**Design Choices**

1. Storyboard.pdf includes the initial design, which models sand. This has later been changed to model the behavior of snow and glacier, simply because I decided that the product I created models glacier and snow more accurately.
2. More detailed description of things I referred to as I created this game is included in TP1.pdf document (also in this directory).
3. Main problems I tried to solve are:
   1. How to model the motion of falling snow, which are broken up into smaller particle-like chunks when falling, and how they stack up. I designed this to be an iterative algorithm that checks every piece of settled snow piece checks its surrounding, and make sure that it’s in a stable state, meaning that it does not stack too high by itself. (For instance, if the heights of 5 consecutive columns are 0, 0, 6, 0, 0, the middle one should be moved and to be evenly distributed in that neighborhood. This is solved by iteratively checking and finding the differences in height.
   2. How to display the fallen snow. In this project, I looked for a way to generate natural, smooth curves to visually represent the glacier. I used best fit lines on intervals to optimize this. In the “sand” mode (which should be the “snow” mode, in the context of this game, but I did not change the variable name), I used around ten intervals across the screen. In the drive mode, I used intervals based on the position of the car to generate a locally smooth solution using those data points.
   3. How to display car in drive mode. In this mode, I found the angle using a local linear approximation to find the current angle of the car with respect to the x axis, and used this to rotate the sky graph. The height of the glacier/hills are also taken with respect to the current absolute height of the vehicle. The height of the car is always at the center, and unless the car is arriving at the end or starting out within an interval, the x position of the car is always at the center of the screen. To draw the hills, I used the trapezoidal approximation and draw polygons.
   4. How to give warnings/heads up in the driver mode about up and down hills. It may be challenging for users to determine the change in slope, especially when the angles are small. Therefore, in the first ¼ of any drive, there would be a reminder on the screen to remind player the potential of having to press up or down button. The threshold of having to press up or down is around 5 degrees, but a notification would be given on screen when there’s a potential need for pressing up or down. This relies on mean value theorem: since we are dealing with a sixth-degree differentiable graph, the change in first derivative must also be smooth; I set this notification to show up when the slope is greater than or equal to one degree. Having this notification showing up implies that there is a possibility that up or down buttons must be pressed in the near future, but is not a definitive indicator (because angle can get smaller as well). But there is no punishment when the angle is between 1-5. This lowers the difficulty level of this game.
4. Main screen design: Users can access a quick instructions manual on this screen, or directly start playing. I used MOUSEMOTION to animate the buttons, so the users can easily see the buttons.
5. Help screen design: Used a gradually changing color scheme. Left and right arrows and the see example button are animated. User can also navigate by using left and right arrows of key press. This is to fit into the theme of iceberg and glacier.
6. Example mode: includes screenshots of three main modes, with a few words of description of the purposes. Can scroll through these screenshots by keypress or clicking left and right arrow.
7. **SNOW MODE:** This is the main chunk of “designing the board.” There will be built-in obstacles determined randomly by a generator. User must choose driverless or drivers before officially beginning this game; this choice cannot be changed as levels go up, but can be reset if one loses. Notice that window captions label what level one is on.
8. **DRIVE MODE:** There’s restart button; It’s a simplistic screen, where the background is an imported image of the sky, and it rotates as the car changes slope. One may click restart during the game. When the car is not automatically driven, one must press up and down, and there are heads ups (explained earlier in this document). Gameover is shown once lost or won.
9. Loserboard and scoreboard: Recap of the scores, with the option to restart or go onto the next level.
10. Notes from user testing:
    1. Brian: Change typos within the project (done), add acceleration to car, continuous press on car (done)
    2. Daniel: Add keypress to switching between screens in the help mode (done)
    3. Mark: Same as Daniel (change adopted)
    4. 4th person that I forgot the name of: Accelerate sand (difficult to be done without changing the program substantially; needs to decrease the list size, so no.)