نام و نام خانوادگی: حسنا اویارحسینی	گزارش تمرین سوم درس سیستم های چندرسانه ای
شماره دانشجویی: ۹۸۲۳۰۱۰	تاریخ: خرداد - ۱۴۰۲

" تمرین سوم - بخش عملی"

بخش اول)

PNG:

- فرمت فایل:

فرمت فایل PNG با یک توالی ثابت از بایت ها به نام امضای PNG شروع می شود. این بایت از هشت بایت زیر تشکیل شده است:
۸۹ ۵۰ ٤e ٤۷ ۰d ۰a ۱a ۰a
. این بایت ها به عنوان یک شناسه منحصر به فرد برای فایل های PNG عمل می کنند.

- ساختار تکه ای:

محتوای یک تصویر PNG به چند تکه سازماندهی شده است که هر کدام هدف خاصی را دنبال می کنند. هر قطعه دارای ساختار زیر است:

- Length: یک فیلد چهار بایتی که طول داده های قطعه را بر حسب بایت نشان می دهد.
- نوع قطعه: یک فیلد چهار بایتی که نوع قطعه را مشخص می کند (به عنوان مثال، IHDR، PLTE، IDAT، IEND).
- داده های تکه ای: داده های واقعی قطعه که طول آن با فیلد Length تعریف می شود.
- CRC (Cyclic Redundancy Check): یک فیلد چهار بایتی که برای بررسی خطا و اطمینان از یکپارچگی داده ها استفاده می شود.

- فراداده و فرمت محتوا:

تکه های کلیدی در یک تصویر PNG که فراداده را ارائه می کند و قالب محتوا را تعریف می کند عبارتند از:

- قطعه IHDR (Image Header): این قطعه در ابتدای تصویر ظاهر می شود و حاوی اطلاعات ضروری در مورد تصویر مانند ابعاد، عمق بیت، نوع رنگ، روش فشرده سازی و روش فیلتر است.
- PLTE (Palette) Chunk: این قطعه فقط برای تصاویر رنگی نمایه شده وجود دارد و پالت رنگ استفاده شده در تصویر را مشخص می کند. این شامل لیستی از مقادیر RGB است که رنگ های موجود را نشان می دهد.
- تکه های IDAT (Image Data): این تکه ها داده های تصویر فشرده شده را ذخیره می کنند. داده های تصویر به چند تکه IDAT تقسیم می شوند تا رندر پیش رونده را تسهیل کنند. الگوریتم فشرده سازی zlib (deflate) معمولا برای فشرده سازی داده های تصویر استفاده می شود.
- IEND (پایان تصویر) قطعه: این قطعه پایان داده های تصویر PNG را نشان می دهد.

- تکه های جانبی:

تصاویر PNG همچنین می‌توانند حاوی تکه‌های جانبی اضافی باشند که فراداده یا ویژگی‌های اختیاری را ارائه می‌دهند. برخی از قطعات جانبی که معمولاً مورد استفاده قرار می گیرند عبارتند از:

- TEXt (Text) Thunk: این قطعه اطلاعات متنی مرتبط با تصویر، مانند نظرات، کلمات کلیدی یا اعلامیه های حق چاپ را ذخیره می کند.
- tIME (Time) Chank: این قطعه آخرین زمان تغییر تصویر را ثبت می کند.
- pHYs (Physical Dimensions) Chunk: این قطعه اندازه فیزیکی تصویر را بر حسب پیکسل در واحد طول مشخص می کند.
- قطعه tRNS (Transparency): این قطعه اطلاعات شفاف و کانال آلفا را برای تصویر تعریف می کند.
- سایر تکه‌های جانبی: تکه‌های مختلف دیگری برای مقاصد مختلف مانند ذخیره‌سازی پروفایل‌های ICC، داده‌های تصحیح گاما و موارد دیگر در دسترس هستند.

محتوای فایل:

Start offset	Raw bytes	Chunk outside	Chunk inside
۰	۸۹ ۵۰ ٤e ٤۷ ۰d ۰a ۱a ۰a	Special: File signature Length: 8 bytes	“ PNG”
۸	۰۰ ۰۰ ۰۰ ۰d ٤۹ ٤۸ ٤٤ ٥۲ ۰۰ ۰۰ ۰۱ ۲c ۰۰ ۰۰ ۰۰ e۰ ۰۸ ۰۰ ۰۰ ۰۰ ۰۰ ۹c ٤b ۰٤ d2	Data length: 13 bytes Type: IHDR Name: Image header Critical (0) Public (0) Reserved (0) Unsafe to copy (0) CRC-32: 9C4B04D2	Width: 300 pixels Height: 224 pixels Bit depth: 8 bits per channel Color type: Grayscale (0) Compression method: DEFLATE (0) Filter method: Adaptive (0) Interlace method: None (0)
۳۳	۰۰ ۰۰ ۳e dd ٤۹ ٤٤ ٤۱ ٥٤ ۷۸ ۹c dd bd ٥۹ ۹۷ ٦٤ ٤۹ ۷۲ ۱e f6 7d 66 ee 37 96 cc ac b5 97 e9 e9 ۱۹ ۷٤ ٤f f۷ ۰c ۳۰ fb 90 20 08 12 a2 a8 03 f2 f0 f0 f0 e8 41 47 47 e7 48 ٦f d۲ ۸۳ fe ۹۹ ۱e f4 2a bd 50 12 08	Data length: 16 093 bytes Type: IDAT Name: Image data Critical (0) Public (0) Reserved (0)	

Start offset	Raw bytes	Chunk outside	Chunk inside
	١٢ dc ٠٤ ٨٢ ... ٢c ٢٥ a٢ ٤٠ e٧ ٦d ٣٦ ٩٤ ٠b fd ٩٨ de d٩ d4 ff e9 f5 f3 93 e4 ff 03 70 c3 9e ٤a ١٦ ٠٦ ٢٢ bd	Unsafe to copy (0) CRC-32: 160622BD	
١٦ ١٣٨	٠٠ ٠٠ ٠٠ ٠٠ ٤٩ ٤٥ ٤e ٤٤ ae ٤٢ ٦٠ ٨٢	Data length: 0 bytes Type: IEND Name: Image trailer Critical (0) Public (0) Reserved (0) Unsafe to copy (0) CRC-32: AE426082	

JPG:

- فرمت فایل:

فرمت فایل JPEG با یک توالی ثابت از بایت ها به نام "امضای فایل" یا "شماره جادویی" شروع می شود. در مورد JPEG، با مقادیر هگزادسیمال FF D٨ نشان داده می شود. این امضا نشان دهنده شروع یک فایل JPEG است.

- بخش ها:

پس از امضای فایل، یک تصویر JPEG از یک سری بخش تشکیل شده است. هر بخش با یک نشانگر شروع می شود که یک کد دو بایتی است که نوع و هدف قطعه را مشخص می کند. برخی از نشانگرهای رایج عبارتند از:

- شروع تصویر (SOI): شروع تصویر را مشخص می کند.
- Start of Frame (SOF): حاوی اطلاعاتی در مورد ابعاد تصویر، فضای رنگی و سایر پارامترها است.
- جداول هافمن (DHT): رمزگذاری هافمن مورد استفاده برای کدگذاری آنتروپی را تعریف می کند.
- جداول کوانتیزاسیون (DQT): شامل جداول کوانتیزاسیون مورد استفاده در فرآیند فشرده سازی است.
- Start of Scan (SOS): شروع داده های واقعی تصویر را نشان می دهد.

- فراداده:

در بخش‌ها، تصاویر JPEG می‌توانند حاوی ابرداده‌های مختلف و اطلاعات اضافی باشند. برخی از بخش‌های فراداده رایج عبارتند از:

- Comment (COM): به کاربران اجازه می دهد نظرات یا توضیحات را در فایل تصویر جاسازی کنند.

- فرمت فایل تصویر قابل تعویض (EXIF): حاوی متادیتا مانند تنظیمات دوربین، تاریخ و زمان عکسبرداری و سایر جزئیات ضبط شده توسط دوربین است.

- داده های تصویر:

بخش عمده ای از محتوای باینری در یک تصویر JPEG به داده های تصویر فشرده اختصاص داده شده است. این داده ها با استفاده از تبدیل های رنگی مانند YCbCr به اجزای رنگی (به عنوان مثال قرمز، سبز، آبی) تقسیم می شوند. داده های تصویر فشرده معمولاً با استفاده از تبدیل کسینوس گسسته (DCT) کدگذاری می شوند و سپس با استفاده از کدگذاری هافمن فشرده می شوند.

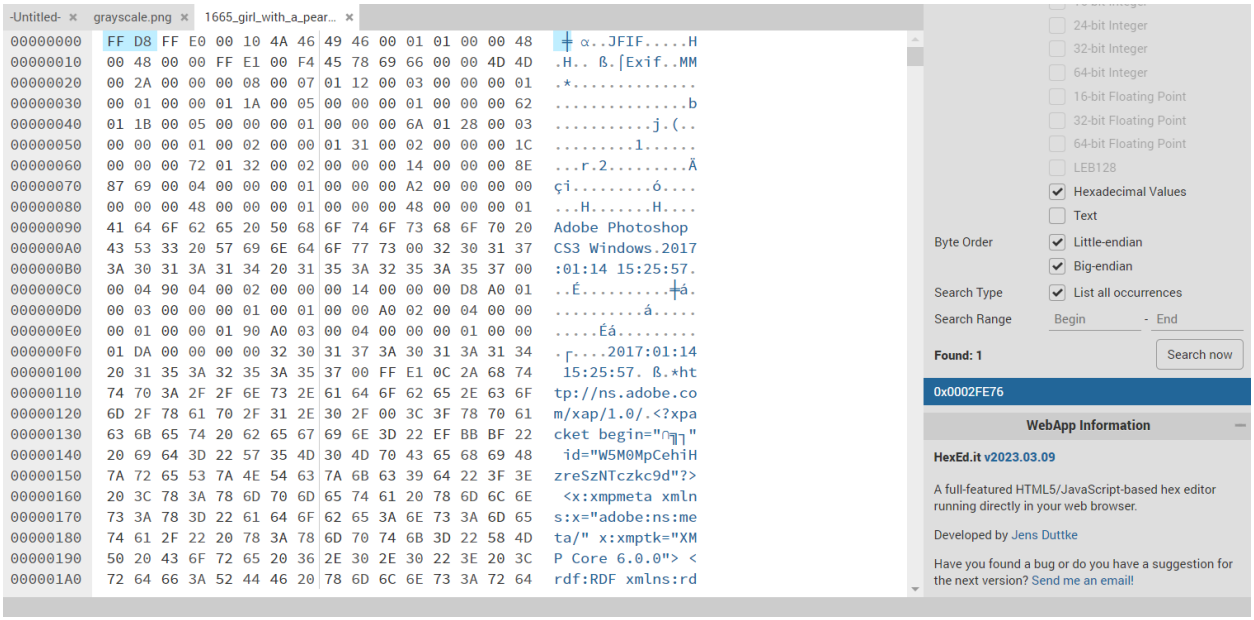
- انتهای تصویر (EOI):

فایل تصویری JPEG با نشانگر پایان تصویر (EOI) به پایان می رسد که با مقادیر هگزادسیمال ۹FF D نشان داده می شود. پایان فایل JPEG را نشان می دهد.

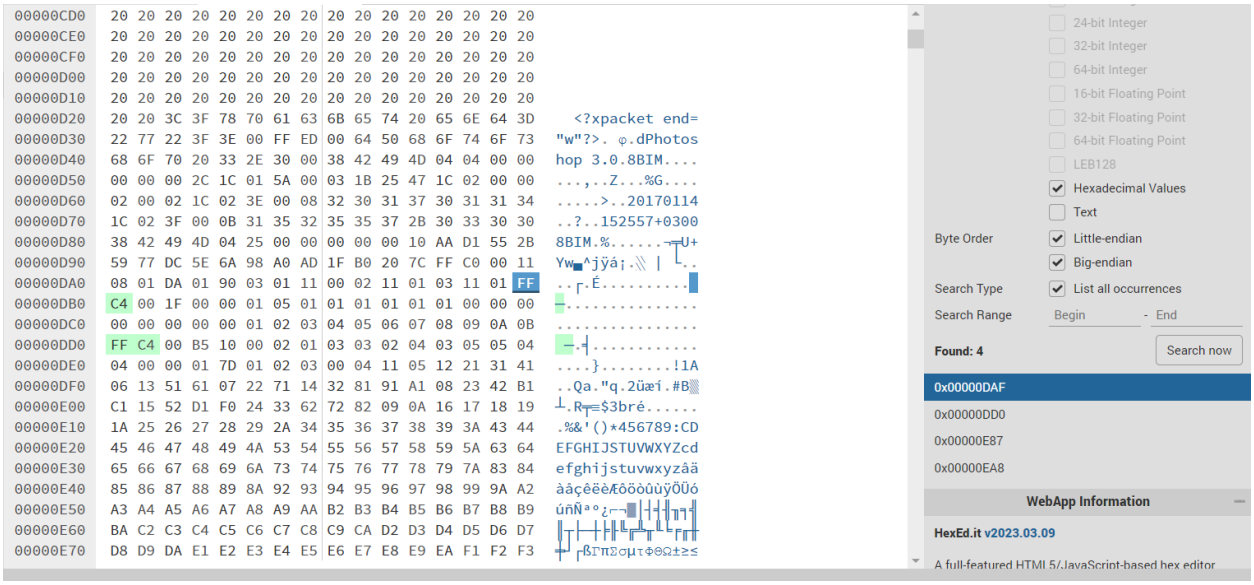
Short Name	Bytes	Payload	Name
SOI	0xFF, 0xD8	none	Start of Image
S0F0	0xFF, 0xC0	variable size	Start of Frame
S0F2	0xFF, 0xC2	variable size	Start fo Frame
DHT	0xFF, 0xC4	variable size	Define Huffman Tables
DQT	0xFF, 0xDB	variable size	Define Quantization Table(s)
DRI	0xFF, 0xDD	4 bytes	Define Restart Interval
SOS	0xFF, 0xDA	variable size	Start Of Scan
RSTn	0xFF, 0xD//n//(//n//#0..7)	none	Restart
APPn	0xFF, 0xE//n//	variable size	Application specific
COM	0xFF, 0xFE	variable size	Comment
EOI	0xFF, 0xD9	none	End Of Image

محتوای فایل:

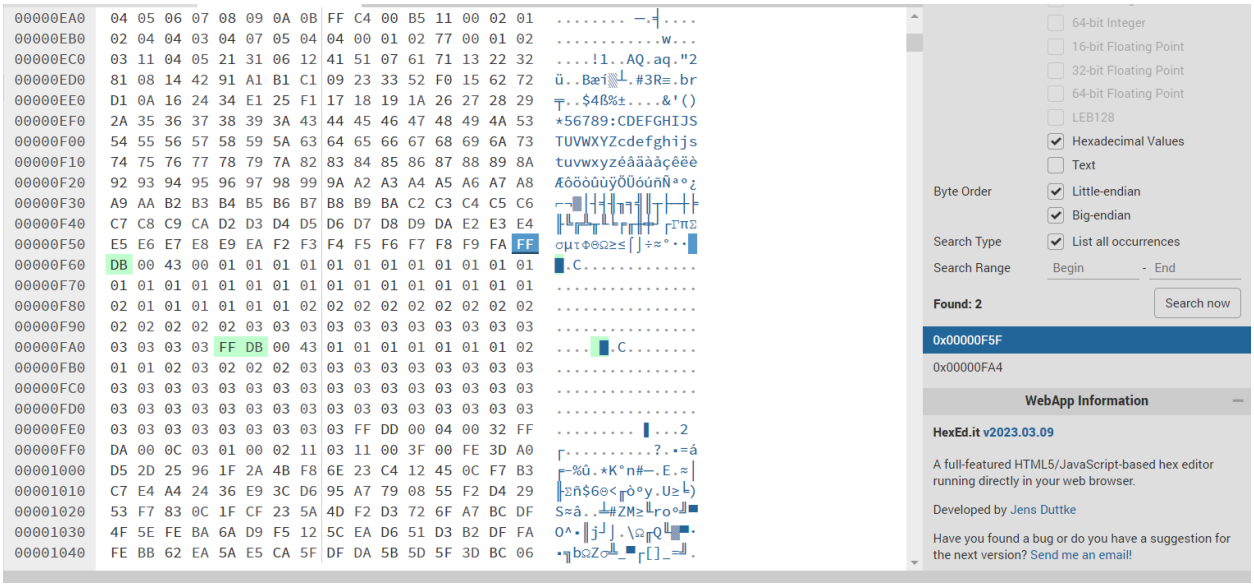
Start of file:



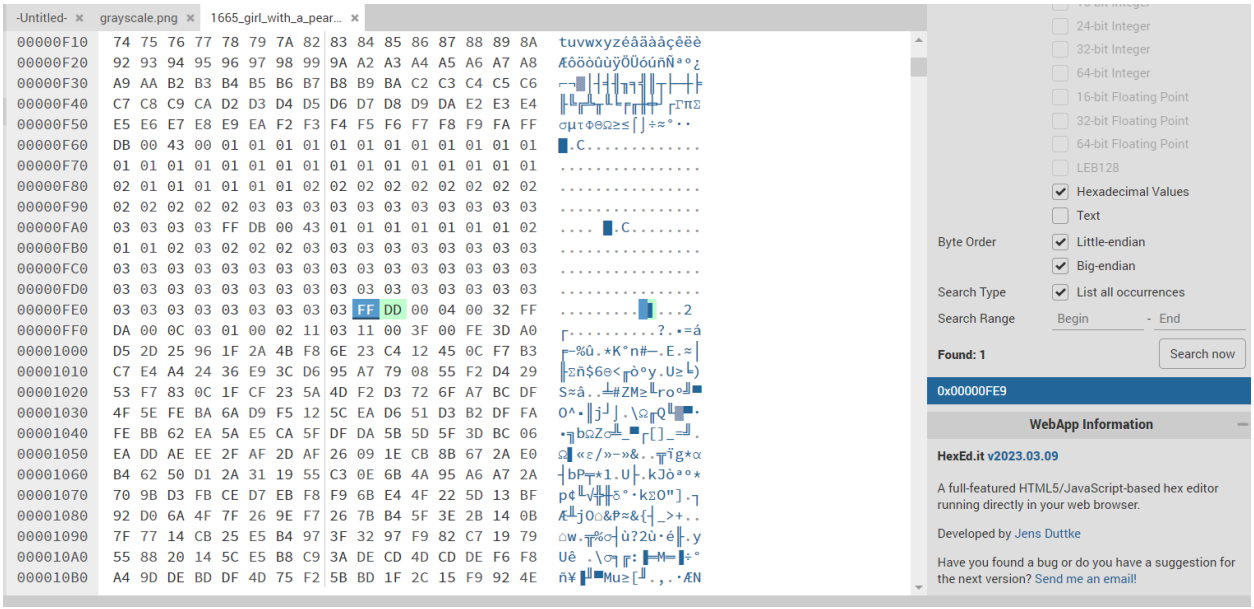
Define Huffman table:



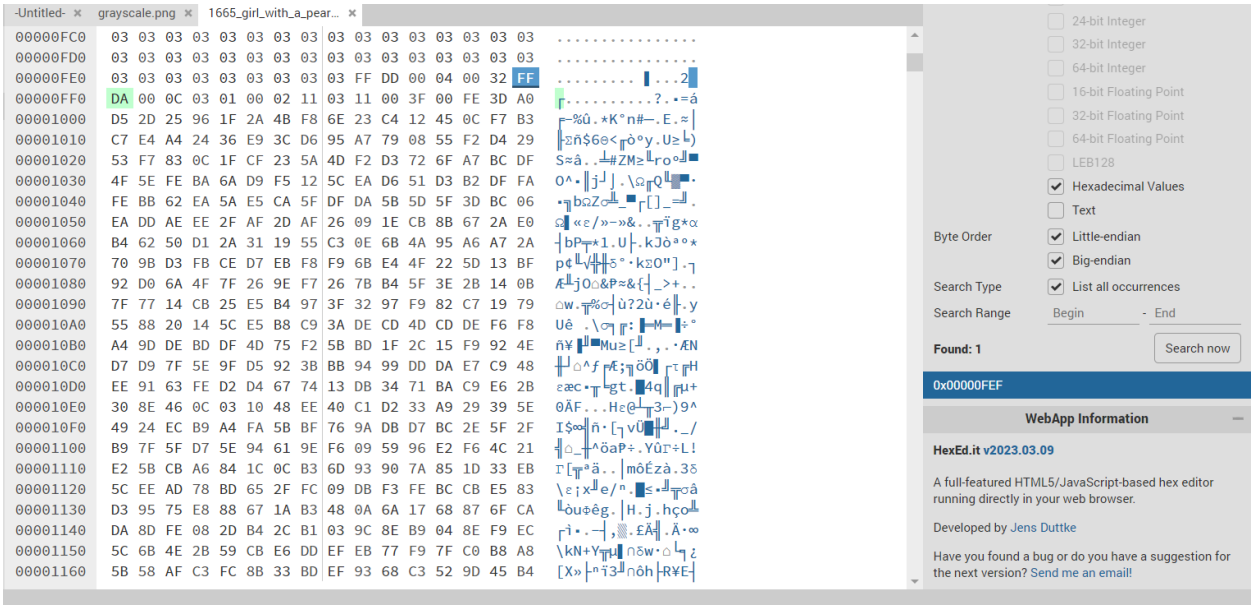
Define quantization table:



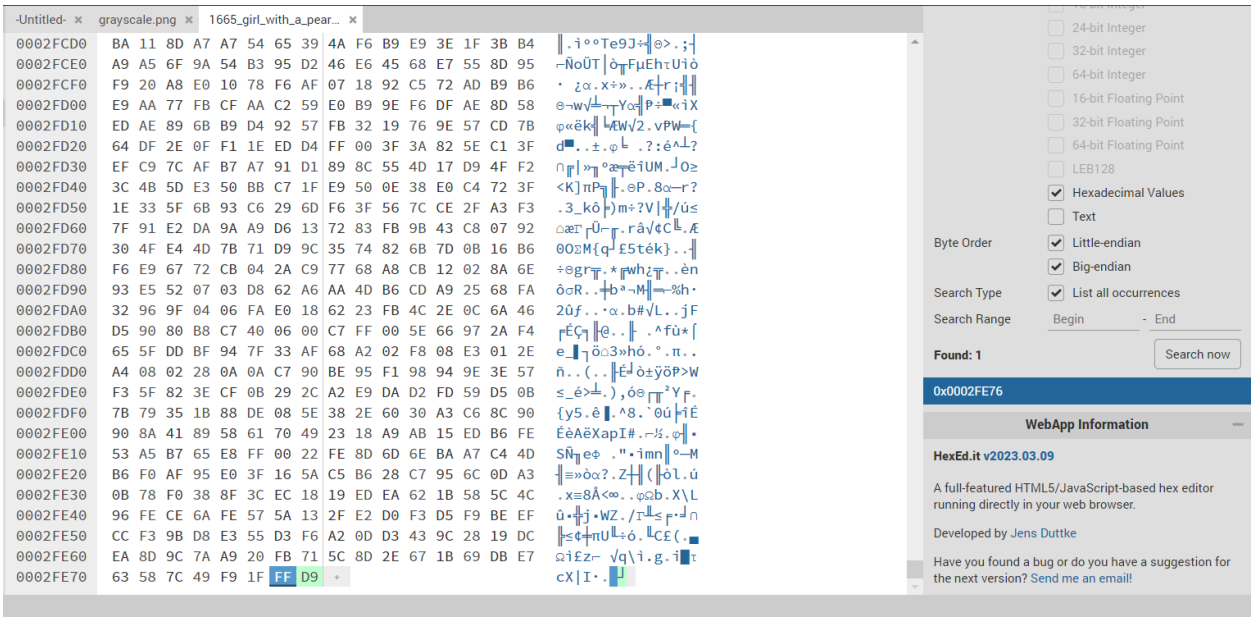
Define restart analysis:



Strat of scan:



EOF:



TIFF:

- فرمت فایل:

فرمت فایل TIFF با یک توالی ثابت از بایت ها به نام "ترتیب بایت" یا "Edianness" شروع می شود. این دنباله مشخص می کند که آیا فایل از ترتیب بایت کم اندین یا بزرگ اندین استفاده می کند. در ابتدای فایل با "II" (برای کوچک-اندیان) یا "MM" (برای بزرگ-اندیان) نمایش داده می شود.

- سربرگ TIFF:

پس از ترتیب بایت ها، یک تصویر TIFF حاوی یک هدر است که اطلاعات مربوط به فایل تصویر را ارائه می دهد. این سربرگ شامل:

- امضای TIFF: یک مقدار دو بایتی (II یا ۴۲ برای small-endian، D۴ برای big-endian) که فرمت فایل TIFF را نشان می دهد و همچنین magic number مربوط به TIFF که ۰۰ ۲A ۴۹ ۴۹ می باشد.
- Offset to the First IFD: یک افست چهار بایت که به اولین فهرست فایل تصویری (IFD) اشاره می کند. IFD حاوی فراداده و اطلاعات مربوط به تصویر است.
- فهرست فایل تصویری (IFD):IFD یک بخش ساختار یافته است که متادیتا و اشاره گر به داده های تصویر را در خود نگه می دارد. این شامل یک سری رکورد به نام "ورودی دایرکتوری" یا "برچسب" است. هر ورودی دایرکتوری شامل موارد زیر است:
  - شناسه برچسب: یک مقدار دو بایتی که نوع اطلاعات ذخیره شده در ورودی را نشان می دهد (به عنوان مثال، عرض تصویر، ارتفاع تصویر، روش فشرده سازی).
  - نوع داده: یک مقدار دو بایتی که نوع داده مقدار تگ را مشخص می کند (به عنوان مثال، ASCII، بایت، کوتاه، طولانی).
  - تعداد داده ها: یک مقدار چهار بایتی که تعداد مقادیر موجود در فیلد مقدار تگ را نشان می دهد.
  - Data Value: داده های واقعی مرتبط با تگ.

  IFD ممکن است حاوی تگ های مختلفی باشد که ابرداده هایی مانند ابعاد تصویر، فضای رنگ، روش فشرده سازی، وضوح و غیره را ارائه می دهد.

Each 12-byte IFD Entry is in the following format:

Bytes	Description
0-1	The Tag that identifies the field
2-3	The field type
4-7	Count of the indicated type
8-11	The Value Offset, the file offset (in bytes) of the Value for the field.The Value is expected to begin on a word boundary; the correspond-ing Value Offset will thus be an even number. This file offset may point anywhere in the file, even after the image data

- داده های تصویر:

پس از IFD، یک تصویر TIFF داده های تصویر واقعی را ذخیره می کند. قالب و سازماندهی داده های تصویر در یک فایل TIFF بسته به روش فشرده سازی و تعداد کانال های رنگی می تواند متفاوت باشد.

- داده‌های تصویر فشرده‌نشده: اگر تصویر فشرده نشده باشد، داده‌های تصویر از IFD در قالب شطرنجی ساده پیروی می‌کنند، جایی که مقادیر پیکسل‌ها به‌طور متوالی ذخیره می‌شوند.

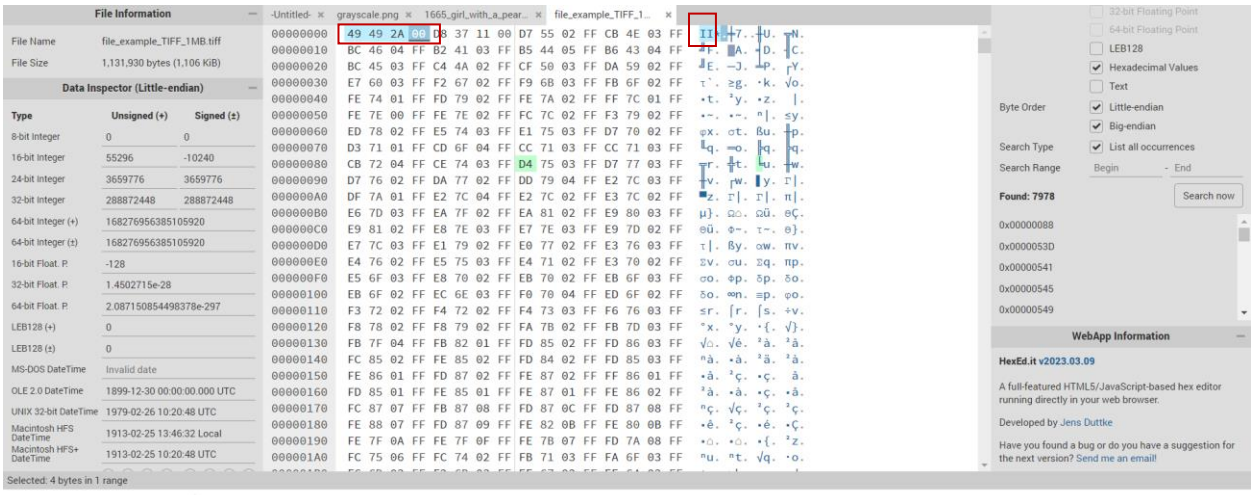
- داده های تصویر فشرده: اگر تصویر فشرده شده باشد، داده های تصویر ممکن است با استفاده از الگوریتم های فشرده سازی مانند LZW، JPEG یا ZIP کدگذاری شوند. در این مورد، داده های تصویر فشرده به طور جداگانه ذخیره می شوند و IFD حاوی اطلاعاتی در مورد روش فشرده سازی و هر پارامتر فشرده سازی مرتبط است.

- IFD های اضافی:

یک تصویر TIFF همچنین می‌تواند شامل چندین IFD باشد، که اجازه می‌دهد چندین تصویر یا چندین نسخه از یک تصویر را در یک فایل واحد ایجاد کنید. هر IFD یک تصویر جداگانه را نشان می دهد و مجموعه ای از ورودی های فهرست و داده های تصویری خاص خود را دارد.

محتوای فایل:

TIFF, little endian



اطلاعات دیگری راجب تصویر که از روی هدر باینری به کمک سایت go.com/۲https://www.metadata بدست آمد:

file_name
file_example_TIFF_1MB.tiff
file_size
1132 kB
file_type
TIFF
file_type_extension
tif
mime_type
image/tiff
exif_byte_order
Little-endian (Intel, II)
image_width
650
image_height
434

bits_per_sample
8 8 8 8
compression
Uncompressed
photometric_interpretation
RGB
document_name
/home/marta/Desktop/www/file ex/files/grafa/tiff/file_example_TIFF_1MB.tiff
strip_offsets
(Binary data 43 bytes)
orientation
Horizontal (normal)
samples_per_pixel
4
rows_per_strip
64
strip_byte_counts
(Binary data 48 bytes)
x_resolution
72.00900354
y_resolution
72.00900354
image_size
650x434
megapixels
0.282
category
image
raw_header
49 49 2A 00 D8 37 11 00 D7 55 02 FF CB 4E 03 FF BC 46 04 FF B2 41 03 FF B5 44 05 FF B6 43 04 FF BC 45 03 FF C4 4A 02 FF CF 50 03 FF DA 59 02 FF E7 60 03 FF F2 67 02 FF F9 6B 03 FF FB 6F 02 FF FE 74 01 FF FD 79 02 FF FE 7A 02 FF FF 7C 01 FF FE 7E 00 FF FE 7E 02 FF FC 7C 02 FF F3 79 02 FF ED 78 02 FF E5 74 03 FF E1 75 03 FF D7 70 02 FF D3 71 01 FF CD 6F 04 FF CC 71 03 FF CC 71 03 FF

