```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 1,
   "metadata": {},
   "outputs": [],
   "source": [
    "import cv2"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 2,
   "metadata": {},
   "outputs": [],
   "source": [
    "# membaca image\n",
    "image = cv2.imread(\"leaf.jpg\")"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 3,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "[[[249 251 251]\n",
      "  [251 253 253]\n",
      "  [251 253 253]\n",
      "  ...\n",
      "  [246 246 246]\n",
      "  [242 242 242]\n",
      "  [248 248 248]]\n",
      "\n",
      " [[245 247 247]\n",
      "  [247 249 249]\n",
      "  [249 251 251]\n",
      "  ...\n",
      "  [245 245 245]\n",
      "  [241 241 241]\n",
      "  [246 246 246]]\n",
      "\n",
```

```
       " [[248 250 250]\n",
       "  [250 252 252]\n",
       "  [251 253 253]\n",
       "  ...\n",
       "  [247 247 247]\n",
       "  [245 245 245]\n",
       "  [247 247 247]]\n",
       "\n",
       " ...\n",
       "\n",
       " [[228 235 232]\n",
       "  [229 236 233]\n",
       "  [230 237 234]\n",
       "  ...\n",
       "  [235 238 236]\n",
       "  [235 238 236]\n",
       "  [235 238 236]]\n",
       "\n",
       " [[231 238 235]\n",
       "  [232 239 236]\n",
       "  [234 241 238]\n",
       "  ...\n",
       "  [235 238 236]\n",
       "  [235 238 236]\n",
       "  [235 238 236]]\n",
       "\n",
       " [[231 238 235]\n",
       "  [232 239 236]\n",
       "  [234 241 238]\n",
       "  ...\n",
       "  [235 238 236]\n",
       "  [235 238 236]\n",
       "  [235 238 236]]]\n"
      ]
     }
    ],
    "source": [
     "print(image)"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": 8,
    "metadata": {},
    "outputs": [],
    "source": [
     "# menampilkan image\n",
```

```
      "cv2.imshow('leaf', image)\n",
      "cv2.waitKey(0)\n",
      "cv2.destroyAllWindows()"
     ]
    },
    {
     "cell_type": "code",
     "execution_count": 5,
     "metadata": {},
     "outputs": [
      {
       "name": "stdout",
       "output_type": "stream",
       "text": [
        "(709, 567, 3)\n"
       ]
      }
     ],
     "source": [
      "print(image.shape)"
     ]
    },
    {
     "cell_type": "code",
     "execution_count": 6,
     "metadata": {},
     "outputs": [
      {
       "name": "stdout",
       "output_type": "stream",
       "text": [
        "blue =  249\n",
        "green =  251\n",
        "red =  251\n"
       ]
      }
     ],
     "source": [
      "# mengakses nilai di pixel x=100, y=20\n",
      "(b, g, r) = image[20, 100]\n",
      "print(\"blue = \",b)\n",
      "print(\"green = \",g)\n",
      "print(\"red = \",r)"
     ]
    },
    {
     "cell_type": "code",
```

```
   "execution_count": 9,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "(560, 280, 3)\n"
     ]
    }
   ],
   "source": [
    "# crop image\n",
    "im_crop = image[100:660, 100:380]\n",
    "\n",
    "cv2.imshow('crop',im_crop)\n",
    "cv2.waitKey(0)\n",
    "cv2.destroyAllWindows()\n",
    "print(im_crop.shape)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 10,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "(709, 567, 3)\n"
     ]
    }
   ],
   "source": [
    "# mengcopy image\n",
    "cp_image = image.copy()\n",
    "print(cp_image.shape)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 15,
   "metadata": {},
   "outputs": [],
   "source": [
    "# mengubah nilai pixel\n",
```

```
      "cp_image[300:350, 170:300] = (100, 255, 240)\n",
      "\n",
      "cv2.imshow('cp_image',cp_image)\n",
      "cv2.waitKey(0)\n",
      "cv2.destroyAllWindows()"
     ]
    },
    {
     "cell_type": "code",
     "execution_count": 16,
     "metadata": {},
     "outputs": [],
     "source": [
      "# resize image (ignore aspect ratio)\n",
      "im_resized = cv2.resize(image, (400,400))\n",
      "\n",
      "cv2.imshow('im_resized',im_resized)\n",
      "cv2.waitKey(0)\n",
