T-637-GEDE Game Engine Architecture
Problem Set 2- Due Friday March 20th, 2015

Problem 1 – Design and Human Interface Devices (25%)

Human interface devices (HIDs) bring game players into the heat of action; they are therefore an incredibly important piece in the whole game experience. But how do you know whether a certain HID is good for a certain game? In an attempt to understand the connection between game engine design and the wide variety of HIDs available, discuss the following two questions:

a) If you are designing a game engine for a certain genre (you can pick one you like, e.g. RPG, Sports, RTS, etc.), what range of HIDs should you support? Do certain HIDs fit that type of game better than others? Why or why not?
b) If the game world and game play is complex, how can the game engine help developers make use of a very simple HID (think of one of the simplest possible HIDs: one button)? Would this work for every game genre?

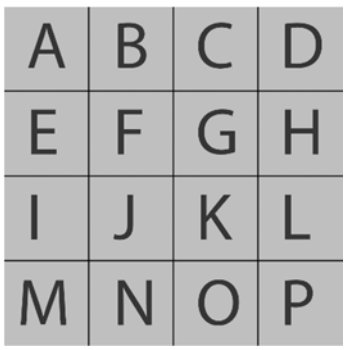
Problem 2 – Hashing Game Object Names (25%)

Consider a game engine where all game objects are identified at design time using a short string, maximally 10 characters in length. We want to avoid as much as possible to use string comparisons in our code to identify objects and therefore decide to hash each string into a unique 32 or 64 bit integer (your choice) for faster comparisons.

a) Create OR find two different hash functions and calculate the hash value for the IDs "VEHICLE1", "VEHICLE2","AMOB" and "BMOB".
b) Now imagine you have a fixed hash table of 1024 entries where you wish to store pointers to the actual objects along with their string IDs. What indices in the table do the above IDs map onto? (If probing is necessary, explain what probing method you are using.)
c) What are the advantages and disadvantages of using each of the two has functions you used?

Problem 3 – Textures as Sprite Maps (25%)

Sprite maps are a useful way to reduce the number of textures the GPU has to have in memory, thus also reducing the number of loading in and out of memory. Sprite maps work by including many different sprites/images in a single texture and then addressing individual elements by offsetting UV coordinates.



Given the texture on the right, **give the set of UV coordinates** to write the word **"GEDE"** on a set of four quads in sequence. Vertices of each quad are numbered and processed **counter-clockwise**, starting in the **upper-left**. You can assume that the texture coordinate **(0,0) refers to the lower left corner of the texture**.

Problem 4 – Shader Programs (25%)

The following **vertex program** is not doing its job properly, please **explain why not,** and propose what has to be done so it will function as expected.

```
void main_time_vp(
        float4 vtx_position         : POSITION,
        float2 vtx_texcoord0        : TEXCOORD0,
        uniform float t,
        out float4 l_position       : POSITION,
        out float2 l_texcoord0      : TEXCOORD0)

{
    float4 temp = vtx_position;
    temp.y = temp.y+cos(temp.x+t);
    l_position = temp;
    l_texcoord0 = vtx_texcoord0;
}
```