The game I am creating is a recreation of the first playable level in the original 007 Goldeneye game for the N64.

**Variables**

There will be a number of global variables that will be accessed throughout the program. These will be defined with the EXTERN keyword.

1. **Settings –** This variable will contain a data structure for the game’s settings, containing the screen resolution set to play.
2. **Menu background** – This variable will contain a texture object for the menu’s background. This has to be an extern variable because Main.cpp and Game.cpp need access to it when displaying loading screens and menu screens.
3. **Loading** – This variable will contain the texture object for the text ‘loading’. This needs to be an extern for the same reasons as menu background
4. **Full screen** – This is a Boolean variable to determine whether the game is currently running in full screen or not.
5. **Is running** - This is a Boolean variable to determine if the game is running or not. If not, the game will close.
6. **Screen width, height and BPP** – these integer variables will hold the game resolution
7. **FPS lock**  - This float variable will hold the maximum frames per second the game will run at
8. **Camera**  - This variable will hold the camera data structure
9. **Max screen res** – This variable will hold the maximum screen resolution the computer can have. This will be used in the settings page in menu to stop the user from changing the screen resolution higher than it can
10. **Move Speed –** This variable will hold the player’s speed
11. **Player**  - This will hold the player variable
12. **Screen Ratio** – This will hold the current screen ratio from 800x600

**Data types**

There will be a number of custom data structures that I will create to easily handle the use of a collection of data.

1. **Vector2** - This data structure will hold an x and y co-ordinate
2. **Vector3 -** This data structure will hold an x, y and z co-ordinate
3. **Vector4 -** This data structure will hold an x, y, z and angle co-ordinate
4. **Size -** This data structure will hold a width, height and depth variable
5. **Matrix1** – This data structure will hold a 4x1 matrix array
6. **Matrix4** – This data structure will hold a 4x4 matrix array
7. **PlayerInfo –** This data structure will hold the player info
8. **WeaponInfo –** This data structure will hold the weapon info
9. **Setting** - This data structure will hold variables for the settings loaded in from ini files
10. **Waypoint -** This data structure will hold Vector3 variables for position, view and up co-ordinates which will be used to control the camera
11. **Rect –** This data structure will hold x1, x2, y1 and y2 coordinates.

**Functions**

There are many functions that will be included in the game, most of them are generic and specific for its class such as ‘drawX’ for the class to draw it. For example ‘drawModel’ draws the model.

Below is a list of functions that are specific for the game rather than the classes.

1. **MainLoop**
2. **resizeScene**
3. **toggleFullScreen**
4. **Conversions (toString, toBool, toInt)**
5. **checkMouse**
6. **toggleFire**
7. **checkInputs**
8. **checkMovement**
9. **generate textures**
10. **loadFor2d**
11. **loadFor3d**
12. **lockFPS**
13. **renderGame**
14. **renderPlayer**
15. **renderGame2D**
16. **renderScreen**
17. **button presses**
18. **show menu**
19. **handleLeftClick**

**Classes**

There will be a few classes I will use in my game to give the ability to create objects and use them as unique items in the game.

1. **AI** – This class will contain the functions as settings needed to control artificial intelligence
2. **Camera –** This class will contain the functions and settings for the camera
3. **FontFace –** This will contain the texture for a font face that will be used to render text
4. **TextString** – This will be used to render text
5. **Map** – This class will be used for loading in models and AI objects into a landscape area for the map.
6. **Model** – This class will be used to load in models and handle the animation
7. **Object** – This class will be used to create anything needed to be rendered in OpenGL (minus models)
8. **Player** – This class will contain settings for the player
9. **Settings** – This class will be used for loading in settings from an INI file
10. **Sound** – This class will be used for handling sound
11. **Texture** – This class will be used for loading and handling textures
12. **UI** – This class will be used for the UI items in the menu such as buttons and sliders
13. **Weapon** – This class will be used to load in weapon settings

**Game Requirements (Minimum)**

**OS:** Microsoft Windows XP/Vista/7   
**RAM:** 1GB   
**Graphics:** 256mb nVidia graphics card or above with OpenGL 3.0 or above support  
**Processor:** 1.5GHz Intel Pentium or above (32bit x86)  
**HDD:** 128mb hard disk space  
**Joystick:** XBOX 360 wired or better controller with support to XINPUT  
  
**Install Instructions**

Copy contents (or folder) to anywhere on your computer, then run the EXE labelled “Goldeneye Revenge.exe”

Please ensure you have the latest OpenGL libraries for your computer.

**How to Play**

**Using a keyboard**You can move the player forward by using the **W (or upward arrow key if that setting is set)** key, left using the **A** **(or left arrow key if that setting is set)** key, right using the **D (or right arrow key if that setting is set)** key and backwards using the **S (or downward arrow key if that setting is set)** key. You can control the camera by using the mouse.

To fire the weapon, press the **Left Mouse** button.

To reload, press the **R** key.

To pause the game either press **P** or **ESC.**

Pressing the **1** or **2** key, you can switch between weapons.

**Using a joystick**You can move the play in any direction using the **Left thumb stick** **(or DPAD if that setting is set)** and control the camera using the **Right thumb stick.**

To fire the weapon press the **Right Trigger.**

To reload, press the **X** button.

To pause the game, press the **Start** button.

To cycle through weapons, press the top **Right/Left Triggers**