Copy Images

Create a Collage

Cool Computing Codes:
- range
- nested loops

Beginnings:
1. Launch JES, on your desktop
2. Load the Python code, icepicture.py (use File, Open, then “Load”)
3. Using interaction pane, at bottom
   >>> setMediaPath()
4. >>> fileName = pickAFile()       { select “robot.jpg” }
5. >>> robotPic = makePicture(fileName)
6. >>> getWidth(robotPic)
7. >>> getHeight(robotPic)
8. >>> show(robotPic)
9. Where is the middle of the robot picture?
10. To make it look like a different robot, what colors should we change?
    (How do we do that?)
11. What does it look like in “negative” filter? (Write a negative function! OR grab it from our filter text files.

Try Copying and Making a Collage
1. Use the empty canvas from, canvas.jpg from your MediaSources folder
2. Open “canvas.jpg”
   >>> filename=pickAFile()
   >>> myCanvas=makePicture(fileName)
3. Open “robot.jpg”
   >>> fileName=pickAFile()
   >>> robotPic=makePicture(fileName)
4. Copy Robot to Canvas, several times
   >>> copyPicture2(robotPic,myCanvas,1,1)
   >>> copyPicture2(robotPic,myCanvas,150,150)
   >>> copyPicture2(robotPic,myCanvas,300,400)
   >>> copyPicture2(robotPic,myCanvas,400,500)
5. Build a Simple Collage
   >>> collage(robotPic,myCanvas)
Create a Collage
Try creating a collage using one of the “collage” functions provided. 
Hint: Your picture should be small, to be able to fit several copies on the “canvas”. 
Beware of “out of bounds” errors.

```python
>>>myPic=makePicture(fileName)
>>>myCanvas=makePicture("canvas.jpg")
>>>collage(myPic, myCanvas)
```

**Challenge: Design your own collage**
1. Select an image
2. Apply changes to the image to create at least 3 new looks:
   Change colors, negate, greyscale
3. Copy at least 4 changed images onto the blank canvas
4. Save your collage using writePictureTo(picture, “filename.jpg”)

---

Copy Image

Recap for Copy Images

**Range Function**
Range returns a sequence between its first two inputs, possibly using a third input as the increment.

```python
>>> print range(1,4)
[1, 2, 3]
>>> print range(1,10,2)
[1, 3, 5, 7, 9]
```

To use `range`, we’ll have to use **nested loops**
- One to walk the width, the other to walk the height
- Be sure to watch your blocks carefully!

Here is an example, another function to increase Red in a picture!

```python
def increaseRed2(picture):
    for x in range(1,getWidth(picture)):
        for y in range(1,getHeight(picture)):
            px = getPixel(picture,x,y)
            value = getRed(px)
            setRed(px,value*1.1)
```
In general to copy a picture, what we want to do is to keep track of a \textbf{sourceX} and \textbf{sourceY}, and a \textbf{targetX} and \textbf{targetY}. These are (X,Y) coordinate pairs for the picture (source) and the canvas (target).

We \textit{increment} (add to them) in pairs

- sourceX and targetX get incremented together
- sourceY and targetY get incremented together

The tricky parts in the copy function below are:

- Setting values \textit{inside} the body of loops
- Incrementing at the \textit{bottom} of loops

# A function to copy a picture  
\texttt{def copyPicture(sourcePic):}
# Input parameter is source picture
# Set up the target picture
\texttt{fileName = getMediaPath("canvas.jpg")}
\texttt{canvas = makePicture(fileName)}
# Now do the actual copying
\texttt{targetX = 1}
\texttt{for sourceX in range (1, getWidth(sourcePic)):
  targetY=1
  for sourceY in range (1,getHeight(sourcePic)):
    color = getColor(getPixel(sourcePic,sourceX,sourceY))
    setColor(getPixel(canvas,targetX,targetY),color)
    targetY=targetY + 1
  targetX= targetX+1
show(sourcePic)
show(canvas)
return canvas}
Transformations to Copy Function

Making relatively small changes in this basic copying program can make a variety of transformations.

1. **Translating**: Change the targetX and targetY, and you copy wherever you want.
2. **Cropping**: Change the sourceX and sourceY range, and you copy only part of the picture.
3. **Rotating**: Swap targetX and targetY, and you end up copying sideways.
4. **Scaling**: Change the increment on sourceX and sourceY, and you either grow or shrink the image.

```python
# places 4 modified images across the middle of canvas
# suited to vertical mirroring
def collage(picture, canvas):
    middleX = getWidth(canvas)/2
    middleY = getHeight(canvas)/2
    targetX = middleX - (getWidth(picture)*2)
    targetY = middleY - getHeight(picture)
    # Check placement values
    copyPicture2(picture, canvas, targetX, targetY)
    increaseRed2(picture)
    targetX = middleX - getWidth(picture)
    copyPicture2(picture, canvas, targetX, targetY)
    negative(picture)
    targetX = middleX
    copyPicture2(picture, canvas, targetX, targetY)
    greyscale(picture)
    targetX = middleX + getWidth(picture)
    copyPicture2(picture, canvas, targetX, targetY)
    show(canvas)
    return canvas
```