      "cv2.destroyAllWindows()"
     ]
    },
    {
     "cell_type": "code",
     "execution_count": 17,
     "metadata": {},
     "outputs": [],
     "source": [
      "# resize image (mempertahankan aspect ratio)\n",
      "r = 400/image.shape[1]\n",
      "dim = (400,int(image.shape[0]*r))\n",
      "im_resized = cv2.resize(image, dim)\n",
      "\n",
      "cv2.imshow('im_resized',im_resized)\n",
      "cv2.waitKey(0)\n",
      "cv2.destroyAllWindows()"
     ]
    },
    {
     "cell_type": "code",
     "execution_count": 19,
     "metadata": {},
     "outputs": [],
     "source": [
      "# Rotating an image\n",
      "(h, w) = image.shape[:2]\n",
      "center = (w/2, h/2)\n",
      "\n",
```

```
      "M = cv2.getRotationMatrix2D(center, 180, 1.0)\n",
      "rotated = cv2.warpAffine(image, M, (w,h))\n",
      "\n",
      "cv2.imshow('rotate',rotated)\n",
      "cv2.waitKey(0)\n",
      "cv2.destroyAllWindows()"
     ]
    },
    {
     "cell_type": "code",
     "execution_count": 20,
     "metadata": {},
     "outputs": [
      {
       "data": {
        "text/plain": [
         "True"
        ]
       },
       "execution_count": 20,
       "metadata": {},
       "output_type": "execute_result"
      }
     ],
     "source": [
      "# menyimpan image\n",
      "cv2.imwrite(\"rotate.png\", rotated)"
     ]
    },
    {
     "cell_type": "code",
     "execution_count": 25,
     "metadata": {},
     "outputs": [],
     "source": [
      "# adjust image contrast\n",
      "import numpy as np\n",
      "im_adjusted = cv2.addWeighted(im_resized, 1.5, np.zeros(im_resized.shape, im_resized.dtype), 0, -100)\n",
      "\n",
      "cv2.imshow('Original Image',im_resized)\n",
      "cv2.imshow('Adjusted Image',im_adjusted)\n",
      "cv2.waitKey(0)\n",
      "cv2.destroyAllWindows()"
     ]
    },
    {
     "cell_type": "code",
```

```
   "execution_count": 26,
   "metadata": {},
   "outputs": [],
   "source": [
    "# detect edges\n",
    "im_edges = cv2.Canny(im_resized, 100, 200)\n",
    "\n",
    "cv2.imshow('Original Image',im_resized)\n",
    "cv2.imshow('Detected Edges',im_edges)\n",
    "cv2.waitKey(0)\n",
    "cv2.destroyAllWindows()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 27,
   "metadata": {},
   "outputs": [],
   "source": [
    "# convert image to grayscale\n",
    "im_gray = cv2.cvtColor(im_resized, cv2.COLOR_BGR2GRAY)\n",
    "\n",
    "cv2.imshow('Original Image',im_resized)\n",
    "cv2.imshow('Grayscale Image',im_gray)\n",
    "cv2.waitKey(0)\n",
    "cv2.destroyAllWindows()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 28,
   "metadata": {},
   "outputs": [],
   "source": [
    "# get all files in a folder\n",
    "import glob\n",
    "\n",
    "imdir = 'dataset/elephant/'\n",
    "ext = ['png', 'jpg', 'gif']    # Add image formats here\n",
    "\n",
    "files = []\n",
    "[files.extend(glob.glob(imdir + '*.' + e)) for e in ext]\n",
    "\n",
    "images = [cv2.imread(file) for file in files]\n",
    "\n",
    "# adjust contrast to all of them nd save to different location\n",
    "i = 1\n",
```

```
     "for img in images:\n",
     "    im_adjusted = cv2.addWeighted(img, 1.5, np.zeros(img.shape, img.dtype), 0, -100)\n",
     "    im_name = \"dataset/elephant_contrast/\" + str(i) + \".jpg\"\n",
     "    cv2.imwrite(im_name, im_adjusted)\n",
     "    i+=1"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": null,
    "metadata": {},
    "outputs": [],
    "source": []
   }
  ],
  "metadata": {
   "kernelspec": {
    "display_name": "Python 3",
    "language": "python",
    "name": "python3"
   },
   "language_info": {
    "codemirror_mode": {
     "name": "ipython",
     "version": 3
    },
    "file_extension": ".py",
    "mimetype": "text/x-python",
    "name": "python",
    "nbconvert_exporter": "python",
    "pygments_lexer": "ipython3",
    "version": "3.8.5"
   }
  },
  "nbformat": 4,
  "nbformat_minor": 2
 }
```