CS 201 Homework 7

This assignment is focused on *Graphics* and *File Streams.*

Grading

- (Pass/Fail) Programs compile and run
- (Pass/Fail) GitHub Project with link to GitHub repository
- (Pass/Fail) Overleaf Report written in LaTeX
- (Pass/Fail) Report submitted on Blackboard as a PDF (lastname.firstname-hwN.pdf)
- (50 pts) Report contains *design* paragraph (~100 words)
- (50 pts) Report contains *post mortem* paragraph (~100 words)
- (50 pts) Report questions
- (50 pts) Report contains *sample run*
- (50 pts) Source Code has *Makefile* or Visual Solution *.sln*
- (50 pts) Source Code is neat and documented
  - Name, Class, Date and Description at top
  - All functions and classes have documentation (besides `main()`)
- (100 pts) 2 of the *additional programs* are completed
- (100 pts) GitHub frequent commits (use program name in your text)

General Requirements

Programs that do not compile will not be graded; there is no point in turning in code that does not compile. For full credit, turn in only what was requested - no project files, executables, etc. As always:

- Code should compile, execute and work correctly.
- It should be clear to the user what the program is doing and what the user is expected to do.
- The source code should begin with comments indicating the source file's filename, author, date, and purpose.
- The source code should be neat and readable.
- Functions should be commented with their purpose.
- Other comments should be used anywhere things might not be completely clear.

Questions

- What is a *file*? What does it mean to *open* a file?
- What should a program *always* do immediately after attempting to open a file?
- When a file has been opened, it will eventually need to be closed. But when we use C++ file streams, we mostly do not need to worry about closing files. Why not?
- Why is error checking more important when doing input than it is when doing output?
- Why should you only check for end-of-file after there is an error on the stream?
- What is a *pointer*? In the context of pointers, what is an *address*? What is a *null pointer* and why do null pointers exist?
- Arrays are built into C++. Why do we prefer `vector`?

Main Program (asciiart.cpp)

Write a C++ program that uses Portable Pixmap Format (PPM) images. The Portable Pixmap Format (PPM) is a very easy-to-use image format that can be opened natively on many operating systems or with IrfanView or GIMP on Microsoft Windows. Detailed information is at https://en.wikipedia.org/wiki/Netpbm, but the basic layout is as follows:

```
P3
3 2
255
# The part above is the header
# "P3" means this is a RGB color image in ASCII
# "3 2" is the width and height of the image in pixels
# "255" is the maximum value for each color
# The part below is image data: RGB triplets
255   0   0  # red
  0 255   0  # green
  0   0 255  # blue
255 255   0  # yellow
255 255 255  # white
  0   0   0  # black
```

