**Image Quantization**

Esraa Mohamed Abdelhamid

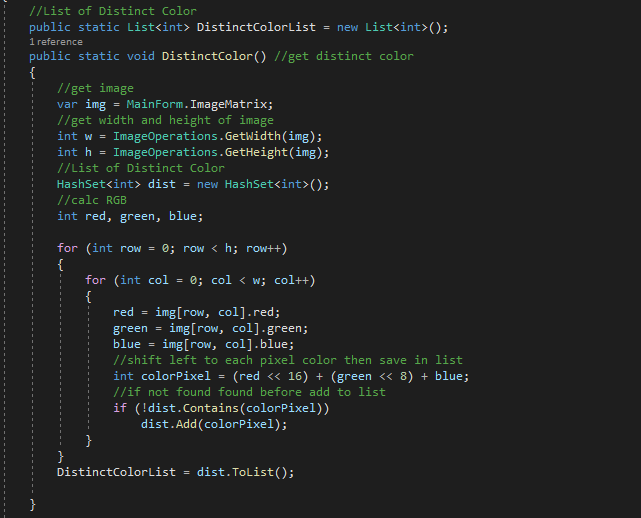
Elham Mohamed

Aya Ebrahim

**Graph construction description and code:**

* **Explication: function DistinctColor ()**

It to get different color from image by loop on image (height, width), get pixel of color from struct then shift left color red + shift left color green + blue than added it to list



**Analysis of the code:** O(N2)

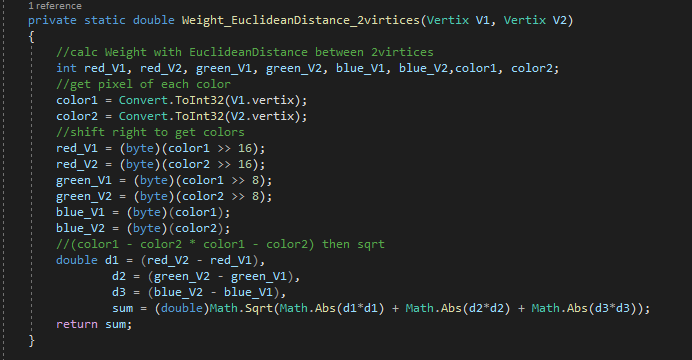
Storage, Add, shift, Statement = O(1)

Loop O(n2) //width\*height

Total = O(1) + O(n2) = O(n2)

* **Explication: function** **Weight\_EuclideanDistance\_2virtices (vertix v1, vertix v2)**

Calculate weight between two vertices by Euclidian Distance Equation **=** (R2-R1)^2 + (G2-G1)^2 + (B2+B1)^2



**Analysis of the code:** O(1)

Statement, Storage, shift O(1)

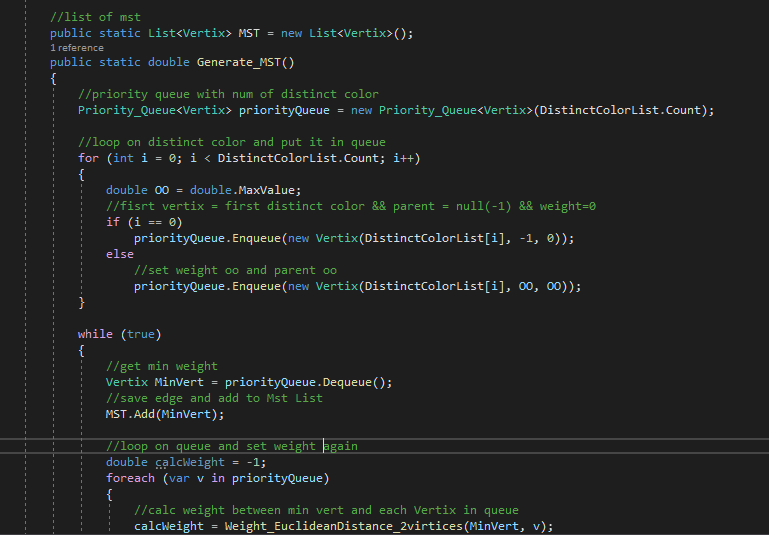
**Minimum spanning tree code.**

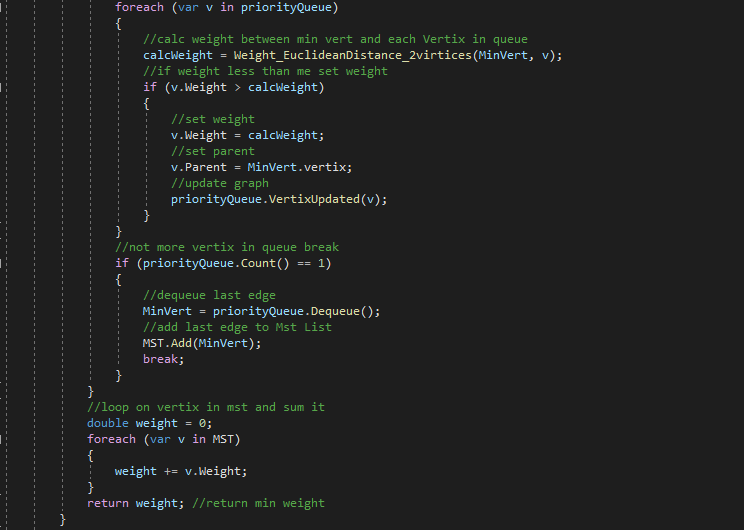
* **Explication: function Generate\_MST ()**

Build tree with list of distinct color, Implemented with Priority Queue.

