General:

I used a 2-D array to represent each pixel. Each pixel is represented by an array again which consists of its color tuple and the elevation. Each path is represented as an array containing its color value and its multiplier.

Speeds:

I used multipliers to decide what speed the person is going to move in a different terrain. A higher multiplier means that the person is moving slower, because it helps to easily maximize the heuristic.
Assuming orienteering is a sport where people have to move longer distances, sprinting or running is not really an option for people. At best people might jog, and mostly brisk walk. Thus the difference between the multipliers was kept to be small.

On open land or paved path the multiplier was 1, because those are the places people move the fastest.
The next was a footpath which was 1.1, but during the fall when there are leaves the footpath multiplier is made to be 1.2.
An easy moving forest was given a multiplier of 1.2 because it is pretty similar to a footpath in fall.
After then there are slow run forest, walk forest and rough meadow all respectively at 1.3, 1.4 and 1.5.
Water, Impassable vegetation and out of bounds were all given a multiplier of infinity because you cannot really access those.
During the winter when the water freezes up and turns into ice, you can walk on that at the speed of 1.3.
During the winter, the land near the water bodies is converted to mud and has a multiplier of 1.4

Heuristic:

H(n):

It is the straight line distance from the neighbor to the goal. It is calculated using pythagoras theorem where the pixel size (10.29 * 7.55 m), number of pixels in between and also the elevation between both the points is accounted for and we get a final value as the straight line distance between both the points.

G(n):

This one represents the cost of going from one pixel to the next. In this the straight line is calculated between both the neighboring pixels. This distance is then multiplied by the multiplier of the neighbor as a change in terrain type affects the heuristic and movement speed as well.

F(n):

This function is the final heuristic which is basically the addition of H(n) and G(n). Here both H(n) and G(n) maximizes as it gets harder to reach our goal. The multiplier affects the G(n) to represent the change in terrain. Also, due to the multiplier we are determined to take the faster terrain type when the straight line distance is equal to the goal.
Thus, we go to the pixel with F(n) as the minimum because it gives us the vertex from which it is easiest to reach our goal.

Seasons:

Fall:
For the fall season I iterated through each of the pixels in the map. If the pixel represented a footpath and had easy movement forest pixel as its neighbors, it was replaced to a leafy path and the terrain multiplier was increased.

Winter:
For winter I iterated through the terrain Array and figured out the boundaries of the water bodies. Once that was done, I iterated through this list and did a BFS on a node if it was not already made ice, until depth 7. Thus, the 7 pixels near the land are turned to ice. And a terrain multiplier of 1.3 is applied to ice, making it an accessible path.

Spring:
Basically it's the same BFS function used for Winter, but this time it converts it to mud and only if the elevation difference is less than 1m and goes to a depth of 15 pixels. The multiplier is changed to 1.4 because it is really difficult to walk in mud / water.