CSC317 Computer Graphics
Tutorial 1

September 11, 2024

Assignment 1: Raster Images

- Due Date: September 17 @ 11:59 pm
- Assignment description can be accessed through course github page, under “Lecture Schedule”

Week	Topic / Event
1	IntroductionDL,KS, tutorial, Assignment 1 (Raster Images), Math Practice**waitlisted ? zip assignment and email to TAs** due 17/09
2	Lecture 2, Assignment 2 Ray Casting due 24/09
3	Lecture 3, Assignment 3 Ray Tracing due 01/10

Assignment 1: Raster Images

- Every student is given ten (10) late days **for the entire semester** which are automatically applied, starting at midnight on the due date. Weekends count as late days.
- If you plan on not submitting an assignment (to take advantage of the best 8-of-9 policy) please let us know so late days aren't applied.

Assignment 1: Raster Images

- If you need help:
  - “Issues” page on github for A1
  - E-mail TA: csc317tas@cs.toronto.edu

Assignment 1: Raster Images

- Task1: rgba to rgb
  - Alpha channel: transparency of each pixel location
  - width*height*4 => width*height*3

- Input: const std::vector<**unsigned char**> & rgba
  - Getting size of vector: rgba.size()
  - Accessing i^{th} element: rgba.at(i) or rgba[i]
  - Debug: convert to *int* and output to console

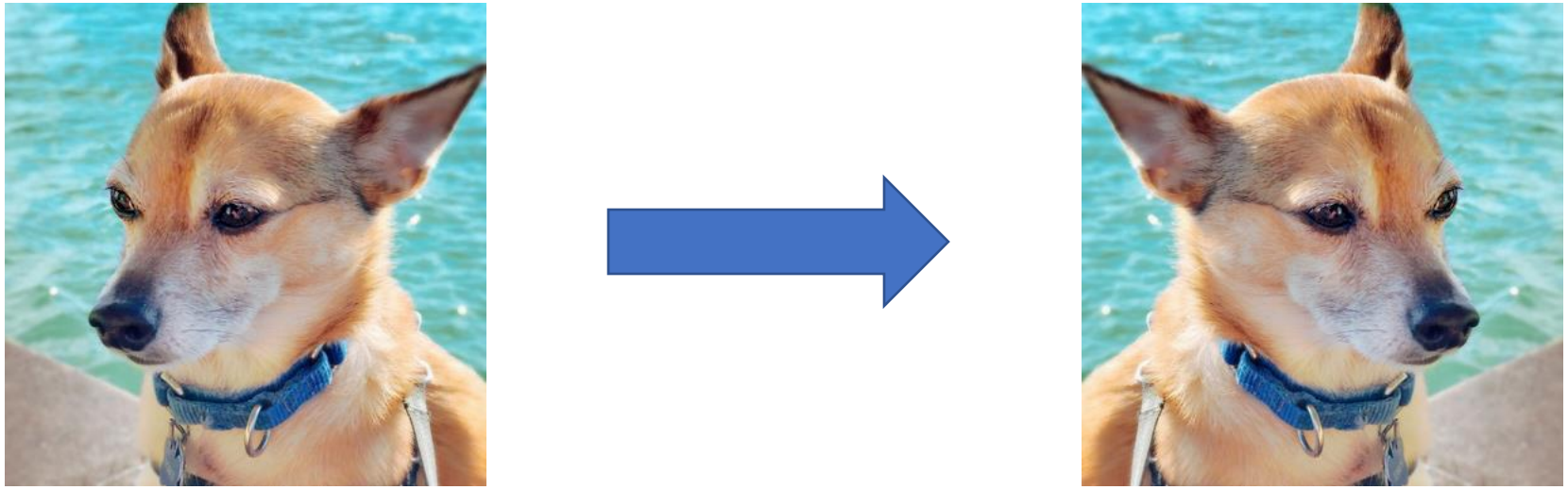
Assignment 1: Raster Images

- Task2: write ppm
  - Output result to a .ppm file
  - Support both rgb and grayscale images

- PPM
  - Uncompressed format
  - Header magic number (text-based): P2/P3
  - Output can be directly opened with a text editor