In first: add all distinct color to queue

Then dequeue vertix one by one ,added to MST List and Loop in all vertix in queue, Calculate min weight by function **Weight\_EuclideanDistance\_2virtices()**

Which take current vertix and one vertix of queue then update vertix with min weight and update parent



**Analysis of the code:** O(E2logV)

Statement = O(1)

Initialize queue O(V) //v=>num of distinct color

Insert, extract O(VlogV)

Update tree O(logV) //e=>num of edges

Loop => E2

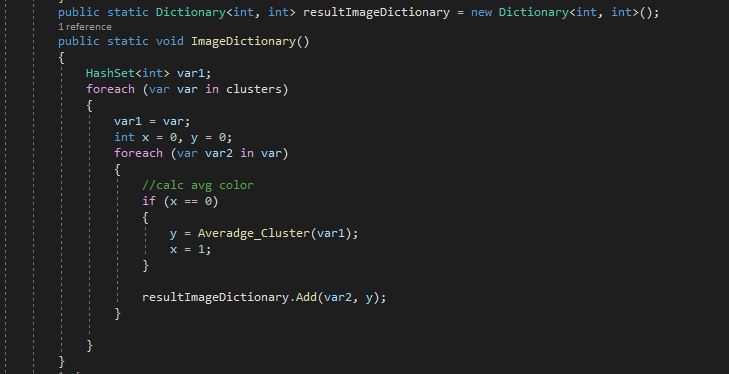
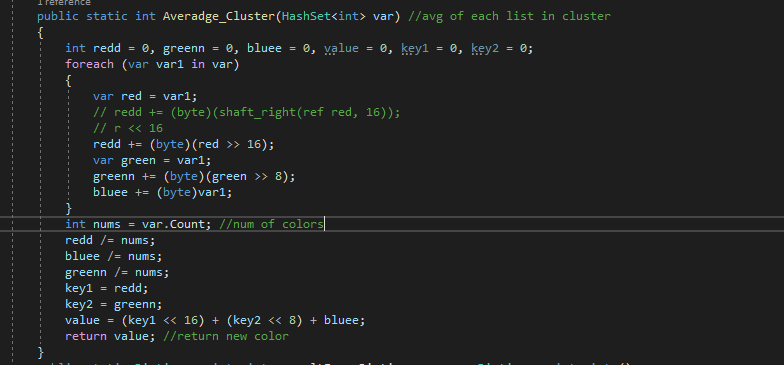
Total = O(E2logV)

**Palette generation code.**

* **Explication: function ImageDictionary()**

Calculate average color(red, green, blue) of each cluster by loop of list of list cluster help it function to calculate new color Averadge\_Cluster(HashSet<int> var) then add to dictionary each distinct color with average of weight cluster colors

Averadge\_Cluster(HashSet<int> var): loop on list, calculate average of color then return new color



**Analysis of the code:** O(k\*V)

Statement = O(1)

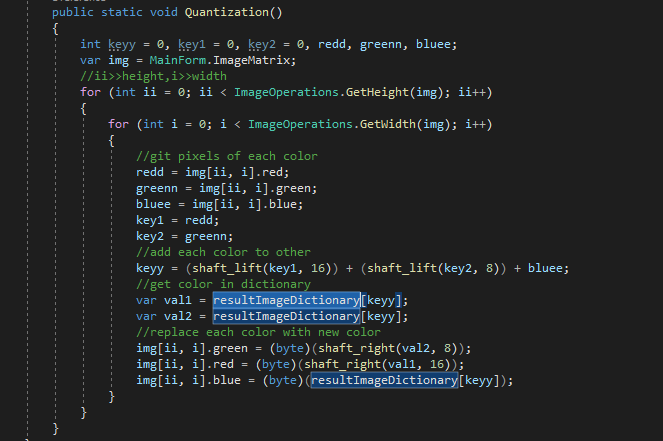
Loop on num of clusters K \*

Loop on vertices (Distinct color) V

Total = O(k\*V) + O(1) = O(k\*V)

**Explication: function Quantization ()**

Replace each color in image with new color which saved in resultImageDictionary



**Analysis of the code:** O(N2)

Storage, Add, shift, Statement =

O(1)

Loop O(n2) //width\*height

Total = O(1) + O(n2) = O(n2)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Image Name** | **# Distinct Colors** | **MST Sum** | **K** | **Execution time** |
| Sample.Case1 | 5 | 730.55 | 3 | 1s |
| Sample.Case2 | 3 | 322.65 | 2 | 1s |
| Sample.Case3 | 2266 | 6106.98 | 500 | 1s |
| Sample.Case4 | 69 | 1120.66 | 10 | 1s |
| Sample.Case5 | 256 | 441.67 | 32 | 1s |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Image Name** | **# Distinct Colors** | **MST Sum** | **K** | * 1. **Execution time** |
| Small.Case1 | 8,708 | 11,785.07 | 192 | 4s 75ms |
| Small.Case2 | 10,265 | 19,888.82 | 2160 | 7s |
| Medium.Case1 | 27,410 | 40,616.4 | 1737 | 42s 71ms |
| Medium.Case2 | 20,041 | 44,831.69 | 2284 | 28s 12ms |
| Large.Case1 | 56,328 | 118,145.09 | 3829 | 3 min 30s 11ms |
| Large.Case2 | 54,223 | 80,957.21 | 25,666 | 4min 19s 18ms |