در این بخش چند تابع پیاده سازی شده که ابتدا توضیحشان میدهیم.

تابع `makeBorder` :

ورودی ها:
1) یک ماسک باینری

خروجی ها:
1) ماسکی که مرز های آن با رنگ خاکستری مشخص شده اند

عملکرد: هدف این است که مرز بین ناحیه سفید و ناحیه سیاه ماسک را خاکستری کنیم. intensity رنگ خاکستری را برابر با `borderColor=100` نشان میدهیم. برای این کار تک تک پیکسل ها را بررسی میکنیم و تمام پیکسل های سیاهی را که در چهار طرف بالا، پایین، چپ و راست پیکسل سفید قرار دارند را خاکستری میکنیم. و در نهایت ماسک حاصل را بعنوان خروجی بازگشت میدهیم.

تابع `laplacian` :

ورودی ها:
1) تصویری که سیاه سفید است یعنی یک چنل دارد
2) مختصات نقطه‌ای که لاپلاسین آن را میخواهیم

خروجی ها:
1) لاپلاسین آن نقطه

عملکرد: طبق فرمول لاپلاسین، باید مقدار پیکسلی که لاپلاسین آن را میخواهیم ضرب در چهار شود و سپس منهای مقادیر پیکسل های اطرافش یعنی بالا، پایین، چپ و راست شود و عدد حاصل بعنوان خروجی بازگشت داده شود.

تابع `getXMatrix` :

ورودی ها:
1) تصویر مبدا و مقصد
2) ماسک که مربوط عکس مبدا میشود

خروجی ها:
1) ماتریس جوابها

عملکرد: باید هر سه تصویر مبدا و مقصد و ماسک داده شده به این تابع ابعاد یکسانی داشته باشند. هدف این است که معادله پواسون را حل کنیم به طوری که لاپلاسین نقاط داخلی را داریم، همچنین مقادیر را روی مرز داریم و باید مقادیر داخل ناحیه سفید ماسک را بدست آوریم. لاپلاسین نقاط داخل این ناحیه از لاپلاسین تصویر مبدا گرفته میشوند و مقادیر مرزی از مقادیر تصویر مقصد گرفته میشوند تا ترکیب رنگ ها مشابه تصویر مقصد و جزئیات مشابه تصویر مبدا باشند. پس نقاط داخل ماسک را اندیس گذاری

میکنیم و اندیس نسبت داده شده به هر یک از نقاط را در یک دیکشنری ذخیره میکنیم که کلید های این دیکشنری مختصات آن نقطه و مقدار هر کلید اندیس آن نقطه است. حال ماتریس A و b را طبق فرمول زیر تشکیل میدهیم و به n معادله و n مجهول میرسیم که n همان تعداد نقاط داخل ناحیه (سفید) است. با حل معادله AX=b تمامی مقادیر $f_{i,j}$ بدست میایند و ماتریس جوابها یا همان X را بعنوان خروجی بازگشت میدهیم.

برای مقدار دهی نقاط داخل ناحیه (سفید): ($g_{i,j}$ لاپلاسین نقطه‌ی (i,j) در تصویر مبدا است)

$$f_{i+1,j} + f_{i-1,j} + f_{i,j+1} + f_{i,j-1} - 4f_{i,j} = g_{i,j} = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \Longrightarrow Af = b$$

برای مقدار دهی نقاط روی مرز ناحیه (خاکستری): ($h_{i,j}$ مقدار نقطه‌ی (i,j) در تصویر مقصد است)

$$f_{i,j} = h_{i,j}$$

حال پس از توضیحات مربوط به توابع پیاده سازی شده به توضیحات روند استفاده از آنها و رسیدن به خروجی مطلوب میپردازیم.

ابتدا مرز ماسک را با استفاده از تابع makeBorder تعیین میکنیم. سپس قطعه عکسی را به ابعاد تصویر مبدا، از تصویر مقصد برش میدهیم که قرار است تصویر مبدا در این قطعه از تصویر مقصد قرار گیرد. حال تصویر برش یافته و تصویر مبدا و تصویر ماسکی که مرز آن مشخص است را به تابع getXMatrix پاس میدهیم و هر سه چنل را مقدار دهی میکنیم به گونه ای که با قرار گرفتن تصویر حاصل با مقادیر جدید، در قطعه‌ی برش یافته از تصویر مقصد، رنگ ها هماهنگ اند و مشخص نیست که دو عکس متفاوت ادغام شده اند.