CS430 Homework Assignment 2

Programming Problem:

1. The new Postscript commands you will be required to implement are:
   1. x y moveto
   2. x y lineto
   3. stroke

moveto command simply moves the pen to a particular location without drawing anything.
The *lineto* command draws a line from the current position of the pen, to the point indicated.
The *stroke* command can be ignored in your program, but actually draws the line sequence in Postscript interpreters (e.g. gv).

The following commands will draw two boxes.

```
%%%BEGIN
100 100 moveto
200 100 lineto
200 200 lineto
100 200 lineto
100 100 lineto
stroke

300 300 moveto
400 300 lineto
400 400 lineto
300 400 lineto
300 300 lineto
stroke
%%%END
```

You can assume that a "moveto"/"stroke" pair defines a single closed polygon with no holes and with vertices given in counter-clockwise order.
Also that the input file will contain only one polygon.

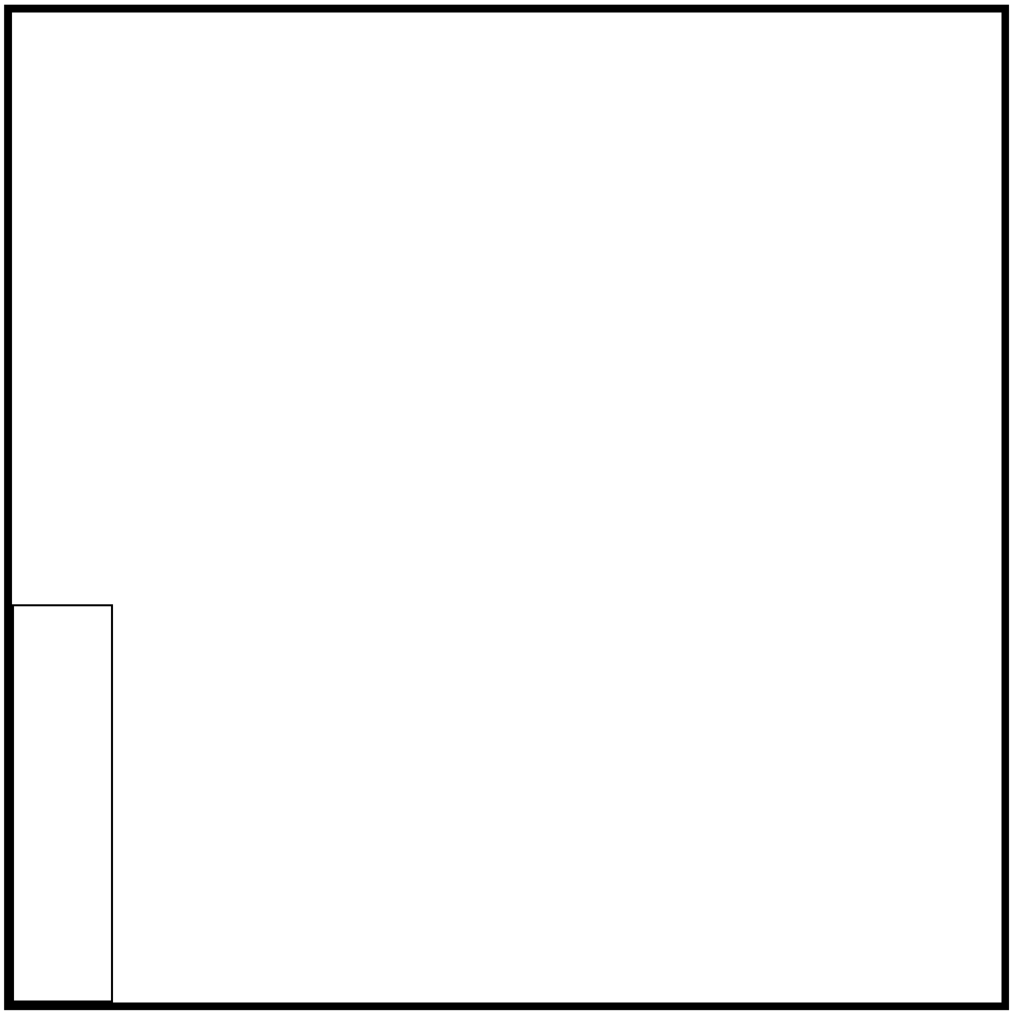
2. Modify your HW1 program to:

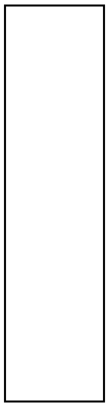
   1. Accept the "Moveto", "Lineto" and "Stroke" Postscript commands.
   2. All the command-line options implemented in HW1 should be supported.
      - The new default value for the "-f" option should be "hw2_a.ps".
   3. Read in a polygon from a file, apply 2D transformations to it, clip it against the world window using the Sutherland-Hodgman algorithm.
   4. Display the polygon edges with the code implemented in HW1.
   5. Your new program should be named CG_hw2.
   6. You can test your program with several files: hw2_a.ps, hw2_b.ps, hw2_c.ps, hw2_d.ps.

