Project #2: A\* Pathfinding (due week 6).

**Additional Notes:**

1. PathfindingExampleSolution.exe is used as reference. Check your project with example solution.
2. After you change map, tiny may end up standing at the wall. Then when you start A\* search, the program may crash and output error message "Can't set color to a grid square that is a wall". It is OK and you will not lose point. If you find it annoying, you can disable this assert in Terrain.cpp line 131.
3. (Completeness) Be sure to test edge case when goal is not reachable.
4. (Completeness) Be sure to add starting node when you build waypoint list.
5. (Optimal) Octile with weight 1.0 may have strange outcomes because of floating point error. Test it with weight 1.01.
6. (Heuristic Methods) Compare results with example solution to see if they are similar.
7. (Straight Line Optimization) Should not do A\* pathfinding if straight line check succeeds. (it is an optimization, you should not start A\* search if you know the path is just a line)
8. (Straight Line Optimization) Use the method taught in class. Do not use ray casting. (the same goes to rubberbanding and smoothing, use the method taught in class)
9. (Rubberbanding) Make sure it deals with edge case such as “less than 3 nodes in the path”.
10. (Smoothing) Make sure it deals with edge case such as “less than 4 nodes in the path”.
11. (Smoothing) Make sure it handles the case where both rubberbanding and smoothing are on. (also do not disable rubberbanding when smoothing is on)
12. “Toggle Test Long” button is used to check the speed of your pathfinding. Your A\* search code should be called in ComputePath(). If you place it somewhere else, it may temper the result of timing test. You need to fix it for project 3 if it’s the problem. Be careful when you modify code, especially in ComputePathWithTiming(). (in most cases you do not need to modify this function)