**< Important Note >**

***All CPE 212 projects are automatically graded in order to provided you timely feedback.  So, it is critical that you follow all directions for the preparation and submission of your projects for grading.  Failure to follow the directions may result in zero credit (0 points).***

**<Project01 Goals>**

A fundamental software engineering skill is the design, implementation, and testing of a software component that may be integrated into a larger software product.  In order to do this, the software component must conform to a previously agreed upon interface format.  As part of this assignment, you will practice this skill by writing several functions that will be integrated into a larger software product provided by the instructor.  Along the way you will review basic C++ programming skills required for successful completion of CPE 212.

**<Project01 Description>**

For this project, you will ***complete*** the provided partial C++ program by implementing ***six functions*** that perform simple image processing operations on images stored as 10x10, two dimensional arrays of integers.  The image processing operations are:

(1)  Load Image

(2)  Horizontal Flip

(3)  Vertical Flip

(4)  Transpose

(5)  Rotate 90° Clockwise

(6)  Rotate 90° Counter Clockwise

You must implement each of these operations as a C++ function.  ***Your code for the six functions above must appear in the file named*** **project01.cpp** ***in order to compile correctly.***  Remember, spelling and capitalization count in Linux/C++.

The function **main()** has already been implemented for you to provide a common way for everyone to test their code.   The function **main()** will scan a sample input file and perform the requested series of image processing operations by invoking your functions as needed.

Prototypes for the six functions are already provided for you in **main.cpp (do not modify main.cpp !!!).    All program output is performed by the code in the main.cpp file – do not include any output statements in the file** **project01.cpp**

***The interfaces to each of these functions you must write are described in detail on subsequent pages and in the prototypes listed within main.cpp***

***Hint:   Match*** ***the output of your program to that of the sample solution!!!***

***Directions for running the sample solution appear on the following page.***

**Running the Sample Solution on blackhawk**

***The best description of what your code should do is the Sample Solution for the project.***

Run the sample solution by typing the following at **blackhawk** terminal window command prompt

**/home/work/cpe212/project01/p01   imagefilename**

**NOTE: You may get a permission denied error running the above line.  In that case use this copy of the executable:**[**p01-ref**](https://uah.instructure.com/courses/57571/files/5421976/download?wrap=1)[**download**](https://uah.instructure.com/courses/57571/files/5421976/download?download_frd=1)**. Copy p01-ref to your working directory and type "./p01-ref p01input1.txt". Try it with all the input files.**

where **imagefilename** is the name of one of the provided input files (for example, ***p01input1.txt***).  ***Your current working directory must contain the input files for this to work.***

**Running the Preview Script on blackhawk**

Run the preview script by typing the following in a **blackhawk** terminal window command prompt

**/home/work/cpe212data/project01/preview01.bash**

**NOTE: You may get a permission denied error.  In that case use this copy of the preview script:**[**preview01.bash**](https://uah.instructure.com/courses/57571/files/5421977/download?wrap=1)[**download**](https://uah.instructure.com/courses/57571/files/5421977/download?download_frd=1)**.  Run this version without arguments to see how to use it.**

**Usage:**

**1. Copy preview01.bash to your working directory.**

**2. Create a directory called "tests".**

**3. Copy all the p01input#.txt files to the tests directory.**

**4. Type "./preview01.bash ./project01 .  p01input1.txt". Try it with all the input files.**

**Also, this new version is different than the /home/work/cpe212data/project01/preview01.bash.  The latter takes no arguments.**

**This script will run both the Sample Solution AND your project01 executable program on the complete set of input files, and it compares the outputs of the two programs line by line to identify errors in your program's outputs.  Make sure that the output of your program exactly matches the output of the Sample Solution.**

**<Project01 Compilation Instructions>**

This project consists of two C++ files – one named **main.cpp** provided by the instructor and one named **project01.cpp** written by you, along with a file named **makefile** to help you compile your program.  A **makefile** contains the sequence of commands required to compile and assemble (link) your executable program.  The provided file works on **blackhawk**.  If you examine the structure of **main.cpp**, you will see an extra **#include** directive near the end of the file – this statement will cause the preprocessor to import the function definitions you have written and saved in the **project01.cpp** file.  So, for this assignment, all you must do to compile the program is to use the following command at the Linux command line

**make**

which will create an executable named  ***project01*** from the provided **main.cpp** and the **project01.cpp** file you have written.  While ***this is not a standard use of or location for a*** **#include** statement, it will help facilitate automatic grading of your submission.

