

Final Exam, CS130

Instructions. This exam is similar to the midterms, except that you have two hours and fifteen minutes to work. The executable file to submit is *Final.exe*. Don't forget to put your ID number in the title bar—failure to do so costs one letter grade. For reference here are the rules again: you may use any books, notes, the course website, your past work, etc. Email, instant messaging, etc. are illegal and so is searching the Internet; only the course website is allowed. Submit the exam when ready at the link on the course website. Multiple submissions are OK. The link will automatically go dead when the exam period is over, so be sure you submit on time.

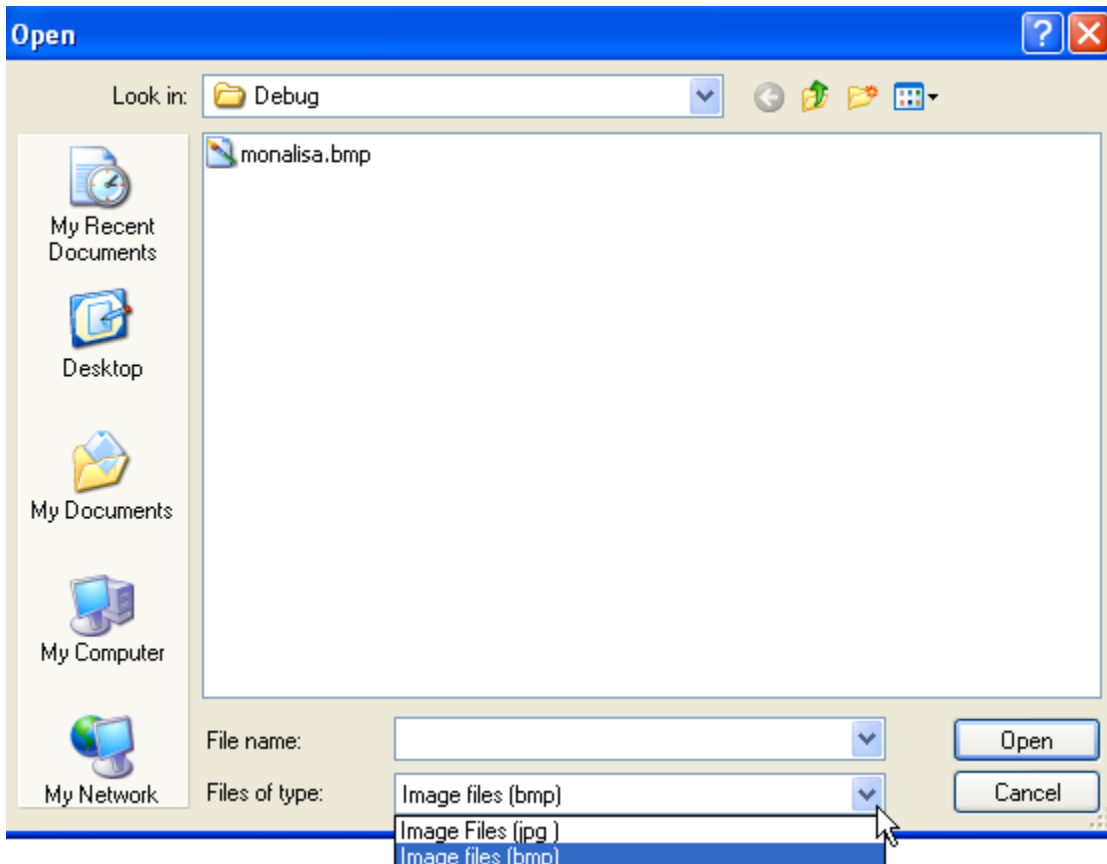
Read every line of the exam sheet before programming!

[There are several parts of the program that can earn you points quickly, and other parts that will make you sweat. Use your time wisely.]