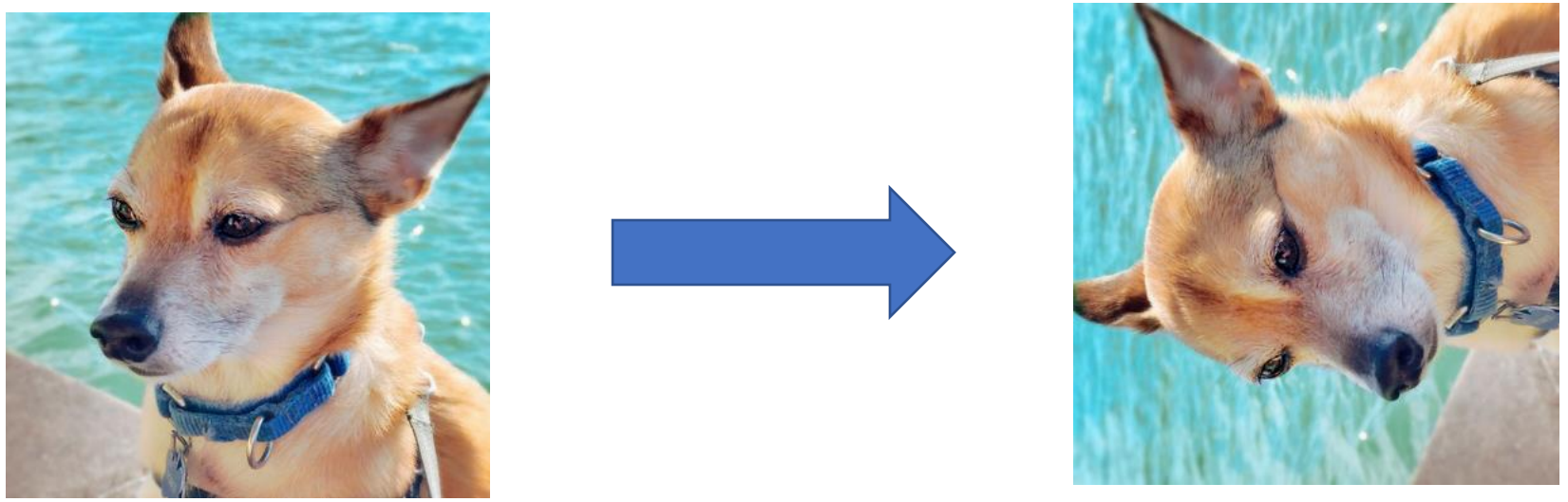
Assignment 1: Raster Images

- Task3: image reflection



Assignment 1: Raster Images

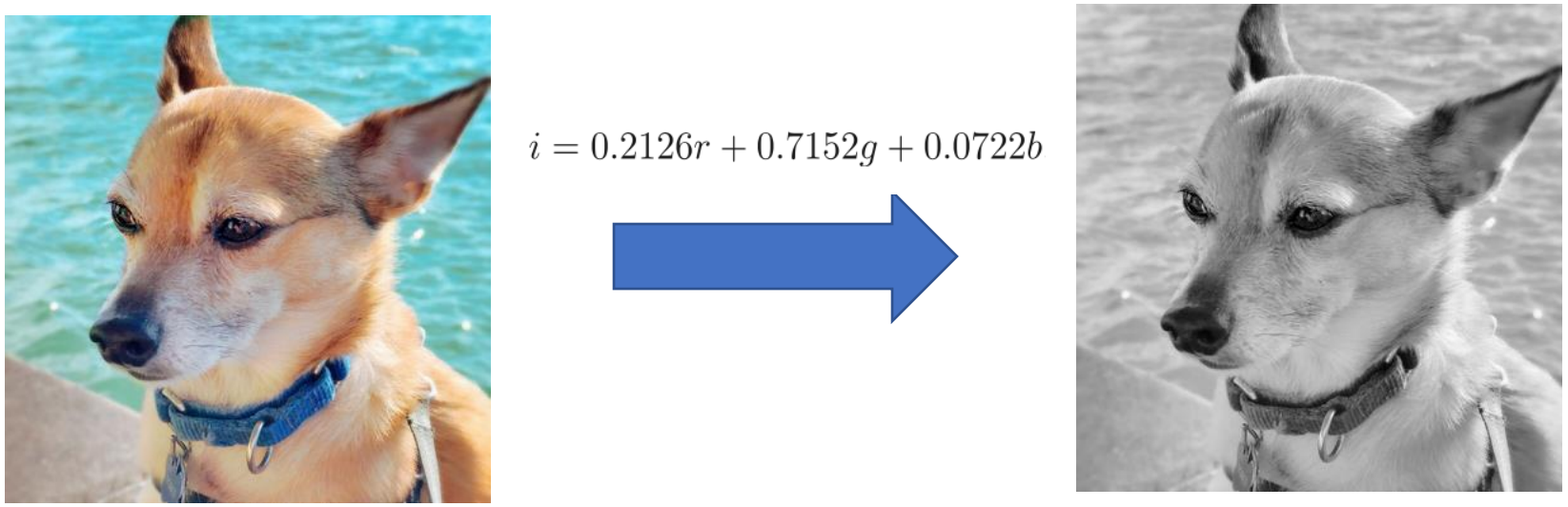
- Task4: image rotation (90 degrees CCW)



- Note: width/height swap

Assignment 1: Raster Images

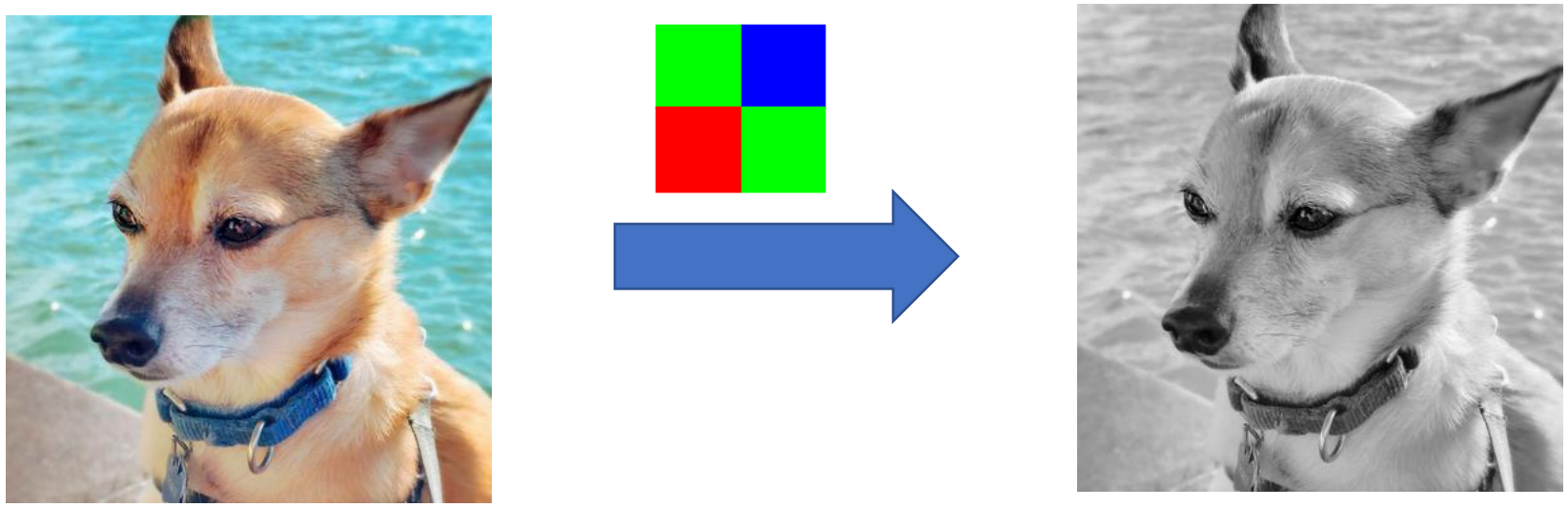
- Task5: rgb to gray



- 3 channels (width*height*3) to 1 channel (width*height)