If your program compiled successfully, you may then type

**./project01   NameOfInputFile**

to execute your program assuming that the input file is located in the same directory with the executable.   (For example,   **./project01  p01input1.txt**  )

Remember, submit only the **project01.cpp** file for grading.

**<Important Project01 Submission Instructions>**

You will be provided with a C++ file named ***main.cpp*** which contains the **main()** function of the program.  The function **main()** contains code that invokes several supporting functions that you must implement.  The support functions you will write must be placed in a file named ***project01.cpp***   [lowercase letters, no spaces] in order for your program to compile and execute.

Important directions you must follow:

* ***Do not modify the file named project01.cpp!!!***

***Failure to satisfy this requirement will result in zero credit (0 points) on this assignment.***

* **All of your work (the support functions) must be placed in a file named** ***project01.cpp.*All lowercase letters in the filename with no spaces.**

***Failure to satisfy this requirement will result in zero credit (0 points) on this assignment.***

* ***All output to the monitor*** (***stdout***) ***will be performed by the code provided in*** ***project01.cpp.***

***Your program must be fully commented (including variable and parameter declarations, functions, logical blocks)  in order to receive debugging assistance from the instructor and teaching assistants***.

***NOTE:  Your project submission will be automatically graded so it is EXTREMELY IMPORTANT that you READ and FOLLOW the project directions.***

**Submit only  project01.cpp to the Canvas assignment.**

**Submisssions by email receive ZERO CREDIT (0 points)!!!**

**Submissions that do not compile receive ZERO CREDIT (0 points)!!!**

**<Project01 Hints>**

Use the function prototypes appearing in **main.cpp** to create a file named **project01.cpp** that contains empty function definitions.  Be sure that you can successfully use ***make*** to compile this skeleton solution before adding any additional code.

The **LoadImage** function is critical – **if this function does not work, then your project will fail every test**.  Make sure that **LoadImage** works before continuing with any other functions.

After **LoadImage**, implement the simplest operations **FlipHorizontal, FlipVertical,**and**Transpose** ***one at a time***, testing each with the appropriate input file.  Rotations may be implemented by  combinations of the three simplest operations.

**<Project01 Support Function Specifications>**

void RotateCW(int image[MAXROWS][MAXCOLS]);

// RotateCW() - must rotate the image 90 degrees clockwise.

// Parameter image is a two-dimensional array of integers representing the image

void RotateCCW(int image[MAXROWS][MAXCOLS]);

// RotateCCW() - must rotate the image 90 degrees counter clockwise.

// Parameter image is a two-dimensional array of integers representing the image

void Transpose(int image[MAXROWS][MAXCOLS]);

// Transpose() - must flip the image across the primary diagonal row = column as

// with a matrix transpose

// Parameter image is a two-dimensional array of integers representing the image

**<Project01 Sample 10 x 10 Image File>**

**# p01image3.txt -- Sample Image #3**

**0 0 0 0 0 0 0 0 0 1**

**0 0 0 0 0 0 0 0 0 0**

**0 0 0 0 0 0 0 0 0 0**

**0 0 0 0 0 0 0 0 0 0**

**0 0 0 0 0 0 0 0 0 0**

**0 0 0 0 0 0 0 0 0 0**

**0 0 0 0 0 0 0 0 0 0**

**0 0 1 0 0 0 0 0 0 0**

**0 1 1 0 0 0 0 0 0 0**

**1 1 1 0 0 0 0 0 0 0**