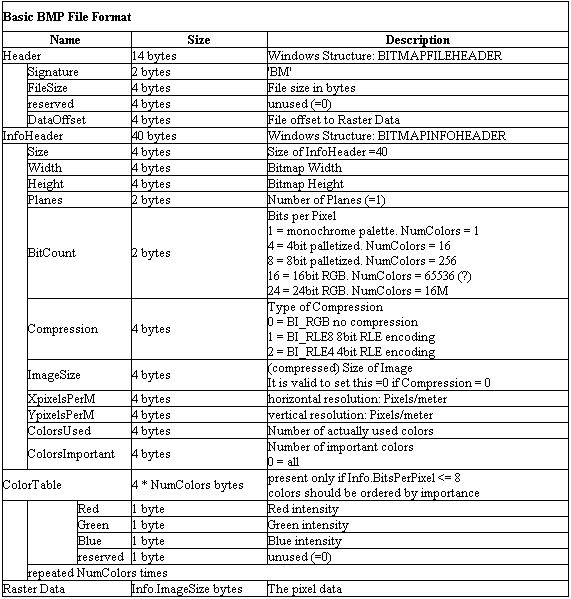
**“BITMAP PROCESSING LIBRARY” RƏHBƏR**

**BITMAP-ın strukturu:**



**BITMAP-ın oxunması:**

BMP faylındakı informasiyanı oxumaq üçün ***BinaryReader*** istifadə edilmişdir. Məlumatlar aşağıdakı byte massivlərinə yazılmışdır:

1. Faylın ilk 14 baytlıq hissəsi FileHeader məlumatlarını təşkil edir. Buraya faylın formatı (Signature), ölçüsü (FileSize) və piksel verilənlərinin başladığı baytın sıra nömrəsi (DataOffset) daxildir. Bu məlumatlar oxunaraq **FileHeader** byte massivinə yazılır. Massivin ilk iki baytının (Signature) saxladığı dəyərin ‘BM’-ya bərabər olub olmadığını yoxlayırıq. Əgər bərabər olmasa faylın BMP olmaması deyə bildiriş qaytarırıq və prosesi dayandırırıq. Əkshalda informasiyanın oxunması davam edir. Byte massivindən bizə lazım olacaq olan DataOffset məlumatlarını götürüb **DataOffset** dəyişənimizə mənimsədirik.
2. Növbəti 40 baytlıq hissə InfoHeader məlumatlarını təşkil edir. Buraya InfoHeader hissəsinin ölçüsü (Size = 40 bytes), şəklin üfüqi (Width) və şaquli (Height) ölçüsü, bir pikselə düşən bit sayı (BitCount) məlumatlar daxildir. (Burada bir sıra başqa parametrlər də mövcuddur, lakin burada layihədə lazım olanlar qeyd olunmuşdur.). Bütün məlumatlar **InfoHeader** byte massivinə yazılır. Oradan lazımlı dəyərlər **Width**,**Height** və **BitCount** dəyişənlərinə yazılır.
3. Növbəti hissə ColorTable yalnız 32 bit BMP-lar üçün mövcuddur. Oradakı informasiyalar layihədə bizə lazım olmadığı üçün oxunulub (**ColorHeader** byte massivinə yazılıb), lakin emal edilməyib.
4. Sonuncu hissə RasterData şəklin piksel məlumatlarını saxlayır. Bu bölmənin ölçüsü olacaq. Ümumilikdə **Height** sayda sətirdən ibarətdir, hər sətirin ölçüsü olacaq. bir pikselin ölçüsüdür. Hər sətirdə sayda piksel olduğu üçün Pikselin ölçüsü piksel sayına vurulur. Hər sətrin ölçüsünün 4-ə tam bölünməli olduğu üçün ifadəyə sətrin ölçüsünü 4-ün vuruğuna tamamlayan toplananı əlavə olunmuşdur.

**BITMAP-ın yazılması:**

BMP faylına informasiyanı yazmaq üçün ***BinaryWriter*** istifadə edilmişdir. İnformasiya saxlayan byte massivləri oxunulduğu ardıcıllıq ilə də fayla yazılır.