Assignment 1: Raster Images

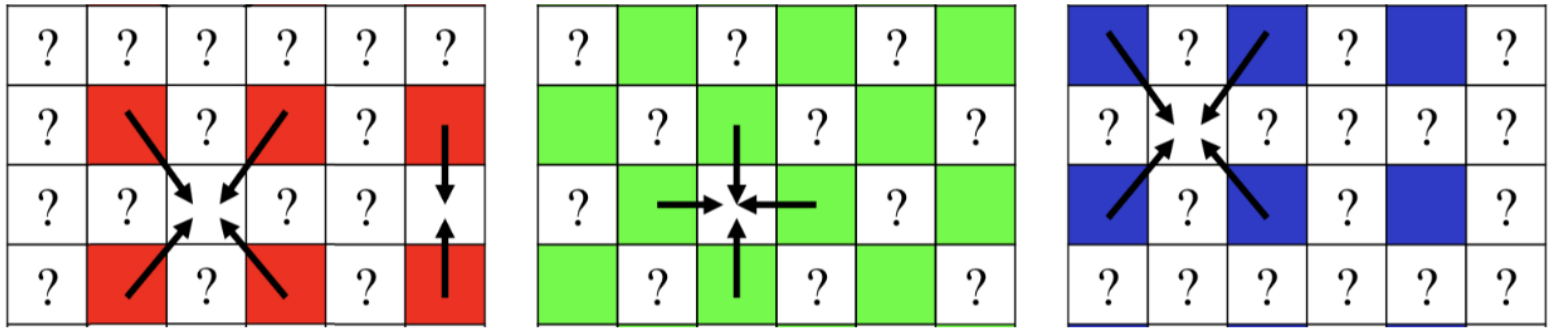
- Task6: simulate bayer mosaic



- 3 channels (width*height*3) to 1 channel (width*height)

Assignment 1: Raster Images

- Task7: Demosaic
- Inverse of the previous task => roughly get the original rgb image
- For each pixel
  - Take exact value if you have it (e.g. take R value if you are on a red pixel)
  - Approximate from the neighbours if you don't have exact value



- 1 channel (width*height) to 3 channels (width*height*3)

Assignment 1: Raster Images

- Task8: helper functions for HSV color space

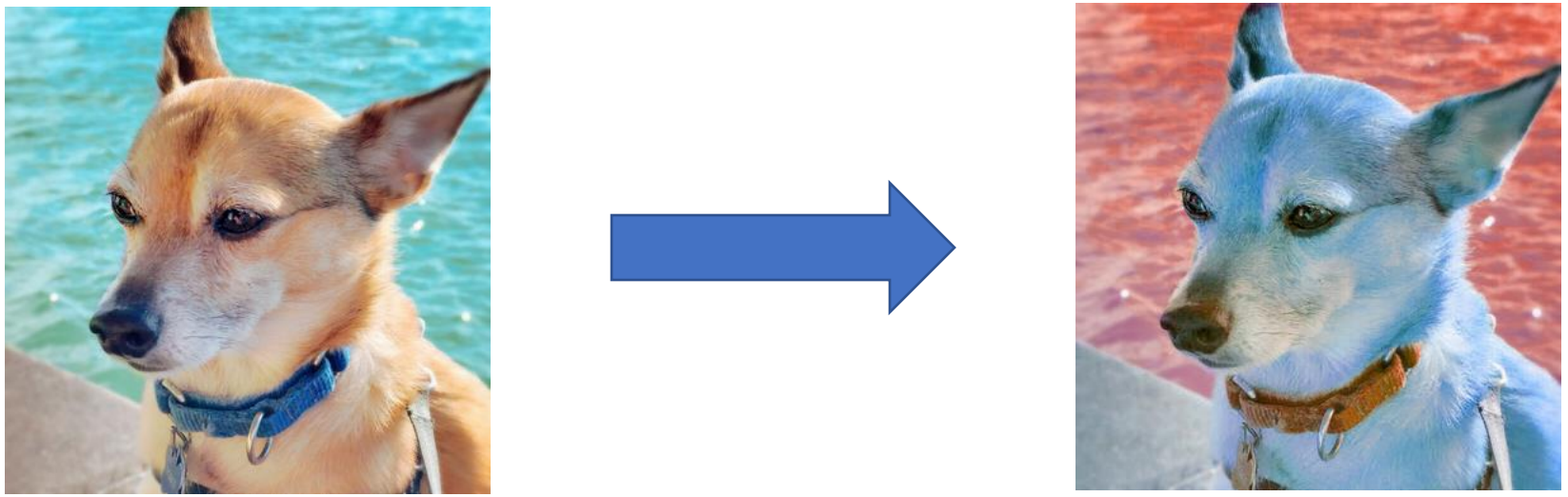
- RGB to HSV
  - ➢Check with standard values online for correctness

- HSV to RGB

- Be careful with data types (i.e. inputs are of type double)
- Be careful with input/output ranges (e.g. 0 to 1 vs. 0 to 255)

