Part I
Computer graphics

The computer graphics part is written in C and compiles using make. Most of the code is contained in the file `assignment.c`, with the exception of the math library given for excercise 1.

The code draws a square and a pyramid. The pyramid rotates around an invisible origin, as well as rotating around its centerpoint. The square rotates around the pyramid, always having the same face towards the pyramid. Pressing the space button will change the center of the camera, from either being centered on the square, the pyramid or the invisible origin.

1 Rendering using scenegraph traversal

The scenegraph is constructed in a very similar fashion to the scene graph given in excercise 2. Each node in the graph is a struct. This node contains a (single) transformation matrix (`transform`), a function pointer to a function that *draws* that node (`draw`), an id (`object_no`), an int containing the number of children (`children_count`), an array with pointers to the child nodes (`children`) and finally a pointer to a parent (`parent`).

The id is not used and is a remnant of the scene graph given in the excercise. Its use is replaced by a pointer to a function. I felt this made the `traverseGraph()`-function easier to read and easier to extend.

Nodes are created with the `make_node()` function which takes the address of a function as a parameter. Children are added to nodes with the `add_child()` function and removed with the `remove_child()` function. The two functions `destroy_node()` and `destory_node_rec()` both delete a node (from memory), where the latter also deletes a nodes children.

Objects are then rendered by traversing this scene graph, by calling the function `traverseGraph()` recursively. This function takes a `node` pointer as a parameter, and for each invocation of the function, the transformation matrix found in the node is pushed on the stack. Then the `draw()` function for the node is called, and the `traverseGraph()` function is called with each of the nodes children as the parameter. As the last step of the traversal the nodes transformation matrix is popped off the stack, leaving the stack as it was before the function was called.

2 Extracting transformations from scenegraphs

3 Per-pixel lighting

Part II
Image processing