```cpp
//
//  Created by Cameron Dietz on 10/8/18.
//
#include <iostream>
#include <vector>
#include <cstdlib>
#include <cmath>
#include <chrono>
#include <fstream>
#include <iomanip>
#include <string>

using namespace std;

int colorPath(const vector<vector<int>>& heightMap, vector<vector<int>>& r,vector<vector<int>>& g, vector<vector<int>>& b, int color_r, int color_g,int color_b, int start_row, int start_col = 0);

int main() {
    //initialize all variables
    int a=0;
    int row;
    int col;
    int minval;
    int maxval;
    int x;
    double val;
    double grey;
    std::string fileName;

    int bestDist = 10000000;
    int dist = 0;
    int bestpath = 0;

    //initialize vectors
    std::vector<int> v;
    std::vector<std::vector<int> > dat;

    //recieve user input
    std::cout << "Enter number of rows: ";
    std::cin >> row;
    std::cout << "Enter number of columns: ";
    std::cin >> col;
    std::cout << "Enter name of file: ";
    std::cin >> fileName;
    std::string outFileName = fileName+".ppm";

    //call the file to recieve data
    std::ifstream ifile(fileName.c_str());
```

```
48
49      //check to see if the file can be opened
50      if(!ifile.is_open())
51      {
52          std::cout << "Error: Could not access file." <<endl;
53          return -1;
54      }
55
56      //creates the output file
57      std::ofstream ofile(outFileName.c_str());
58
59      //checks to see if the output file can be accessed
60      if(!ofile.is_open())
61      {
62          std::cout << "Error: Could not access output file." <<endl;
63          return -1;
64      }
65
66      //find max and min values
67      while(ifile >> x){
68
69          if(a==0){
70              minval=x;
71              maxval=x;
72              a++;
73          }
74          if(x < minval){
75              minval = x;
76          }
77          if(x > maxval){
78              maxval = x;
79          }
80
81          v.push_back(x);           //puts the values into the vector v in
              appopriate number of columns
82          if(v.size() == row)
83          {
84              dat.push_back(v);           //adds that vector to the dat
                  vector
85              v.clear();
86          }
87      }
88
89      //checks to see if the end vector matrix is the apprpriate size
90      if(v.size()!=0&&dat.size()!=col)
91      {
92          std::cout << "Error: Recieved more or less data values than
              expected." <<endl;
93          return -1;
```

```
94      }
95      //initializes vectors within a vector one for each row and main one is set to the number of columns
96      std::vector<std::vector<int> > r(col);
97      std::vector<std::vector<int> > b(col);
98      std::vector<std::vector<int> > g(col);
99
100     //goes through each value in the vector matrix and adds the calculated grey scale rbg value to the vector rbg
101     for(int i=0;i<dat.size();i++){
102         for(int j=0;j<dat.at(i).size();j++){
103             if(minval==maxval){
104                 val=0;
105             }else{
106
107                 val=round(((dat.at(i).at(j) - minval)*255.0)/(maxval - minval));
108             }
109             grey=val;
110             r.at(i).push_back((int)grey);
111             b.at(i).push_back((int)grey);
112             g.at(i).push_back((int)grey);
113         }
114     }
115
116
117     for(int i=0;i<dat.size();i++){
118
119         dist = colorPath(dat,r,g,b,252,25,63,i);
120
121         if(dist<bestDist){
122             bestDist = dist;
123             bestpath = i;
124
125
126         }
127
128     }
129
130     dist = colorPath(dat,r,g,b,31,253,13,bestpath);
131
132     //inputs the first statements into the ppm file and then adds the three of the rbg value from the vector into the file
133     ofile<<"P3"<<endl;
134     ofile<<col<<" "<<row<<endl;
135     ofile<<255<<endl;
136
137     for(int i=0;i<dat.size();i++){
138         for(int j=0;j<dat.at(i).size();j++){
```

```
139             ofile<<r.at(i).at(j)<<" "<<g.at(i).at(j)<<" "<<b.at(i).at(j)
                  <<" ";
140         }
141         ofile <<endl;
142     }
143 }
144
145 // Output distance
146 int colorPath(const vector<vector<int>>& heightMap, vector<vector<int>>&
      r,
147     vector<vector<int>>& g, vector<vector<int>>& b, int color_r, int
        color_g,
148     int color_b, int start_row, int start_col) {
149
150     int j = 0;
151     int i = start_row;
152     int diff0 = 0;
153     int diff1 = 0;
154     int diff2 = 0;
155     int minNum;
156     int distance = 0;
157     r.at(start_row).at(0) = color_r;
158     g.at(start_row).at(0) = color_g;
159     b.at(start_row).at(0) = color_b;
160
161     while (j < heightMap[0].size() - 1) {
162         if (i <= 0) {
163             diff1 = abs(heightMap.at(0).at(j) - heightMap.at(0).at(j +
                  1));
164             diff2 = abs(heightMap.at(0).at(j) - heightMap.at(1).at(j +
                  1));
165             diff0 = max(diff1, diff2) + 1;
166         }
167         else if (i >= heightMap.size() - 1) {
168             // Check column values of row and row - 1
169             diff0 = abs(heightMap.at(i).at(j) - heightMap.at(i - 1).at(j +
                    1));
170             diff1 = abs(heightMap.at(i).at(j) - heightMap.at(i).at(j +
                  1));
171             diff2 = max(diff0, diff1) + 1;
172         }
173         else {
174             diff0 = abs(heightMap.at(i).at(j) - heightMap.at(i - 1).at(j +
                    1));
175             diff1 = abs(heightMap.at(i).at(j) - heightMap.at(i).at(j +
                  1));
176             diff2 = abs(heightMap.at(i).at(j) - heightMap.at(i + 1).at(j +
                    1));
177         }
```

```
178
179         minNum = min(diff0, min(diff1, diff2));
180
181         if ((minNum == diff0) && (diff0 != diff1) && (diff0 != diff2)) {
182             --i;
183         }
184         else if (((minNum == diff2) && (diff1 != diff2)) || (diff0 ==
              diff2 && (diff2 < diff1))) {
185             ++i;
186         }
187
188         distance += minNum;
189         r.at(i).at(j + 1) = color_r;
190         g.at(i).at(j + 1) = color_g;
191         b.at(i).at(j + 1) = color_b;
192         ++j;
193     }
194     return distance;
195 }
196
197
198
199
200
```