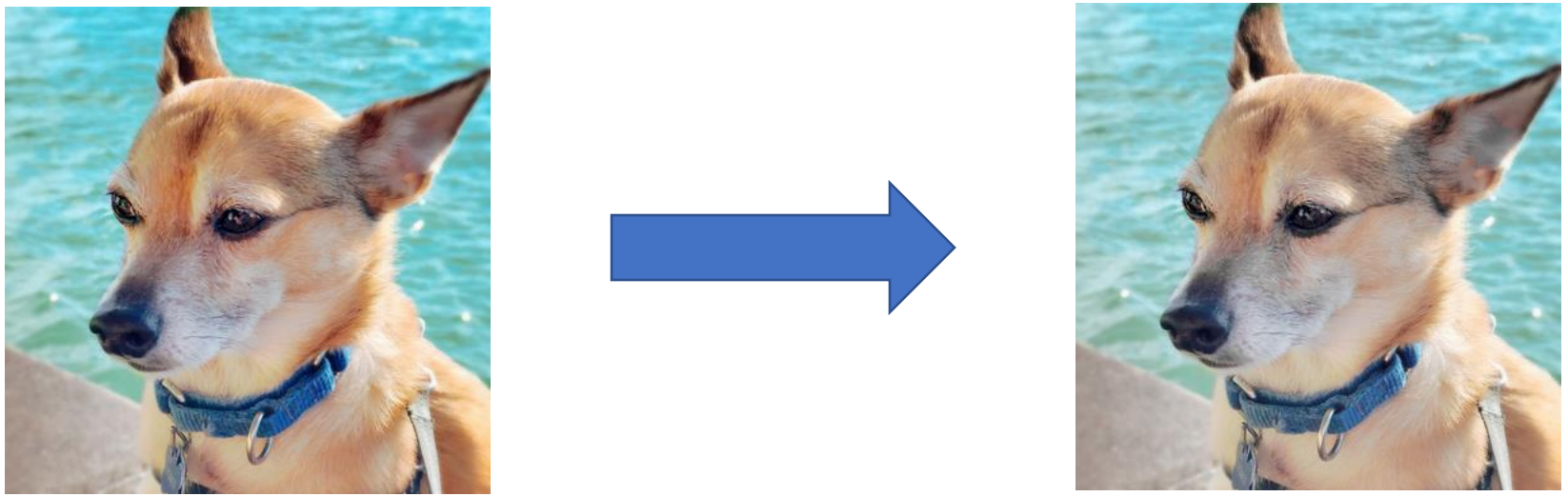
Assignment 1: Raster Images

- Task9: hue shift
- Use the HSV helper functions implemented previously



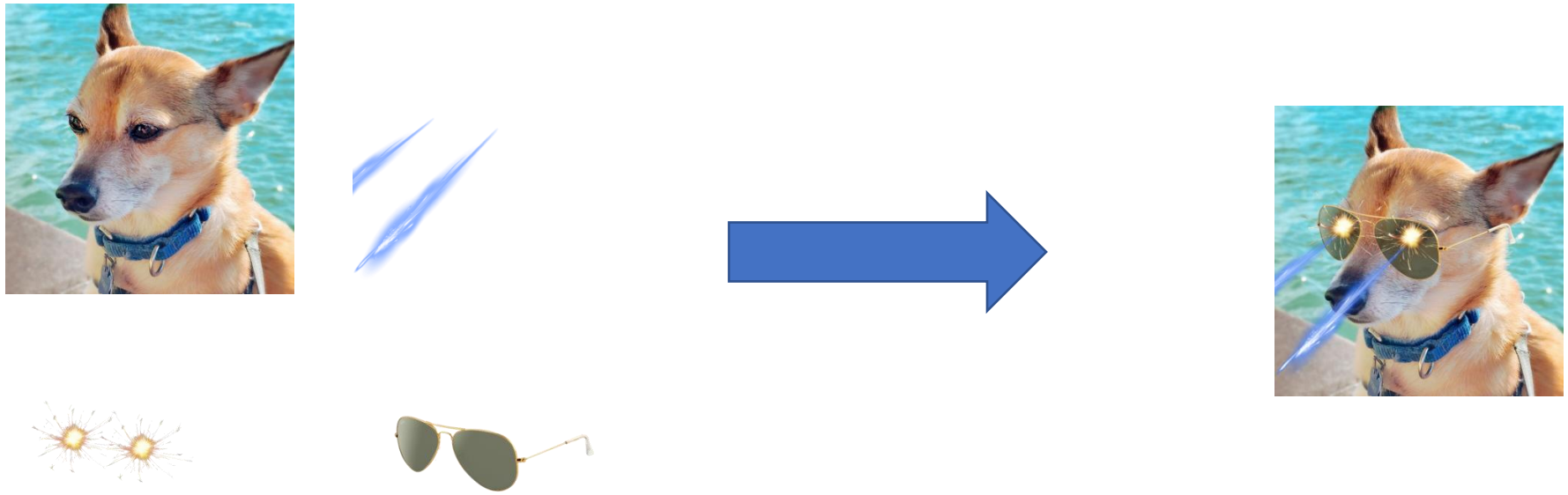
Assignment 1: Raster Images

- Task10: desaturate
- Use the HSV helper functions implemented previously



Assignment 1: Raster Images

- Task11: composite



Assignment 1: Raster Images

- Submission
  - Where? Markus
  - What? All .cpp files in the *src* directory (also listed on Markus)

Assignments

ASSIGNMENT	DUE DATE
A1: Raster Images	Tuesday, September 17, 2024, 11:59:00 PM EDT

demosaic.cpp

desaturate.cpp

hsv_to_rgb.cpp

hue_shift.cpp

over.cpp

reflect.cpp

rgb_to_gray.cpp

rgb_to_hsv.cpp

rgba_to_rgb.cpp

rotate.cpp

simulate_bayer_mosaic.cpp

write_ppm.cpp

Assignment 1: Raster Images

- Read assignment handout carefully: https://github.com/dilevin/computer-graphics-raster-images/tree/master
- Program entry: main.cpp
  - Input images
  - Overall flow
- Check PPM outputs online
- Experiment with different test cases

Assignment 1: Raster Images

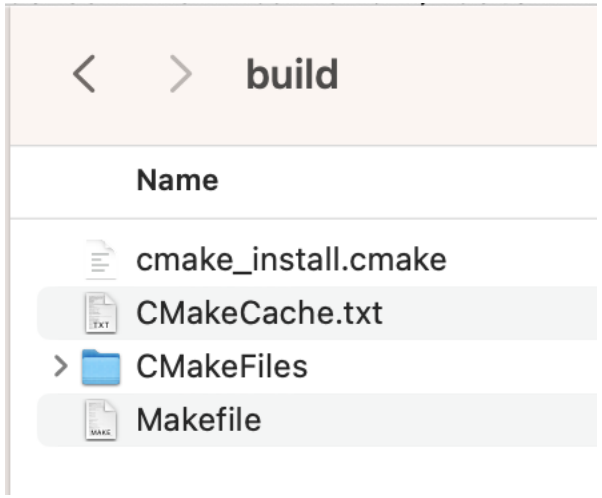
- Before implementing a function, check its corresponding header file for description

```h
#ifndef RGBA_TO_RGB_H
#define RGBA_TO_RGB_H

#include <vector>

// Extract the 3-channel rgb data from a 4-channel rgba image
//
// Inputs:
//   rgba  width*height*4 array of 4-channel rgba instensities (i.e., rgb +
//     alpha channel for transparency)
//   width  image width (i.e., number of columns)
//   height  image height (i.e., number of rows)
// Outputs:
//   rgb  width*height*3 array containing rgb image color intensities
//
void rgba_to_rgb(
  const std::vector<unsigned char> & rgba,
  const int & width,
  const int & height,
  std::vector<unsigned char> & rgb);

#endif