**BITMAP üçün yaradılmış əməliyyatlar:**

1. **Konstruktor BMP(path)** - bu funksiya verilmiş ***path*** ünvanında yerləşən bitmap faylının oxumaq üçündür.
2. **Konstruktor BMP(w,h,a)** - bu funksiya verilmiş ***w*** enində və ***h*** hündürlüyündə olan bitmap yaradır. ***a*** isə bool dəyişəni olub bitmapın şəffaqlıq parametrinin olub-olmamasını təyin edir.
3. **SaveAs(path)** - bu funksiya bitmapı verilmiş path ünvanında və ad ilə daimi yaddaşa qeyd edir.
4. **GetSize()** - bu funksiya şəklin en və hündürlük ölçülərini qaytarır.
5. **GetBitCount()** - bu funksiya şəkildə bir pikselə düşən bit sayını qaytarır.
6. **GetPixel(x,y)** - bu funksiya şəkildə (x,y) koordinatlarında yerləşən pikselin rəngini RGB formatda qaytarır.
7. **SetPixel(x,y,c)** - bu funksiya şəkildə (x,y) koordinatlarında yerləşən pikseli c rəngində rəngləyir.
8. **DrawLine(x1,y1,x2,y2,c)** - bu funksiya (x1,y1) nöqtəsindən (x2,y2) nöqtəsinə c rəngində düz xətt çəkir. İş prinsipi düz xəttin düz və tərs funksiyalarına əsaslanır. Piksellərin əmələ gətirdiyi düz xətt diskret olduğu üçün düz və tərs funksiyalara ehtiyac duyuruq. funksiyası (x1,y1) və (x2,y2) nöqtələrindən keçən düz funksiya olsun. Onda funksiyası həmin nöqtələrdən keçən tərs funskiya olacaq. Əgər şərti ödənərsə, min(x1,x2)-dən max(x1,x2)-yə qədər bütün x iterasiyaları üçün düz funksiyada y qiyməti hesablanır və (x,y) koordinatlarında yerləşən piksel rənglənir. Əks halda, min(y1,y2)-dən max(y1,y2)-yə qədər bütün y iterasiyaları üçün düz funksiyada x qiyməti hesablanır və (x,y) koordinatlarında yerləşən piksel rənglənir.
9. **Grayscale()** - bu funksiya şəkli boz çalara çevirmək üçündür. Hər bir pikselin RGB (Red-Green-Blue) rəng dəyərləri olur. Pikseli boz rəngə çevirmək üçün bu üç rəngin hər biri öz əmsallarına (R-0.3, G-0.59, B-0.11) vurulub toplanılır və alınmış boz çalarlı dəyər pikselə yazılır.
10. **Scale(xscale,yscale)** - bu funksiya şəklin enini ***xscale*** mislində, hündürlüyünü isə ***yscale*** mislində miqyaslamaq üçündür. Nəticəyə çıxan şəklin ölçüləri ***Width×xscale*** və ***Height×yscale*** olacaq. Yeni şəklin hər bir (x,y) koordinatlarında yerləşən pikseli orijinal şəkildəki () koordinatlarında yerləşən pikselin rəngini alır.
11. **Rotate(angle)** - bu funksiya şəkli verilmiş ***angle*** dərəcə çevirir. Şəkli çevirmək üçün hər bir pikseli tək-tək müəyyən mərkəz ətrafında çevirmək lazımdır. Şəkilin mərkəzində yerləşən nöqtə (piksel) mərkəzi M nöqtəsi olaraq işarə olunsun. Hər bir P piksel üzərində bu əməliyyat aparılır:  
     1) P-nin koordinatlarından M-in koordinatları çıxılaraq PT nöqtəsi əldə edilir. PTx=Px-Mx və PTy=Py-My.
12. PT nöqtəsi R fırlanma matrisinə vurulur.
13. PT-nin koordinatlarına M-in koordinatları əlavə edilirək P-nin M ətrafında α dərəcə çevrilmiş P` nöqtəsi əldə edilir. P`x=PTx+Mx və P`y=PTy+My.

Nəticə şəkildə P` nöqtəsində yerləşən piksel orijinal şəkildəki P nöqtəsində yerləşən pikselin rəngini alır.

1. **Blur(sigma, kernelsize)** - bu funksiya şəkli bulanıqlaşdırmaq üçündür. ***kernelsize*** ölçülü kvadrat matris yaradılır (bu ölçü tək ədəd olmalıdır). Daha sonra isə matrisin sətir və sütunları -dən -dək nömrələnir. Məsələn ***kernelsize***=5 olarsa matrisin sətir və sütunları -2-dən 2-yə qədər nömrələnəcək. Matrisin hər bir elementinin qiymətini hesablamaq üçün onun sütun nömrəsini (x) və sətir nömrəsini (y) və verilmiş ***sigma*** dəyərini aşağıdakı düsturda yerinə (σ) qoyuruq və matrisin elementinə düsturdan aldığımız qiyməti veririk.

Məsələn, ***kernelsize***=2 və ***sigma***=1 olarsa, matris aşağıdakı kimi olacaq:

Hər bir element matrisin ən kiçik qiymətinə bölünüb ən yaxın tam ədədə qədər yuvarlaqlaşdırılsa, matris aşağıdakı kimi olar:

Daha sonra isə şəkildəki hər bir piksel götürülür və pikselin özünü ortada qəbul edərək ətrafındakı piksellərdən ibarət ***kernelsize*** ölçülü kvadrat götürülür. Hər pikselin rəng dəyəri K matrisindəki uyğun dəyərə vurulur və alınmış hasillər toplanaraq ortadakı seçilmiş pikselə yazılır.

1. **DetectEdges(sigma,kernelsize,discrete)** - bu funksiya şəkildə kənar xətləri təyin etmək üçün nəzərdə tutulmuşdur. İş prinsipi Canny metoduna əsaslanır. Əvvəlcə şəkil boz çalara çevrilir və bulanıqlaşdırılır. Hər bir piksel ortada qəbul edilərək ətrafındakı 8 piksel ilə 3x3 pəncərələr götürülür və uyğun qiymətlər aşağıdakı Sobel operatorların uyğun qiymətlərinə vurulur.

Birinci operatora vurulmadan alınan hasillərin cəmi Px, ikinci operatora vurulmadan alınan hasillərin cəmi isə Py olsun. Onda P dəyəri hesablanır və pikselə bu dəyər verilir.

Bu P dəyəri nə qədər parlaq olarsa, yəni qiyməti daha yüksək olarsa, bir o qədər kənar xəttə aid olmağa namizəddir. Kənar xətləri diskret vəziyyətə də gətirmək olar, yəni bir piksel ağ olarsa, deməli, bu nöqtə hansısa kənar xətti təşkil edir, qara olanda isə heç bir kənar xəttə aid olmur. Bunun üçün P-nin [0,255] aralığında diskret qiyməti hesablanır: