تمرین ۳

۱-یک تصویر را لود کنید . سپس در ۲ حلقه تودرتو تمام مقادیر را بخوانید و بیت اول هر پیکسل را صفر کنید . مقدار جدید را در یک تصویر جدید ذخیره کنید . تصویر جدید و تصویر اولیه را در یک تصویر یکی کنید و نمایش دهید . ماسک را به صورت زیر تعریف کنید :

```
mask = int('11111110', 2)
```

برای صفر کردن بیت اول هر پیکسل کافی است که مقدار پیکسل یعنی img[I,j] را با ماسک بالا bitwise and کنیم

```
for i in range(img.shape[0]) :
    for j in range(img.shape[1]):
        new_value = img[i, j] & mask
        new_img[i,j] = new_value
```

برای یکی کردن ۲ تصویر از دستور cv2.hconcat([img1,img2]) استفاده کنید.

۲- تمرین قبلی را تکرار کنید ولی این بار ۲ بیت اول را صفر کنید . نتیجه را نمایش دهید . بیتهای بعدی را یکی یکی صفر کنید و ببینید که تا کجا تصویر از نظر ظاهری بدون تغییر می ماند .

ماسکی که ۲ بیت را صفر میکند به صورت زیر است :

```
mask = int('11111100', 2)
```

۲- میخواهیم از ۲ بیت اول هر پیکسل برای پنهان کردن داده ها استفاده کنیم . برای این منظور یک تصویر را به عنوان تصویر پایه انتخاب کنید . یک تصویر از خودتان را هم به عنوان تصویر رمز انتخاب کنید که سایز آن حداکثر یک چهارم تصویر اولیه باشد . برای سادگی هر دو تصویر را gray کنید.

حالا تصویر رمز را پیکسل به پیکسل بخوانید و مقدار هر پیکسل را به ۸ بیت تبدیل کنید.

بیتها را ۲تا -۲تا از راست به چپ جدا کنید . مقدار هر ۲ بیت را در پیکسل های تصویر پایه جاگزین کنید که اینکار را با عملیات بیتی مشابه تمرین قبلی انجام دهید .

هر بایت که ۲ بیت آن تغییر یافته است را در تصویر جدید ذخیره کنید .

تصویر جدید از نظر ظاهری شبیه تصویر پایه است ولی دو بیت اول هر بایت متعلق به یک تصویر دیگر است .

۳- در تمرین قبلی یک تصویر را پنهان کردیم. حالا برنامه ای بنویسید که تصویر پنهان شده را از تصویر قبلی بیرون بکشد .

برای این منظور تصویر را ۴بایت به ۴بایت بخوانید و هر بایت را به رشته بیتی تبدیل کنید

از هر بایت ، ۲ بیت اول را بخوانید . ۴ تا دوبیت را به هم بچسبانید و یک بایت بسازید و در محل مناسب از یک آرایه دیگر ذخیره کنید .

فرآیند را تا کامل شدن تصویر پنهان ادامه دهید .

۴ – در این تمرین میخواهیم یک عمل ساده از خانواده Augmented reality یا واقعیت افزوده انجام دهیم که بسیار ابتدایی و ساده است. برای این منظور یک تصویر به عنوان ماسک انتخاب کنید . مثلا یک شخصیت کارتونی .

در یک ویرایشگر تصویر مانند فتوشاپ تصویر را از نظر ابعادی بررسی کنید . فاصله بین ۲ گوشه چشم را به عنوان فاصله مرجع بدست آورید.

حالا تصویر وبکم خود را در یک حلقه بخوانید و با استفاده از dlib لندمارکهای هر فریم را بدست آورید و سعی کنید با استفاده از نقاط گوشه چشم تصویر ماسک را در محل مناسب قرار دهید .

در صورتیکه سر خود را به سمت شانه متمایل کنید چه اتفاقی می افتد ؟ چه راه حلی پیشنهاد میکنید؟ آنرا پیاده سازی کنید .

در صورتیکه از دوربین فاصله بگیریم یا به دوربین نزدیک شویم چه اتفاقی می افتد ؟ چه راه حلی پیشنهاد میکنید؟ آنرا پیاده سازی کنید.

.

۵- یک تصویر پایه (base.png) و یک تصویر هدف (target.png) در اختیار شما قرار گرفته شده است . سعی کنید با فیلترهای average,gaussian, bilateral و با سایزهای مناسب تصویر پایه را به تصویر هدف تبدیل کنید .

بعد از پیدا کردن ترکیب مناسب از فیلترها ، یک تابع بنویسید و آنرا به تصور وبکم اعمال کنید . آیا امکان اعمال تابع به صورت real-time وجود دارد ؟