```

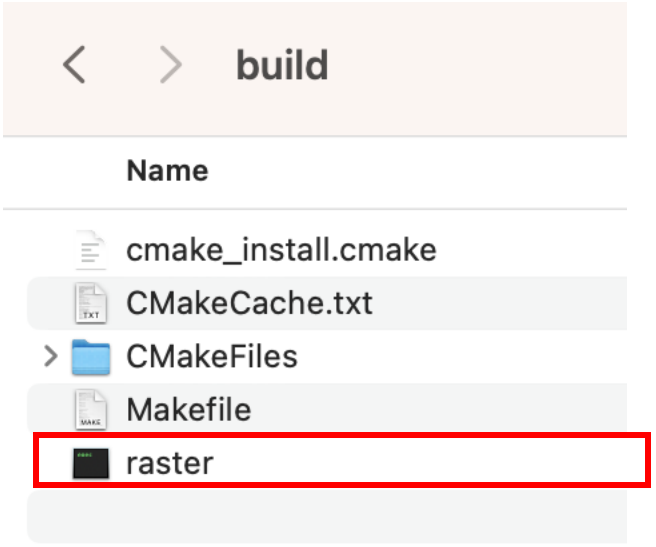
```cpp
#include "rgba_to_rgb.h"

void rgba_to_rgb(
  const std::vector<unsigned char> & rgba,
  const int & width,
  const int & height,
  std::vector<unsigned char> & rgb)
{
  rgb.resize(height*width*3);
  ////////////////////////////////////////////////////////////////////////////
  // Add your code here
  ////////////////////////////////////////////////////////////////////////////
}
```

Assignment 1: Raster Images

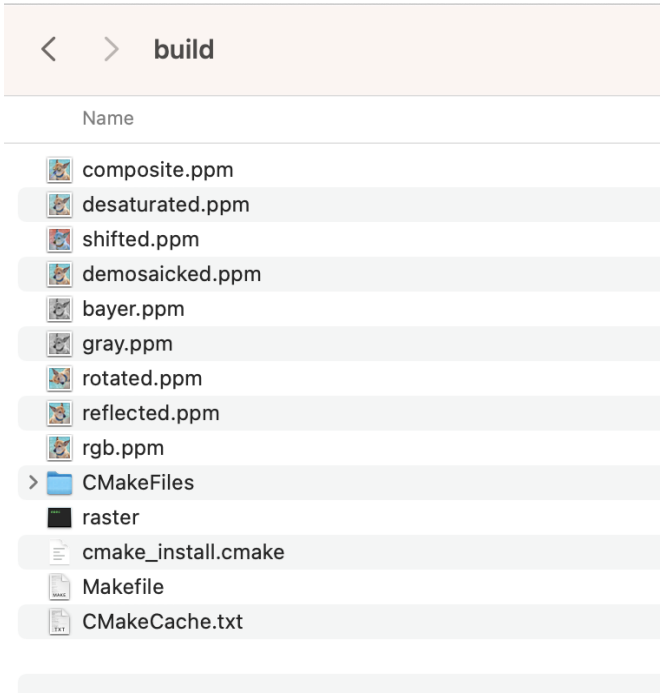
- Compilation and Running (Mac/Linux)



After “cmake ..”



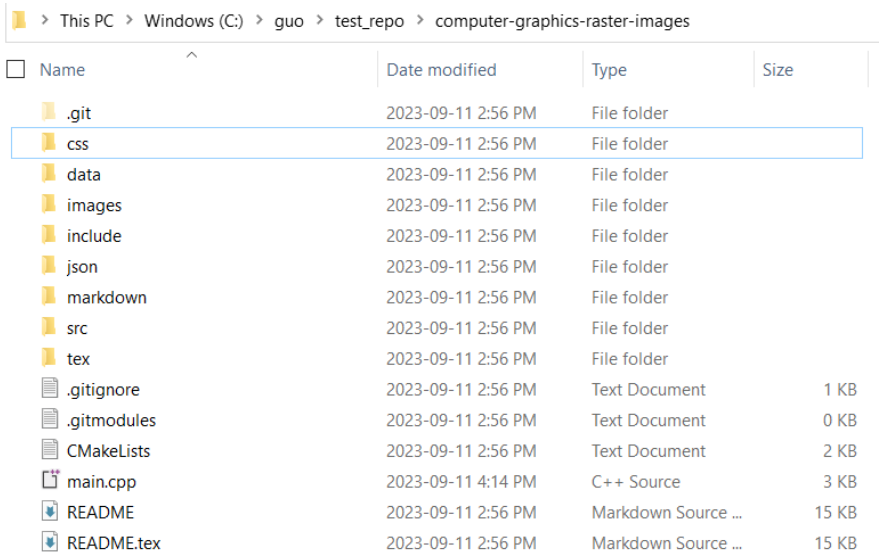
After “make”



After “./raster”

