NATIONAL UNIVERSITY OF SINGAPORE
SCHOOL OF COMPUTING

EXAMINATION FOR
Semester 2, 2011/2012

CS3241 - COMPUTER GRAPHICS

April 2012 Time Allowed: 2 Hours

INSTRUCTION TO CANDIDATES

1. This is an OPEN book examination.
2. This examination paper contains **SEVEN (7)** questions and comprises **EIGHT(8)** printed pages.
3. Answer *ALL* questions within the spaces provided in this booklet.
4. You are allowed to use the back of the paper but please remember to state "P.T.O."
5. *Cross out any draft* or otherwise we will mark the poorer answers.
6. No calculator should be needed in this exam. You can leave your answer in surd form, namely, you can write $\sqrt{2}$ instead of 1.4142...
7. **In the programming questions, you can assume some basic normal vector arithmetic functions are provided for the 2D or 3D vector class `Vector2D (Vector3D)`, even a function `normalize(Vector2D)` to compute a normalized vector.**
8. Please write your matriculation number below, but NOT your name.

TIDINESS COUNTS!

We will deduct marks if your writing is too messy.

MATRICULATION NUMBER:_____

(this portion is for examiner's use only)

Question	Max. Marks	Score	Check
Q1	10		
Q2	6		
Q3	8		
Q4	6		
Q5	10		
Q6	6		
Q7	6		
Total	52		

Question 1 [10 marks]

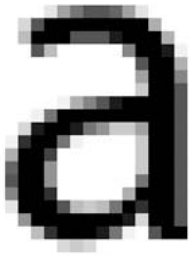
You are given 10 statements below. State if each statement is True (T) or False (F) by writing your answer in the box provided. A correct answer will give you 1 mark. An empty answer will give you zero, but ***a wrong answer will result in -1 mark***. (The lowest mark for this whole question is ZERO)

	Environmental mappings are used to create motion blurring.
	If two parametric curves are C1 continuous, they must be G1 continous.
	In OpenGL, you can effectively change the model view matrix inside a pair of glBegin() and glEnd().
	In OpenGL, you can effectively change the type of primitive inside a pair of glBegin() and glEnd(), for example, changing from drawing a polygon to a line loops.
	With Ground shading, a circular spotlight may become polygonal on a flat surface.
	If we sample a continous signal with four times its maximum frequency, we can reconstruct the signal by the samples correctly.
	If we do not care about computational time, the best way to render hairs with raytracing is to model each piece of hair by a Bezier curve.
	It is preferable to represent a sphere by its analytic equation rather than a triangle mesh in ray tracing.
	If the whole scene only contains one piece of polygon that is transparent, z-buffer algorithm can render the scene correctly.
	Normal 3D games, e.g. first person shooting games, are using orthographic projections.

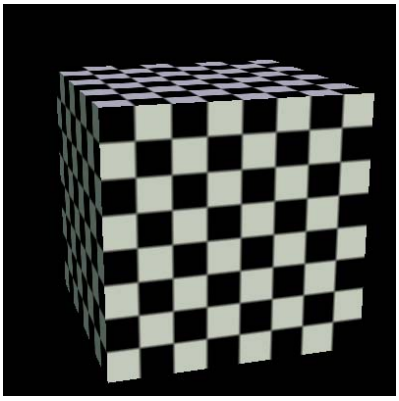
Question 2 [6 marks]

Describe the best techniques taught in our course to draw the following scenes. For each picture, give ONE most important and unique technique.

