p), parametric curves (4p)
655057	12.5	10	2.5	2	2	2	2	2		VCS: it does look like you were given the point last round already. Boundaries are not quite right. There is a problem with at least corners of the patch object.	2.5							

Student number	point total	req total	extra total	R1 Bezier (2p)	R2 B-spline (2p)	R3 Gen triangles (2p)	R4 new positions (2p)	R5 old positions (2p)	mod	notes	boundary handling (3p)	local coordinate frames (1p)	surfaces of revolution (3p)	gencyls (3p)	new subdiv schemes (?p)	bezier interp. camera path (4p)	other (put points here)	what other extras?
655086	10.5	9.5	1	2	2	1.5	2	2		R3: pushed tris have inconsistent winding order, makes R4, R5 logic more complicated than it needs to be	1							
655109	10.5	9.5	1	2	2	2	2	1.5		R5: not using connectivity information, unnecessarily heavy O(n^2) nested loops							1	VCS (1p)
655251	0	0	0															
655264	0	0	0															
655471	10	10	0	2	2	2	2	2		R5: works but is really slow, as you said. Other attempts had the correct~ish idea, but you seemed to be overthinking it a bit. You were almost there, good effort!								
655691	0	0	0															
655853	8	8	0	2	2	2	2	0										
656250	8	8	0	2	2	2	1	1		R4: neighborEdges should be indexed with [i][j] directly, result offset and modulo'd. Also, you create a new variable v3 that is immediately discarded. R5: good try								
656454	14	10	4	2	2	2	2	2			3	1						
656616	7	7	0	2	1	2	2			R2: for loop should increment by +1, use partially overlapping points								
657291	13	10	3	2	2	2	2	2			3							
657314	0	0	0															
657327	8	8	0	2	1.5	2	2	0.5		R2: curve initialized or indexed wrong, which leads to zero vertices (lines to origin). R5: right idea, but quite far from working solution. Version control log picture missing from zip!								
657482	0	0	0															
657796	10	10	0	2	2	2	2	2										
657893	7.5	7.5	0	2	2	1.5	2	0		R3: inconsistent winding order, some triangles CW, others CCW								
659914	8	8	0	2	2	1.5	2	0.5		R3: inconsistent winding order (some triangles CW and other CCW). R5: somewhat reasonable attempt: can't assume that n=6 always!								
660246	13.5	10	3.5	2	2	2	2	2		Local frame: B not continuous across segment boundaries (by updating Binit), causes campath hitching and flipping. Boundary handling: you need a second while loop in the opposite direction	2	0.5					1	Subdiv colors (1p)
660877	0	0	0															
660893	0	0	0															
663191	0	0	0															
663272	0	0	0															
665380	14	10	4	2	2	2	2	2			3						1	VCS (1p)
665898	10	10	0	2	2	2	2	2										
666172	11	10	1	2	2	2	2	2									1	Subdivision coloring (1p)
666350	10	10	0	2	2	2	2	2										

Student number	point total	req total	extra total	R1 Bezier (2p)	R2 B-spline (2p)	R3 Gen triangles (2p)	R4 new positions (2p)	R5 old positions (2p)	mod	notes	boundary handling (3p)	local coordinate frames (1p)	surfaces of revolution (3p)	gencyls (3p)	new subdiv schemes (?p)	bezier interp. camera path (4p)	other (put points here)	what other extras?
666680	9	9	0	2	2	2	2	1		R5: reasonable attempt, logic somewhat too complex. No need for nested while-loops!								
667249	10	10	0	2	2	2	2	2										
67137M	0	0	0															
67627H	13	10	3	2	2	2	2	2			3							
677734	0	0	0															
678089	0	0	0															
68933B	0	0	0															
69247N	0	0	0															
700436	0	0	0															
705570	62	10	52	2	2	2	2	2		Maybe a bit inelegant to make the closed loop coord system correction entirely at the end of the loop instead of distributing evenly. Curve editor is really bare-bones and somewhat difficult to use, but does what it is supposed to. Attempt at isosurface extraction.	3	1	3	3	6	4	32	Adaptive step size (4p), Subdivision coloring (1p), Srev textures, curvature visualization (5p). Catmull--Rom splines (3p), Trefoil, corkscrew curves (5p), Piecewise beziers (3p) Cylinder curve scaling (4p), Isosurface extraction (2p), Curve editor (5p)
706566	2	2	0	2	0	0	0	0										
708784	19.5	10	9.5	2	2	2	2	2		You just have to manually take the discontinuity into account. Compute the difference in the orientation and distribute the correction evenly along the curve.	3	1	3	2.5				
708904	0	0	0															
708920	10	10	0	2	2	2	2	2										
708933	0	0	0															
709291	7.5	7.5	0	2	2	1	1	1.5		R3: I don't quite follow your logic. What you had commented out was closer to the correct answer. R4: you are not using the edge information at all and not necessarily finding the correct vertex. R5: logit at least almost right, the way you compute n is quite involved and heavy.								
709628	14	10	4	2	2	2	2	2			3						1	Subdivision coloring (1p)
710086	18	10	8	2	2	2	2	2			3	1	3				1	VCS (1p)
710497	17	10	7	2	2	2	2	2			3	1	3					
710743	17	10	7	2	2	2	2	2		Srevs: As the appendix says, the direction of the normals depends e.g. on the direction that you are moving on the curve. Flipping the normals may be necessary to follow same conventions as the example, as you've done here!	3	1	3					
710976	9	8	1	2	2	2	2	0									1	VCS (1p)
711182	8	8	0	2	2	2	2	0										
711467	23.5	10	13.5	2	2	2	2	2		Gencyl: discontinuity with weird (er).	3	1	3	2.5		4		