Assignment 1: Raster Images

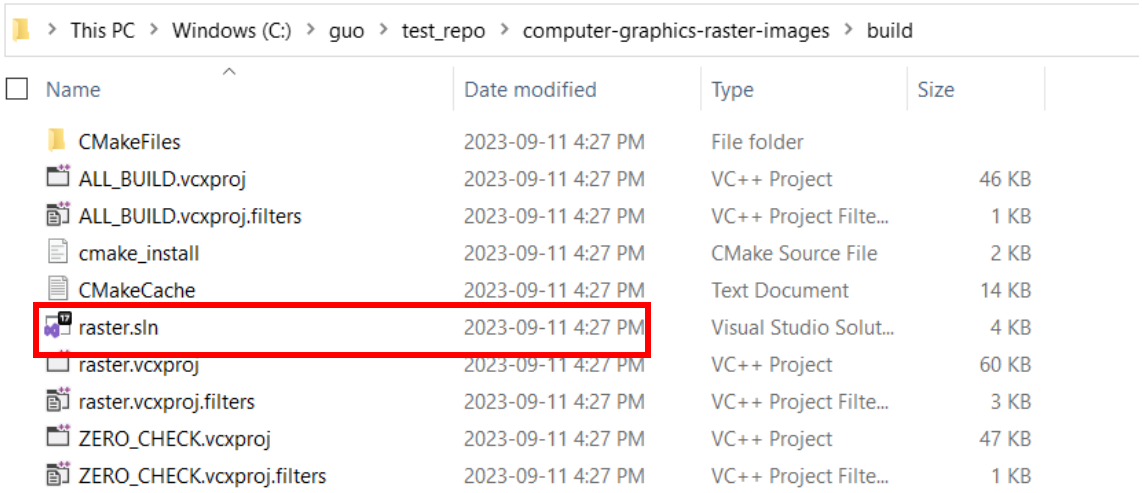
- Compilation and Running (Windows):



Fresh clone

Assignment 1: Raster Images

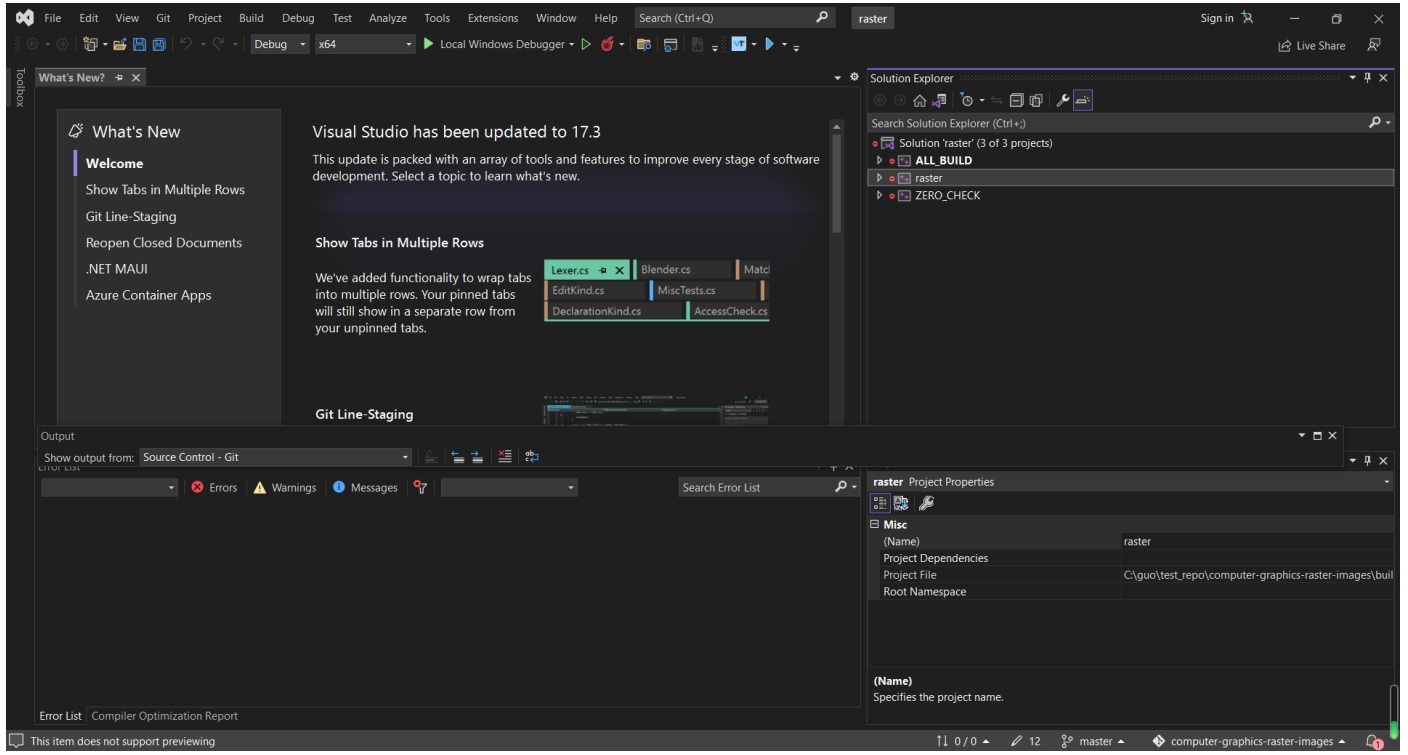
- Compilation and Running (Windows):



After “cmake ..”