بخش دوم)

توضیح کد)

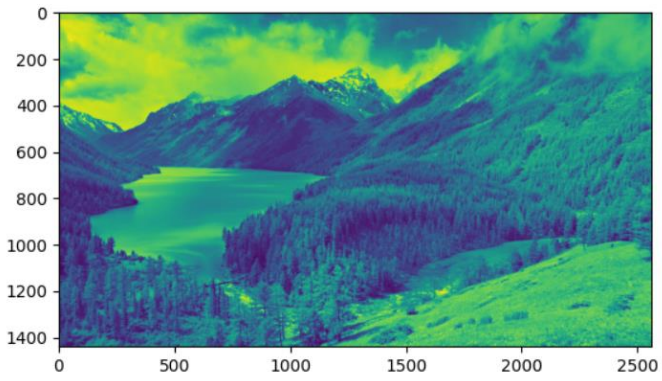
ابتدا عکس ورودی را با تفکیک ۳ کانال رنگی و ترکیب آن ها به تصویر gray scale تبدیل میکنیم:

```
[2]: img = Image.open('image.png')

[6]: def gray_scale(img):
         rgb = np.array(img)
         r, g, b = rgb[:,:,0], rgb[:,:,1], rgb[:,:,2]
         gray_img = 0.2989 * r + 0.5870 * g + 0.1140 * b
         gray_img = gray_img.astype(int)
         return gray_img

[7]: gray_img = gray_scale(img)
     plt.imshow(gray_img)

[7]: <matplotlib.image.AxesImage at 0x25de6e07d50>
```



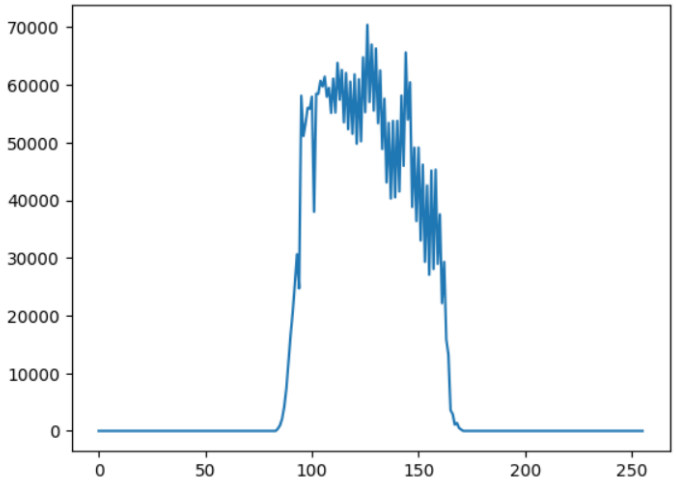
❋ چون به عنوان یک عکس RGB نمایشش داده ایم رنگی شده است.

سپس هیستوگرام رنگ های تصویر را به کمک تابع buncount رسم میکنیم:

```
[8]: def create_histogram(array):
         occurrence_array = np.bincount(array.flatten(), minlength=256)
         return occurrence_array
     histogram_array = create_histogram(gray_img)
```

```
[17]: plt.plot(histogram_array)
```

```
[17]: [<matplotlib.lines.Line2D at 0x25de6e3a4d0>]
```



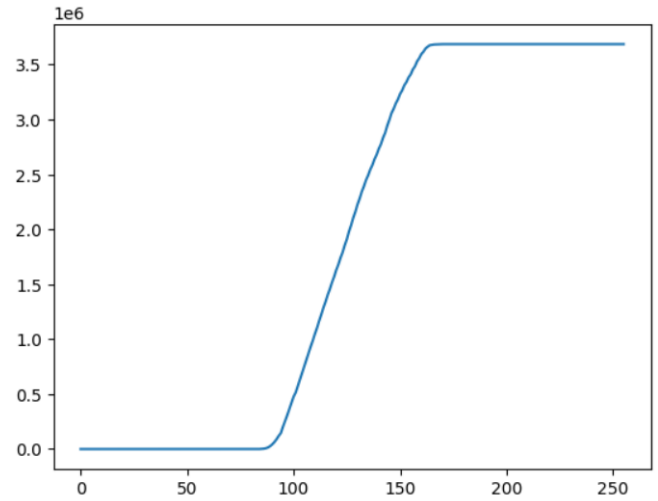
و به کمک تابع cumsum جمع تجمعی را نیز به صورت زیربدست می آوریم:

```
[10]: def count_cumulative_array(array):
          c_array = np.cumsum(array)
          return c_array
```

```
[11]: cumulative_array = count_cumulative_array(histogram_array)
```

```
[12]: plt.plot(cumulative_array)
```

```
[12]: [<matplotlib.lines.Line2D at 0x25de7e41c90>]
```