Student number	point total	req total	extra total	R1 Bezier (2p)	R2 B-spline (2p)	R3 Gen triangles (2p)	R4 new positions (2p)	R5 old positions (2p)	mod	notes	boundary handling (3p)	local coordinate frames (1p)	surfaces of revolution (3p)	gencyls (3p)	new subdiv schemes (?p)	bezier interp. camera path (4p)	other (put points here)	what other extras?
711551	16.5	10	6.5	2	2	2	2	2		Local coord frames: discontinuities between segments - you are not propagating the binormal between segments. Srev: tiny boundary mistake (-0p) - there seems to be edges between the first and last points of the curve (see norm.swp in wireframe)	3	0.5	3					
711810	4	4	0	2	2	0	0	0										
711904	0	0	0															
712550	10	10	0	2	2	2	2	2										
712686	0	0	0															
712819	4	4	0	2	2	0	0	0										
712958	0	0	0															
713672	0	0	0															
714985	11	10	1	2	2	2	2	2		Local coord frames: what you have looks quite reasonable to me, can't give any points for fully commented out code, though. There is an UI-element for hiding the resulting coord frames anyway.							1	Subdivision coloring (1p)
716080	4	4	0	2	2	0	0	0										
716462	13.5	10	3.5	2	2	2	2	2		Local coord frames: lack of B and N starts from losing T in the intersection of segments. There seems to be some indexing issue near the intersections that causes the Geometry*Spline matrix to be singular in a way which causes the its product with the basis-derivative to be exactly the zero-vector.	3	0.5						
716718	0	0	0															
716860	10	10	0	2	2	2	2	2										
717377	19.5	10	9.5	2	2	2	2	2		You were storing the color of a vertex as the normal on boundaries giving a somewhat weird-looking shading. In wireframe everything was correct. Gencyls: weirder discontinuity. Reading the instructions first is indeed a good skill to have!	3	1	3	2.5				
717474	0	0	0															
717513	13	10	3	2	2	2	2	2		Boundaries: you were using the color-vector in the computation of normals in the boundaries resulting in somewhat weird shading (-0p). Local coord frames: the derivative of the basis is wrong - we take the derivative w.r.t. t=i/s, not just i. You are using elementwise multiplication instead of the cross product for N and B.	3	0						
717539	0	0	0															
718020	0	0	0															
718871	0	0	0															
722427	0	0	0															

Student number	point total	req total	extra total	R1 Bezier (2p)	R2 B-spline (2p)	R3 Gen triangles (2p)	R4 new positions (2p)	R5 old positions (2p)	mod	notes	boundary handling (3p)	local coordinate frames (1p)	surfaces of revolution (3p)	gencyls (3p)	new subdiv schemes (?p)	bezier interp. camera path (4p)	other (put points here)	what other extras?
723691	8.5	7.5	1	2	2	1.5	2	0		R3: inconsistent winding order of new triangles. Some are defined CCW, some CW, so the indexing breaks down. After fixing this, R4 works correctly.							1	VCS (1p)
723905	0	0	0															
728667	14	10	4	2	2	2	2	2			3						1	Subdivision coloring (1p)
728900	11	10	1	2	2	2	2	2									1	VCS (1p)
729132	0	0	0															
729967	8.5	8.5	0	2	2	1.5	2	1		R3: Some new triangles CW, other CWW, after this stuff upto R4 works correctly. R5: reasonable attempt, however it looks like you were at least computing n wrong, since you are including the center-vertex to those via the v.size(). (you don't really need to store the indices to a vector, just accumulate the positions during the while loop!)								
730309	0	0	0															
732080	0	0	0															
732255	0	0	0															
732323	0	0	0															
732336	0	0	0															
732352	0	0	0															
732459	0	0	0															
76509T	28.5	10	18.5	2	2	2	2	2		Gencyls: closed loop continuity not ensured, check 'weirder'. Love the clean code!	3	1	3	2.5		4	5	Subdiv colors (1p), Marching cubes (4p)
765510	19.5	10	9.5	2	2	2	2	2		Local coord frames: you are not propagating the binormal over segments but instead using either (0,0,1) or a random vector if T was +/- (0,0,1). This causes discontinuities.	3	0.5	3	3				
766331	0	0	0															
767042	0	0	0															
767136	7.5	7.5	0	2	2	1.5	2	0		R3: inconsistent winding order: some CW, other CCW.								
768504	0	0	0	0	0	0	0	0		Submission had no source files :(								
769396	11	10	1	2	2	2	2	2									1	Subdivision coloring (1p)
77388B	8	8	0	2	2	2	2											
779124	0	0	0															
780058	8	8	0	2	2	2	2	0										
780346	0	0	0															
782917	0	0	0															
783563	0	0	0															
783709	0	0	0															
786667	0	0	0															
78708M	0	0	0															
787543	0	0	0															
787640	0	0	0															
788380	0	0	0															
788678	0	0	0															
791982	0	0	0															

Student number	point total	req total	extra total	R1 Bezier (2p)	R2 B-spline (2p)	R3 Gen triangles (2p)	R4 new positions (2p)	R5 old positions (2p)	mod	notes	boundary handling (3p)	local coordinate frames (1p)	surfaces of revolution (3p)	gencyls (3p)	new subdiv schemes (?p)	bezier interp. camera path (4p)	other (put points here)	what other extras?
795700	0	0	0															
795755	0	0	0															
796039	0	0	0															
804183	19.5	10	9.5	2	2	2	2	2		Closed loop discontinuity needs to be handled manually by distributing the angle error evenly across the curve.	3	1	3	2.5				
829155	0	0	0															
838191	0	0	0															
83873J	0	0	0															
84308F	0	0	0	0.5	0.5				-1	Submission contains a single c++ file, no readme, no VS project files (-1). R1, R2: crashes in debug mode.								
84858E	0	0	0															
848754	22	10	12	2	2	2	2	2		Local coord frames: you are not propagating the binormal over segments -> discontinuity in campath + weird(er).swp! Cam-path: the camera is completely erratic at segment boundaries!	3	0.5	3	2.5		2	1	Subdivision coloring (1p)
875170	14	10	4	2	2	2	2	2			3						1	Subdivision coloring (1p)
875251	10.5	10	0.5	2	2	2	2	2		Local coord system: If you take N = BxT, then you should have B = TxN (instead of B = NxT) to avoid flipping one of your axes at each step.		0.5						
875303	0	0	0															
875617	0	0	0															
876399	0	0	0															
877107	0	0	0															
877152	17.5	9.5	8	2	2	2	2	1.5		R5: looked like you were double counting a single vertex resulting in a slightly wrong answer. Boundaries: not quite the expected results compared with the reference. Local coord frames: you are not propagating the binormal between segments resulting in discontinuities (e.g. in campath). Gencyls: weirder discontinuity.	2	0.5	3	2.5				
878591	0	0	0	0	0	0	0	0		Could not find the source in the submission :(It seeming has everything else, including the framework source.								
878627	9.5	9.5	0	2	2	2	2	1.5		R5: you are double counting the two vertices of the triangle you are starting at, resulting in a slighly wrong answer. This is annoying since it does not show up in highlighted indices, unless you specifically check its size. You should actually be able to see what only the requiremets should look like: the icosahedron has no boundaries so it is unaffected by the extra.								
878889	8	8	0	2	2	2	2	0										
879105	9.5	9.5	0	2	2	2	1.5	2		R4: indexing logic not quite right. After fixing this, R5 works correctly.								