Assignment 1: Raster Images

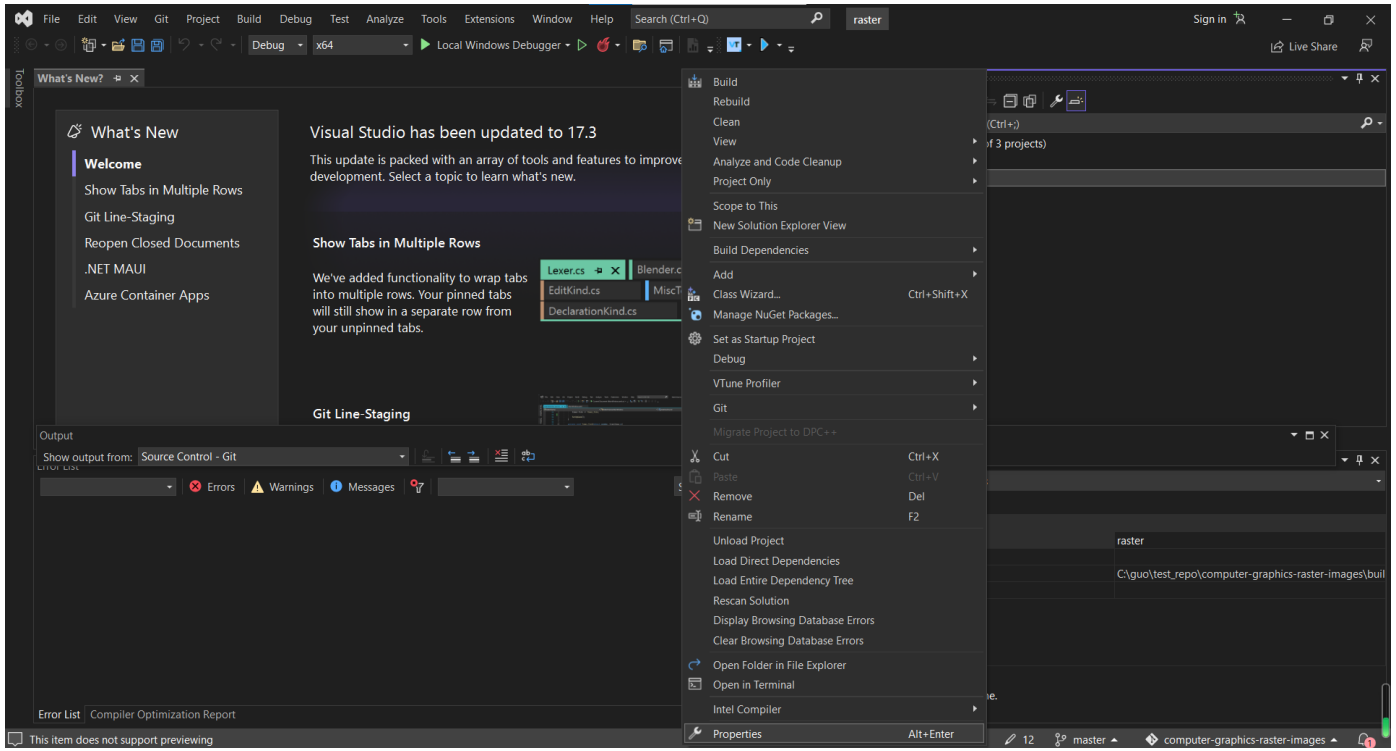
- Compilation and Running (Windows):



Visual Studio solution file

Assignment 1: Raster Images

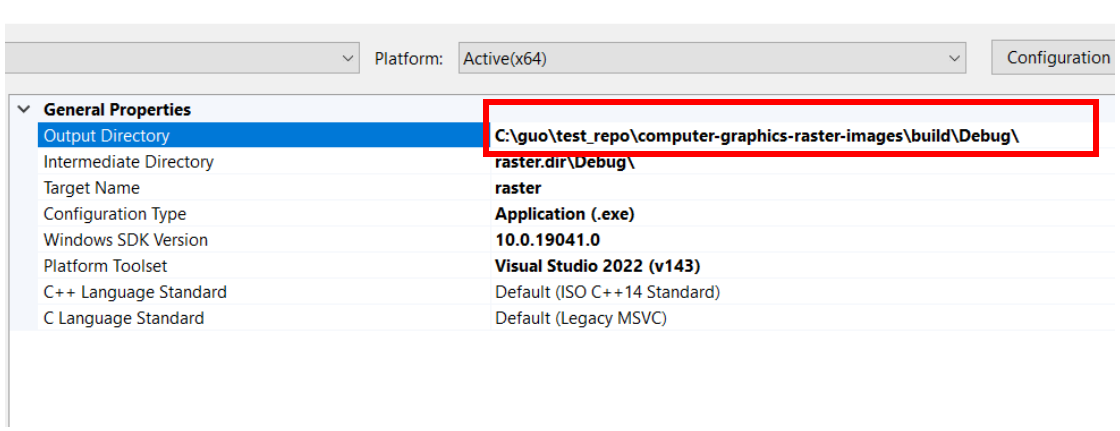
- Compilation and Running (Windows):

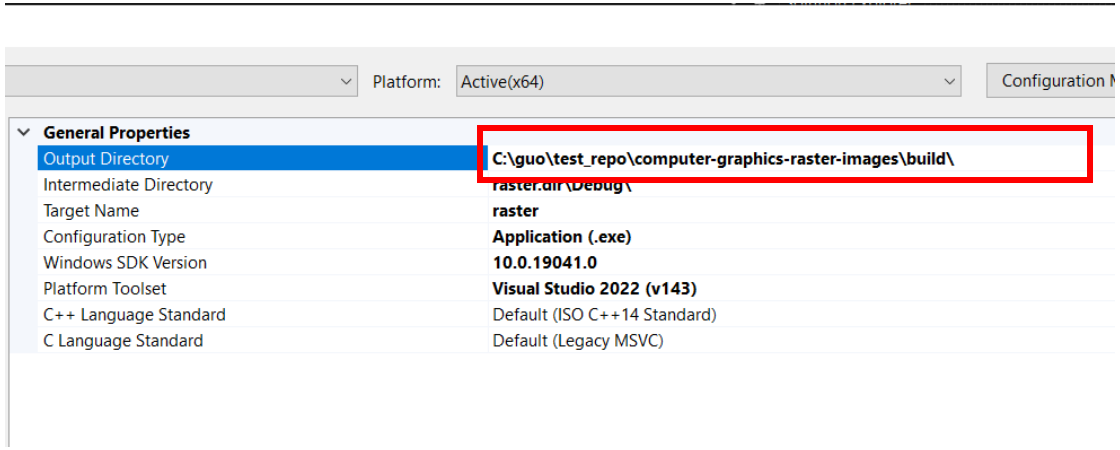


Change output path

Assignment 1: Raster Images

- Compilation and Running (Windows):