سپس با توجه به فرمول زیر که در دستور کار آمده نگاشت را ایجاد میکنیم:

T(c) = round((#color_levels - 1) * cumulative_sum(c) / (image_height * image_width))

در این مثال هر پیکسل میتواند رنگی بین ۰ تا ۵ داشته باشد بنابراین تعداد رنگهای مجاز ۶ خواهد بود #levels_color

c جمع تجمعی برای رنگ: sum_cumulative(c)

ارتفاع و عرض تصویر: width_image ,height_image

[13]:
```
def create_mapper(img, color_levels, cumulative_array):
    width = gray_img.shape[1]
    height = gray_img.shape[0]
    color_levels = 256
    transform_map = np.round((color_levels - 1) * cumulative_array / (height * width))
    transformed_image = [transform_map[pixel] for pixel in list(gray_img.flatten())]
    transformed_image = np.reshape(np.asarray(transformed_image), gray_img.shape)
    return transformed_image
```

[20]:
```
output_image_array = create_mapper(gray_img, 256, cumulative_array)
output_image_array = np.uint8(output_image_array)
output_image_array
```

[20]:
```
array([[234, 234, 234, ..., 185, 185, 185],
       [239, 239, 236, ..., 185, 185, 189],
       [239, 239, 239, ..., 185, 185, 189],
       ...,
       [ 14,   8,  25, ..., 183, 189, 189],
       [ 14,   8,  21, ..., 185, 183, 185],
       [ 33,  18,  29, ..., 185, 189, 199]], dtype=uint8)
```

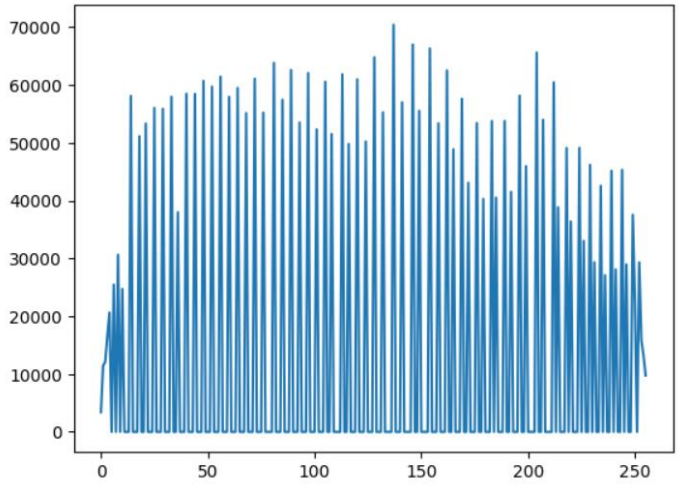
پس از ساختن نگاشت در ۴ خط اول تابع تصویر را با توجه به map تولید شده بازسازی و reshape میکنیم و آن را نمایش میدهیم. خروجی به صورت زیر خواهد بود:

مجدد هیستوگرام رنگ و نمودار جمع تجمعی را رسم میکنیم تا با حالت اولیه مقایسه کنیم:

```
[23]: histogram_array = create_histogram(output_image_array)
```

```
[24]: plt.plot(histogram_array)
```

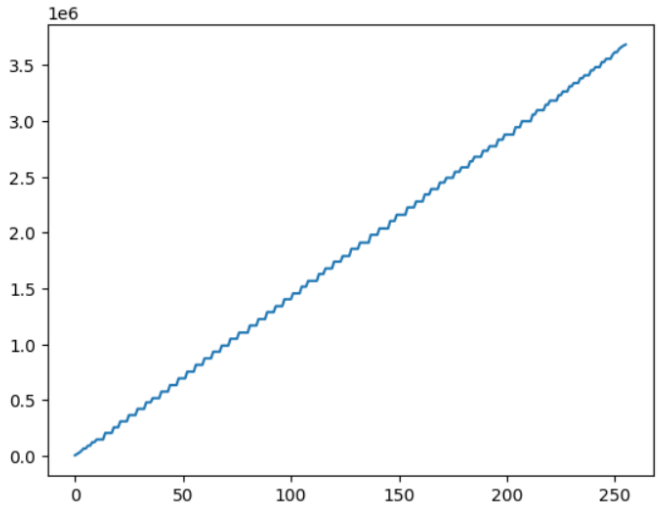
```
[24]: [<matplotlib.lines.Line2D at 0x25d8b565250>]
```




```
[25]: cumulative_array = count_cumulative_array(histogram_array)
```

```
[26]: plt.plot(cumulative_array)
```

```
[26]: [<matplotlib.lines.Line2D at 0x25d8f3fbd90>]
```



مشاهده میکنیم که هیستوگرام در محور x بازتر شده است در تصویر اصلی پراکندگی عمدتا در رنگ های مرکزی است ولی در تصویر نهایی، رنگها پراکنده تر هستند. نمودار جمع تجمعی به یک خط نزدیک تر شده است درحالیکه قبل از آن تا بازه مشخصی نزدیک صفر بود و بعد از بازهای خطی شدن، به اشباع میرسید. یعنی عمل متعادلسازی سبب میشود رخداد رنگهای مختلف برابر شود.