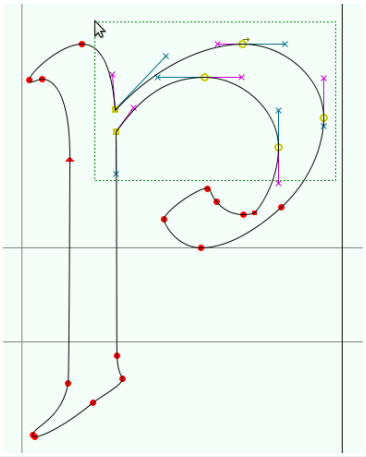
↑



↑



↑



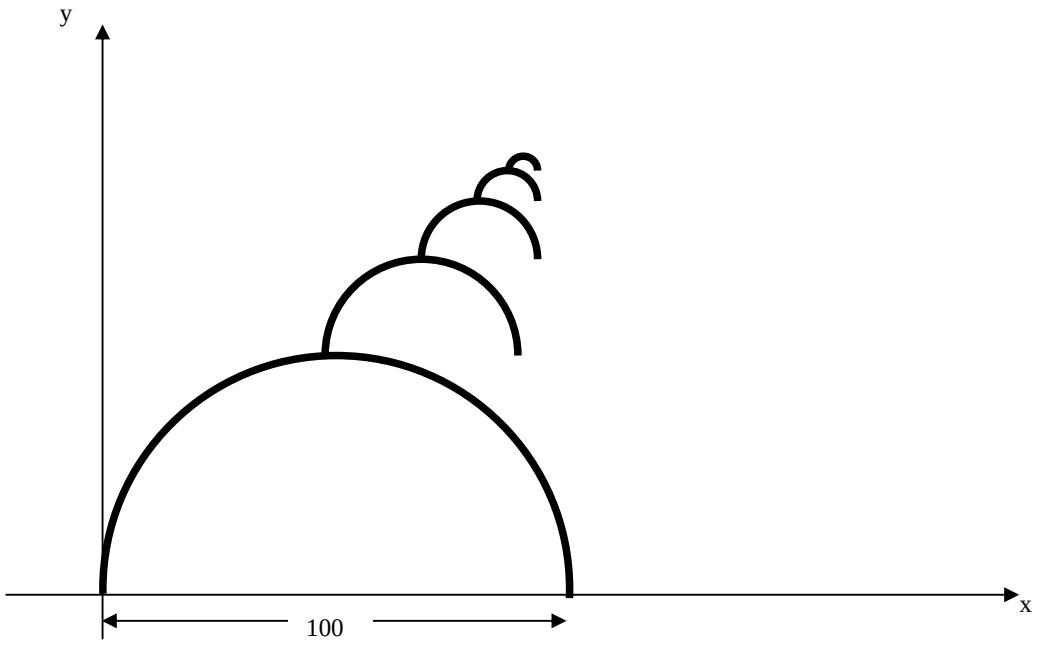
↑

↑

↑

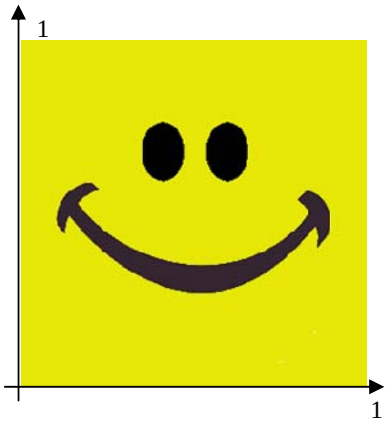
Question 3 [8 marks]

Assume a function `drawHalfCircle()` is given and it draws an upper half (y > 0) circle around the origin with radius 100. Write OpenGL code to draw the following pattern (only the half circles that are thickened), with "for-loops", that is, no recursion. The half circle shrinks by half in length for each iteration.



Question 4 [6 marks]

We want to create a 3D smiley ball with a sphere. The following are the texture image data on the left (you can assume that it is already loaded into the memory and enabled by OpenGL already) and what it should look like in the middle. The smiley is only on one **half** of the sphere (like the right picture, which does not matter on which half). The follow code for drawing the sphere is provided for you. Fill in the empty space on how to assign the texture coordinates and rules for the pattern.



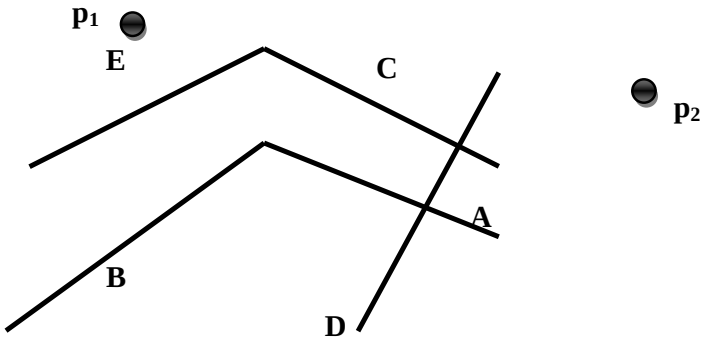
```
void drawSphere(double r)
{
  glDisable(GL_NORMALIZE);  // Disable the automatic normalization of normal vector
  int i,j;  int n = 20;  double x1,x2,x3,x4,y1,y2,y3,y4,z1,z2,z3,z4;

  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, _____________________);
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, _____________________);

  for(i=0;i<n;i++)
    for(j=0;j<2*n;j++)
      { // Smooth shading code here
        x1 = sin(i*M_PI/n)*cos(j*M_PI/n);
        y1 = cos(i*M_PI/n)*cos(j*M_PI/n);
        z1 = sin(j*M_PI/n);
        x2 = sin((i+1)*M_PI/n)*cos(j*M_PI/n);
        y2 = cos((i+1)*M_PI/n)*cos(j*M_PI/n);
        z2 = sin(j*M_PI/n);
        x3 = sin((i+1)*M_PI/n)*cos((j+1)*M_PI/n);
        y3 = cos((i+1)*M_PI/n)*cos((j+1)*M_PI/n);
        z3 = sin((j+1)*M_PI/n);
        x4 = sin(i*M_PI/n)*cos((j+1)*M_PI/n);
        y4 = cos(i*M_PI/n)*cos((j+1)*M_PI/n);
        z4 = sin((j+1)*M_PI/n);

        glBegin(GL_POLYGON);


          glVertex3d(r*x1,r*y1,r*z1);


          glVertex3d(r*x2,r*y2,r*z2);


          glVertex3d(r*x3,r*y3,r*z3);


          glVertex3d(r*x4,r*y4,r*z4);

      glEnd();
      }
}
```

Question 5 [6 + 4 marks]

The following FIVE lines are the cross sections of five polygons in a 3D space.

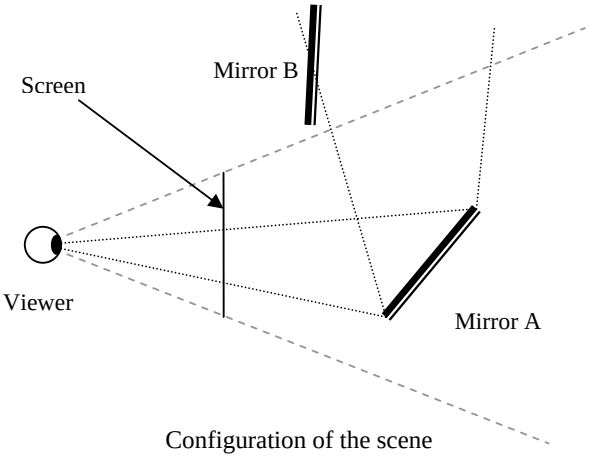
a) Construct their BSP tree according to the priority A > B > C > D > E.
b) Give the order of polygons to be drawn from the viewpoints p_1 and p_2

(Use dash lines in the diagram to help AND show your construction)

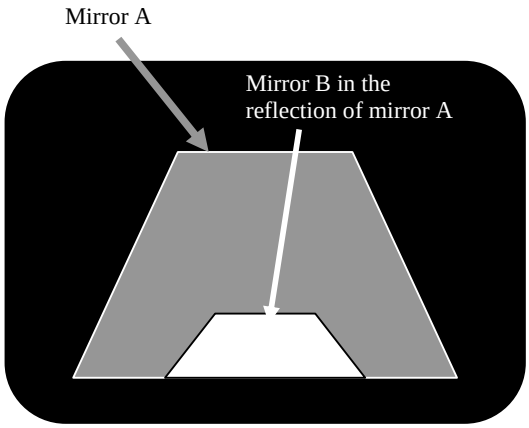


Question 6 [6 marks]

There are two mirrors in the space and a viewer. Assuming that the viewer can only see Mirror **A** directly in the screen and see Mirror **B** in the reflection of Mirror A, as the following configuration on the left.



Configuration of the scene



Picture to be seen on the screen
(Not drawn in the correct scale)

Assuming there is no light source nearby but only some ambient lighting. The viewer should see the following picture in the screen on the right picture. You can notice that the viewer cannot see Mirror **B** directly on the screen, nor any other object 'in' Mirror **B**.

If we render the picture with ray tracing, **how many times do we need to test if a ray intersects an object**? Assuming the screen dimension is 1024x768 pixels. The grey area in the picture has 200000 pixels (excluding the white area) and the white area has 60000 pixels. Assuming this is the most basic ray tracing without any other techniques to reduce the number of ray-object intersection test. However, you can assume a ray does not intersect the object which the ray just comes out from. (Note that the viewer is NOT an object in the scene.) **List the steps and explain your answers also**.

Question 7 [6 marks]

In the 2D plane, given two Bezier curves with the same degree `d`, their control points are stored in two arrays `cp1[d+1]` and `cp2[d+1]`. Write a piece of C++ code to check if they are G1 continuous. Note that each curve has two end points and they may join in different ways (four combinations). The variable `d` should be a "changeable" valuable.

You can assume that miraculously there is no numerical error. Feel free to write more than one function to modulate your algorithm. You can also assume that the class `Vector2D` is well implemented with some operations such as vector addition. To access the coordinate of a class `Vector2D v`, you can use `v.x` and `v.y`.

```
bool checkG1Continuity(int d, Vector2D cp1[], Vector2D cp2[])
{
     // return true if the two Bezier curves are G1 continuous.
```

--- END OF PAPER ---