The program is called *Photo Cropper* and its purpose is to let you crop your photographs (or any other image). To crop a photograph is to select a rectangle from the photograph and keep only the part of the image inside the selected rectangle, discarding the rest.

The program should exhibit the following behaviors. (Some programming hints are also given in between the specifications.)

1. Create a menu offering *File* and *Image* on the menu bar. Under *File* you have *Open* and *Save As*. (You don't have *Save* because it's not allowed to save the image in the same file it came from.) Under *Image* you have *Crop*. The *Crop* menu item should be initially disabled, and should become enabled later when an image file is opened.
2. Implement your file dialog for *Open* so that you can only see bitmap (bmp) and jpg (or jpeg) files. The user should see two choices of file type, one for bmp and one for jpg or jpeg files. The correct file extension (.bmp or .jpg) should be supplied automatically and only files of the selected type should be visible. (Of course folders are always visible.) The default file extension (i.e. the filter that comes up initially) is .bmp, not .jpg. Here's a screen shot, showing the desired filters. Of course, the exact appearance of this dialog is determined by Windows and the user's settings within Windows, so your results may differ, but the filters should be as shown.



3. When *File / Open* is chosen, the selected image is displayed. If there is already an image file displayed, the new image replaces the old one (without any warning or dialog). *Programming hint: this is almost the same as in your Animals homework.*

4. The user can use the mouse, as in the lecture on *Dragging*, to select a rectangle using the left mouse button. The selected rectangle is outlined in red and the image is still visible inside and outside the rectangle. At the top of the next page is a screen shot. (The Mona Lisa's face is in a selection rectangle. I hope this will be visible when it's printed in black and white.) The image file was the familiar *monalisa.bmp*.



(Programming hint: Use a *PictureBox* to hold the image, but be aware that the mouse messages will come to the *PictureBox*, not to your main form.)

If you drag the mouse off the image, the red rectangle is never drawn outside the rectangle, but it does invisibly follow the mouse, so when you drag back on the rectangle the red rectangle is still following the mouse.

5. When there is a selection rectangle, and the user chooses *Image / Crop*, then the image is replaced by the cropped image. In other words, only the cropped portion remains.



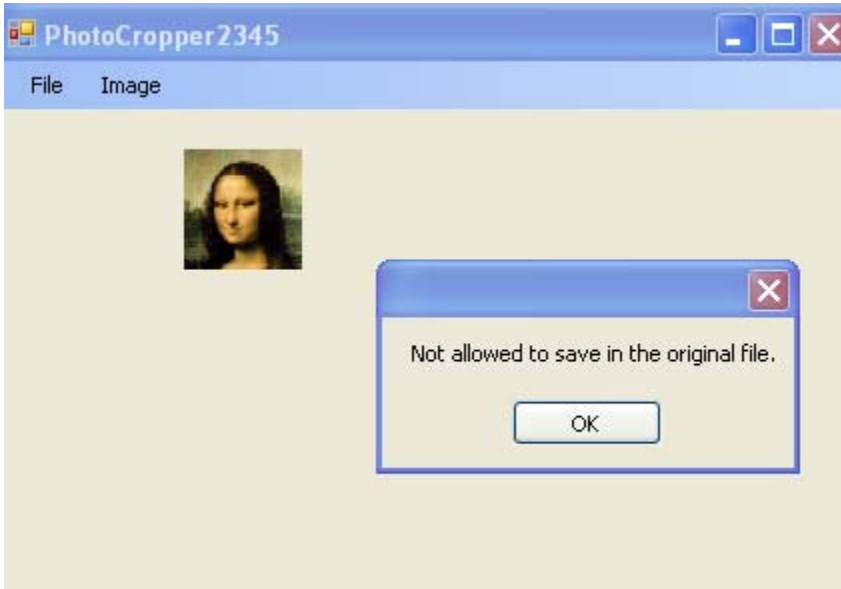
Programming hint--you can use the following code, which uses ideas and .NET methods discussed in the lectures on double-buffering and animation.

```
private Image crop(Image im, Rectangle r)
// return the portion of im specified by the rectangle r
{
    Graphics g0 = CreateGraphics();
    Bitmap bm = new Bitmap(r.Width, r.Height, g0);
    Graphics g = Graphics.FromImage(bm);
    Rectangle destination = new Rectangle(0, 0, r.Width,r.Height);
    g.DrawImage(im,destination,r,GraphicsUnit.Pixel);
    return bm; // note that a Bitmap can be returned as an Image
}
```