Student number	point total	req total	extra total	R1 Bezier (2p)	R2 B-spline (2p)	R3 Gen triangles (2p)	R4 new positions (2p)	R5 old positions (2p)	mod	notes	boundary handling (3p)	local coordinate frames (1p)	surfaces of revolution (3p)	gencyls (3p)	new subdiv schemes (?p)	bezier interp. camera path (4p)	other (put points here)	what other extras?
882134	0	0	0															
885128	8	8	0	2	2	2	2	0		R4: looks right to me. patch.obj is not required to work perfectly for requirements.								
886648	10.5	9.5	1	2	2	2	2	1.5		Subdivision seems to work, but is very slow since you are looping throgh all vertices in search of the neighbors. You can do this with ~10 lines of code using the connectivity information. Using floats as keys in std::map seems like playing with fire to me.	1							
889645	0	0	0															
892292	0	0	0															
898351	4	4	0	2	2	0	0	0										
899130	0	0	0															
900016	0	0	0	0	0	0	0	0		You returned assignment 1 again?								
901170	0	0	0															
901196	0	0	0															
913249	0	0	0															
913333	0	0	0															
913346	0	0	0															
913566	14	10	4	2	2	2	2	2			3	1						
915221	0	0	0															
915250	0	0	0															
917863	23.5	10	13.5	2	2	2	2	2		Gencyls: weird(er) discontinuity.	3	1	3	2.5		4		
918150	0	0	0															
918228	10	9.5	0.5	2	2	2	1.5	2		R4: slight indexing error - instead of neighborEdges[neighbor][j] use ...[i][j]. R5: missing a times B when computing new colors and normals (-0p). Local coord frames: you are not propagating the binormal over segments but starting each with the static (0,0,1). Hence the flipping.		0.5						
918257	0	0	0															
918309	14.5	10	4.5	2	2	2	2	2		Local coord frames: some discontinuities at segment boundaries, can be seen as flipping camera in the camerapath. You can just multiply a matrix and vector together with operator * : no need to do elementwise with dot-products!	3	0.5					1	Subdivision coloring (1p)
918396	9	9	0	2	2	2	2	1		R5: slight weight issues, you are probably counting some vertex multiple times or missing one. patch.obj crashes the program already at first subdivision, bunny after a couple of subdivisions. Surf. revolution: can't find code for this?	0		0					

Student number	point total	req total	extra total	R1 Bezier (2p)	R2 B-spline (2p)	R3 Gen triangles (2p)	R4 new positions (2p)	R5 old positions (2p)	mod	notes	boundary handling (3p)	local coordinate frames (1p)	surfaces of revolution (3p)	gencyls (3p)	new subdiv schemes (?p)	bezier interp. camera path (4p)	other (put points here)	what other extras?
918464	33.5	10	23.5	2	2	2	2	2		gencyls: weirder discontinuity. Some of the task are much worse than other when it comes to pure credit/effort metric. This is especially true for some of the more 'exotic' extras. Feel free to try to optimize this, but please also try to have fun and enjoy learning new things!	3	1	3	2.5	5		9	Subdivision coloring (1p), Catmull-Rom splines (3p), Non-spline curves (5p)
918671	13.5	10	3.5	2	2	2	2	2		It looks like you are losing the tangent at single points at the boundaries of segments. Since the two other directions are cross products with this (zero) vector, they also vanish.	3	0.5						
918875	0	0	0	0	0	0	0	0		Looks like you returned assignment 1 again?								
930484	7	7	0	2	2	2	1	0		R4: reasonable attempt, some additional debugging required with the boundaries.								
932440	9.5	9.5	0	2	2	2	1.5	2		R4: somewhat too complex logic resulting in a slight indexing mistake.								
935625	4	4	0	2	2	0	0	0										
939375	11	10	1	2	2	2	2	2									1	Subdivision coloring (1p)
942618	15	10	5	2	2	2	2	2		Local coord frames: the loop in evalBezier seems to always end after first iteration. Hence you are not ever using the last Binit (that you correctly attempt to use for the next segment) and always using the initial (0,0,1). This causes camera flipping in campath.	1.5	0.5	3					
k28342	0	0	0															