Write a program that reads a PPM image and outputs an ASCII Art version of that image similar to:

```
                        .+@*-+..            ..       .-.. ...---       ---.
                     .0#0@0#####00...    .+@@0*..  .- .  .... .        .+-. .
                  .-0#000####0###0###0@##########0@@@    .-.                  ..
                .*@000000###0000####################0-++ ...           .  .   ..
               ++**@@@000000000#####################0###0. .           ..     ..
             .++++****@@@00@@00#############0#######00000. ..            .   ...
             ++---++*@@@@@@*@@00000#######0000####00000000@.                .. .
             ------+**@*******@0@@0000000##0000##0000#0#####-       .-   .-..+--
-            +-.-.--++*+@@@0@@@@@*@@@0000#0#@@@@000#0#######0.  .   -*+---.-.--.
@            -..-..--***@00##0@@**@00000000@@0@@0@000########0...   .+++**--....
              .+.-..+@@00000##00@@@@000@00@@@@*++**@00####@00##-...--**+@@*.--
              *0+.-.+0000*.00###00@@@@@@00@@@@****@@@*@##0*000#0  ..--++*@**@*..
              ##0-..-0000@@0#00#00@00@@00@@@@@@@@0@@0@@@@@@000#-.   ..-+++*@*+.
    ..  .    +##00..-@000000@@00#00000000@0@@@0000##00@*@@00000-.. ..  .--+*-.
     . . ..  0###0@.-00000000####0000@@@@@@0000000@*+*@@@@@@@00@0@  .   ..-..-..
    ..    .. #####+..@0#########00@00@@@0@@@@@@000@*+*@@*@@*@00##0  .. ...  +*-*
            .#0###+..+#00#######0@00@@@000@*@@@@@*++**@@@@@@@0##0000@......-**@0
            .#00##*.-*000#####0#0@0@000000@@@@@*+++++***@@@@@0##0#0@*.--..-.**@@
            .00000+-.-@########0000@@00000@00****@++++*@@*@@@@0####00@@@@.--+*+-
            .0000*..  *########000@@@@@@0@*@0***@@*++*@@****0###0000**00#@++*+--
             0000+-   -##########000@@@@**+*@**@@@@**@@+***@00##0*@+-@@###@-++-+
             *00@+.+-.-#######0#00@@@***++******@@***@+****++0###**--**00#+*...+
              @@*---..+*@###000000@@@***++*+***@@++**++****++@###0*+-++@@@@*-++*
         -    *@*----+--.-0#0000@@@@*+****+*+***@*.-+*+++*@++*0##0@++++*@@@0@+-*
         ..   -**-+--.-....-@@@*@@@@@*+****+++***---+*+**@0@*@@0###*++-**@#+#-..
         .-   .--+*-. .......+********++**++.-+++-..-+*@*@00**@@0#00*++*+**@*@..
          ..  -@**++- .. .....-*+*+*@*++++++.---+-..-+@@*@@@@**@000@@++**@0*@**.
-++++.     ..+@@@*+.--. ..    -*+*++*+---.+*---+++.-++@@***@****@0**+-+@**@***+-
0@@@**     .*00@***--+-. .....++-*++++-+--++-.-++---+**@*++**+++**@**--+**@0****
0@@@@@.   .@0#@@***+++*+.....-*+.+++**.-+-.-++-++--++*-+++++*+++*****-.+++*@**@@
*****-    @@@@00*+-+*+++......+. -*+*+.-+-..++-+---++*+--++++++++****+.+++@****0
**+      +000@0*+*+-+*-++-...    .----..+.- .+--.---+++.------++*****+++*+++**++
**@.     0*++0**+++--++.--....  . --.-............----.....---++****+**-+-+++*@.
++-     +++++@++-+*+.-+-......... -..-................----+-++++++*++*+--*+++*0.
  .-.   *0@-++----*+-... ........ .-.....  .....-.....--++++++++++++++++-+@-+*@.
```

- The R, G and B values will vary between 0 and 255
- Convert color to grayscale using the formula Y = 0.2126R + 0.7152G + 0.0722B
- Divide Y [0...255] by 16 to reduce the range to [0..15]
- Create an array of chars to hold 16 characters that range from dark to light (e.g. const char values[] = “@.............. “ - replace . with appropriate characters)
- Test your program with Parrot.ppm and include the output in your report

Additional Program 1 (ppm2pgm.cpp)

Modify asciiart.cpp to output a PGM image. Reuse and refactor your code from the main program as much as possible. Include the grayscale image in your report.

Additional Program 2 (rule30.cpp)

Write a program to evaluate Rule 30, which is a cellular automaton discovered by Stephen Wolfram. Use the int argc and char** argv arguments in your main() function to take two optional integers to specify the number of columns and rows to process. The default should be 40 columns and 20 rows. This kind of cellular automata is initialized by setting the middle cell to 1 and all other cells to 0. Then you will evaluate every cell according to these rules and formula:

Current	111	110	101	100	011	010	001	000
New	0	0	0	1	1	1	1	0

It is called Rule 30 because if you convert binary 00011110 to decimal, you get 30. You should read from cells i-1, i and i+1. Which we will call left, center and right. Assume that a cell out of range is a 0. Apply the formula

New = Left XOR (Center OR Right)

You will need a `vector<int>` for Current and one for New to get correct results. After each iteration, print out the vector to a file called rule30.txt. Include sample output in your report.

Refer to https://en.wikipedia.org/wiki/Rule_30 for more information.

Additional Program 3 (caesar-cypher.cpp)

A **Caesar Cypher** is a simple encryption/decryption method that shifts letters by some given number of steps. For example, a shift of 1 takes A to B, B to C, and so on. It takes Y to Z, while Z rotates around to A. A shift of 2 takes A to C, B to D, and so on. It takes W to Y, X to Z, Y to A, and Z to B. Only letters are modified. So, a Caesar Cypher with a shift of 3, applied to the string “Hello, everyone!” would become “Khoor, hyubrqh!”.

Write a C++ program that repeatedly inputs a line of text and an integer used a shift. It should apply the Caesar Cypher with that shift to the text and output the result. There should be some way for the user to end the program and proper error handling should be done.