6. When the user chooses *File / Save As* then the dialog should suggest as a filename the original filename with *Cropped* appended. For example, *monalisaCropped*. The suggested filename should not have an extension as that will be supplied automatically.

The only filter in this dialog is *bmp*. (It's possible to write a version of this program that can open a .jpg file and save it as a .bmp, but if you use the simplest *Save* method in the *Image* class, you'll get a .bmp file no matter what you name it, and that's all that is required for the exam.)

7. If you attempt to save in the original file, you get an error message as shown in the next screen shot (although it doesn't have to come up in this exact location). If you attempt to save in some *other* existing file, that works fine, *without* a warning dialog.



8. Add a *Resize* item under the *Image* menu. It should bring up the following dialog:



Note that the dialog uses two masked textboxes that will only accept four (or fewer) digits. *Programming hint: in the Mask property, enter zeroes where numbers are required.* This way, you can't enter unacceptable data. The dialog should initialize with the size of the current image. The *Resize* menu item should be disabled until an image is loaded.

Up to here is the A level. What comes after are for the A+ students.

9. When OK is pressed in the Resize dialog, the image is resized to the specified width and height (in pixels). *Programming hint: the code is very similar to that given above for **crop**, except now you have a different source rectangle—the entire original image.*

10. Finally, add items to the *Image* menu: *Rotate clockwise* and *Rotate counterclockwise*. These rotate the image 90 degrees in the specified direction. *Programming hint: use Image.RotateFlip. You will need to invalidate the pictureBox!*

Here is a screen shot of the A+ level, showing the Mona Lisa doubled in size and rotated; and then I used the program itself to reduce the size of the screen shot to fit on this page:



Grading Criteria

1. Menu implemented as specified. Menu item *Crop* is disabled initially. [10 points]
2. *File | Open* menu item brings up a file dialog with the specified features. (.bmp the default extension, can select from two choices of file type. [10 points]
3. Pressing OK in the file dialog correctly causes an image to be displayed and menu item *Crop* to be enabled. [15 points]
4. You can use the left mouse to draw a visible red outlined rectangle in the image, without otherwise destroying any part of the image. The red outline will not extend outside the rectangle occupied by the image. [20 points]
5. Menu item *Crop* functions as specified, i.e. the portion of the image inside the red rectangle becomes the new displayed image. [15 points]
6. *File / Save As* brings up a file save-as dialog that functions as specified (no built-in overwrite warning, special warning against saving in the original file, correct filter, corrected suggested filename, extension supplied correctly). [10 points]
7. When you click *OK* in the file save-as dialog, the file is actually saved correctly, as can be verified by closing the program, opening it again, and opening the saved file. [10 points]
8. The *Resize* dialog is properly implemented: You can't enter invalid data. It initializes correctly. If you cancel nothing changes. When you press OK the image resizes, or if not, then the picture box has a frame so you can see the picture box is resized, so that at least shows you got the data back from the dialog. [10 points]

This is the A level. You can't get an A+ unless all of the above are working perfectly. For an A+ continue. You'll get 4.15 for one of these and 4.3 for both. There are 100 points above here. The A+ is not awarded by point count.

9. The image does resize properly when you press OK in the resize dialog (showing that you understood what you learned about bitmaps and images enough to modify the *crop* code a little).
10. The two rotation menu items are implemented correctly (showing that you can access the documentation enough to call a new function).