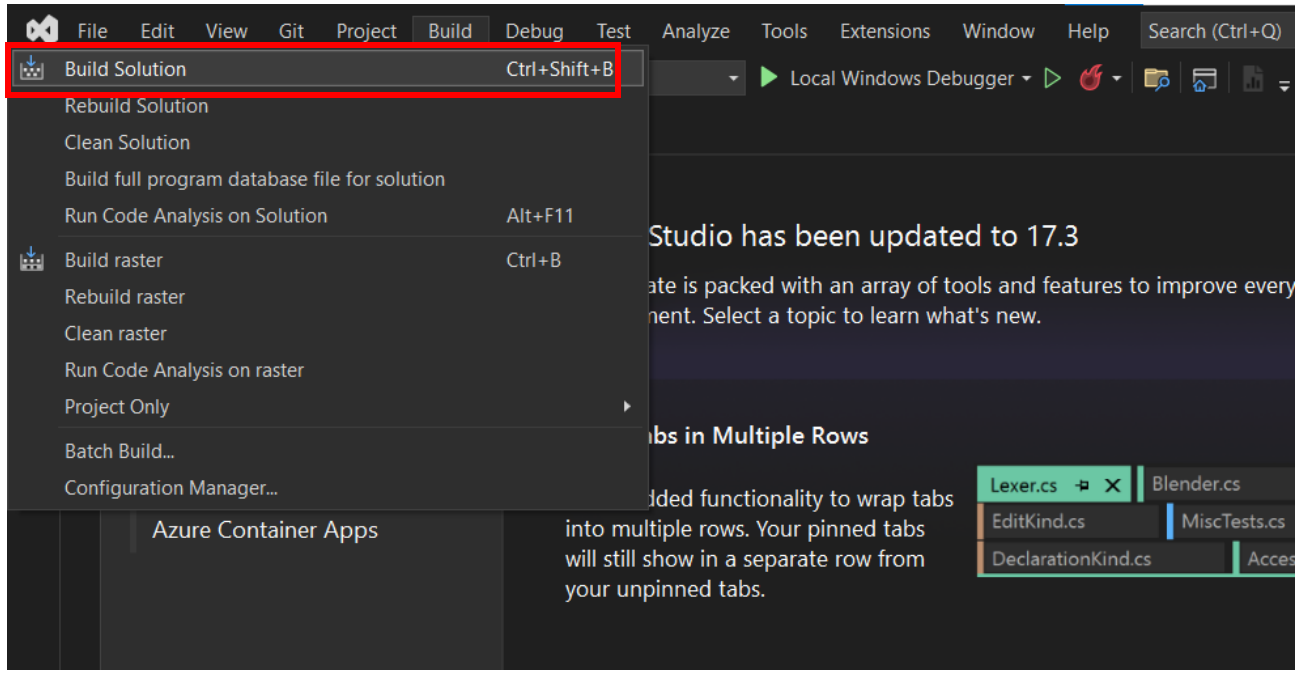




Change output path

Assignment 1: Raster Images

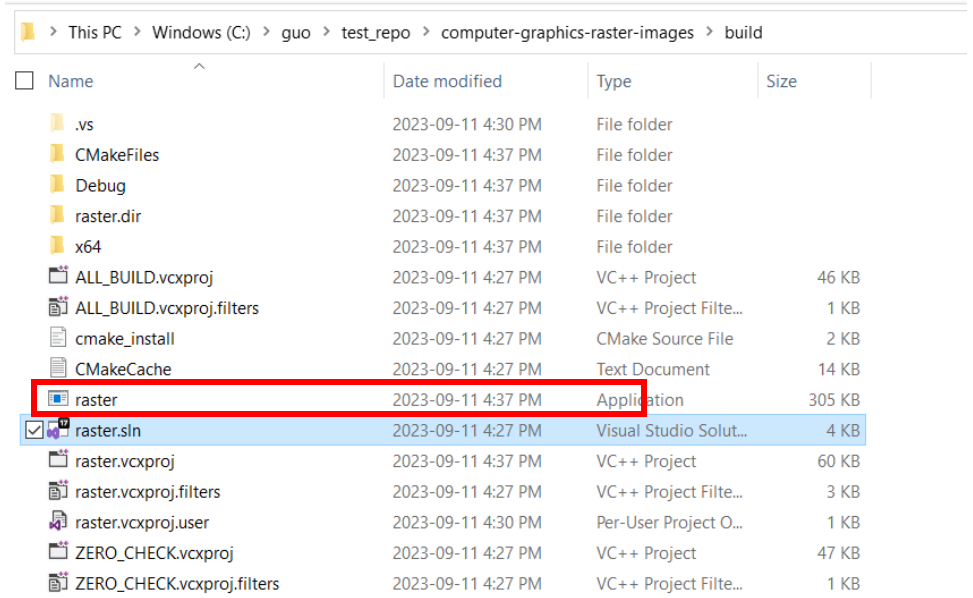
- Compilation and Running (Windows):



Build

Assignment 1: Raster Images

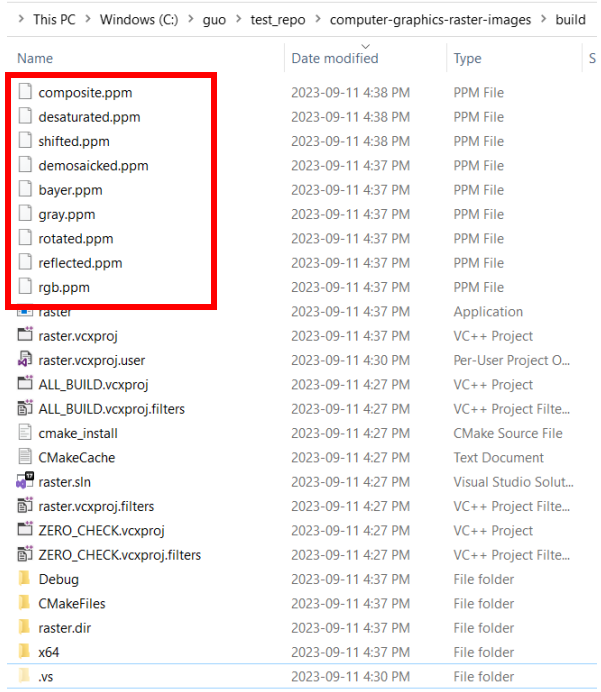
- Compilation and Running (Windows):



Run executable

Assignment 1: Raster Images

- Compilation and Running (Windows):



Check outputs