   7. Input/Output Example: (Dark border is the window boundary)
      i. Input: `./CG_hw2 -f hw2_a.ps -s 1.5 > hw2_ex1.pbm`   ii. Output:

8. Input/Output Example: (Dark border is the window boundary)

i. Input: ./CG_hw2 -f hw2_a.ps -m -250 -n -200 > hw2_ex2.pbm ii. Output:

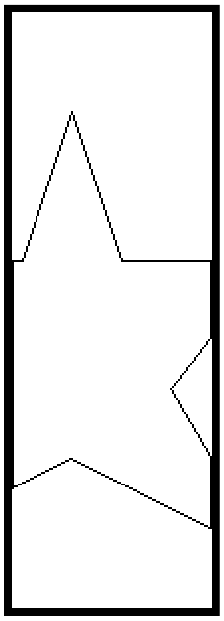


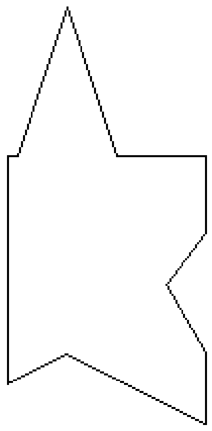


9. Input/Output Example: (Dark border is the window boundary)

i. Input: ./CG_hw2 -f hw2_b.ps -a 170 -b 100 -c 270 -d 400 > hw2_ex3.pbm

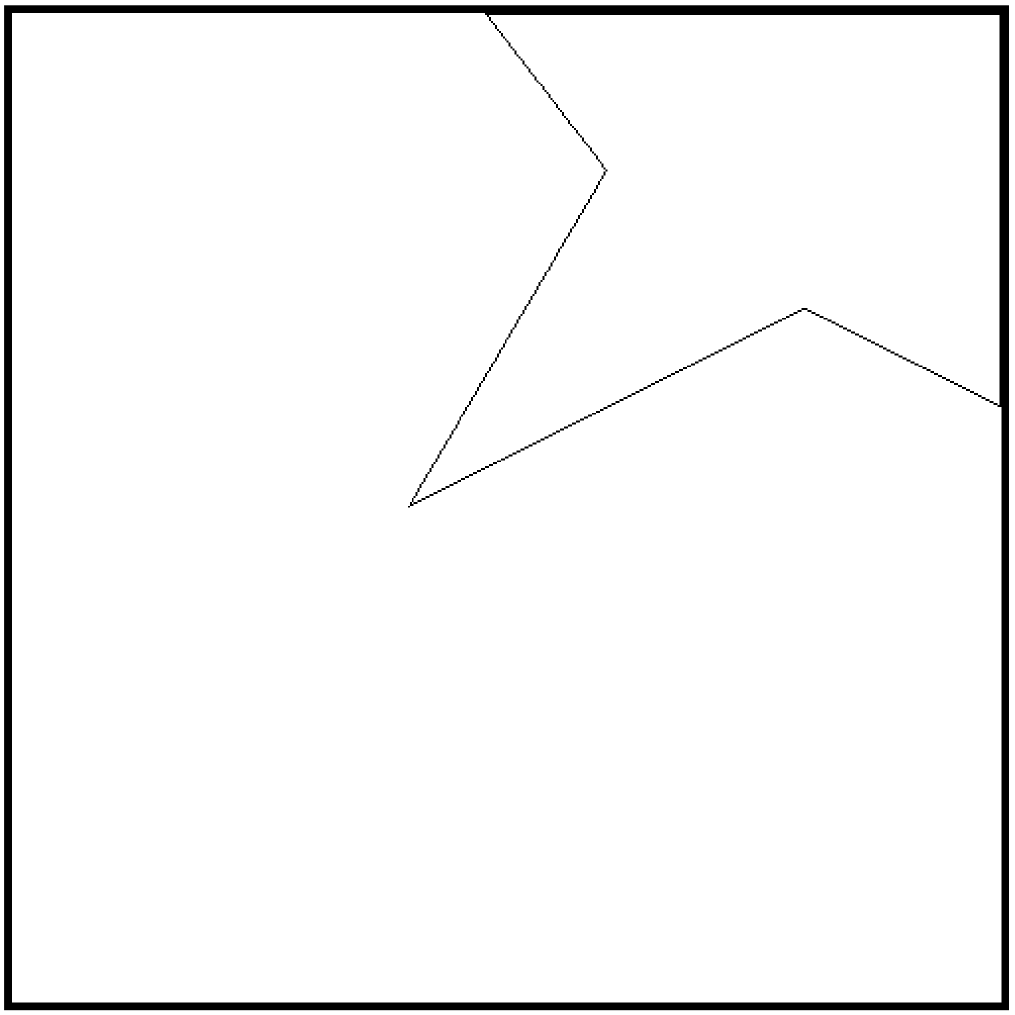
ii. Output:

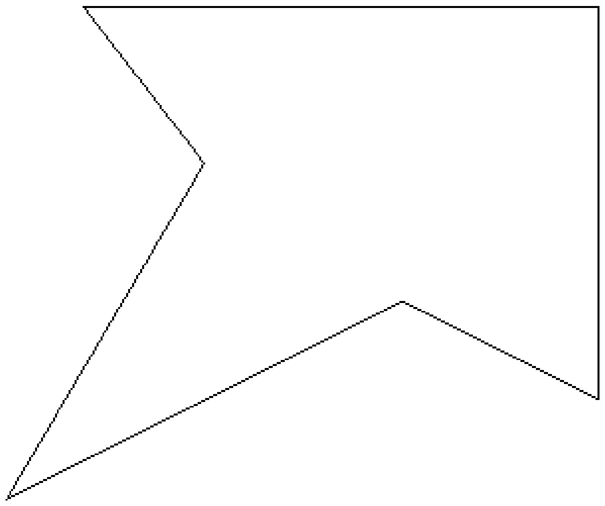




10. Input/Output Example: (Dark border is the window boundary)

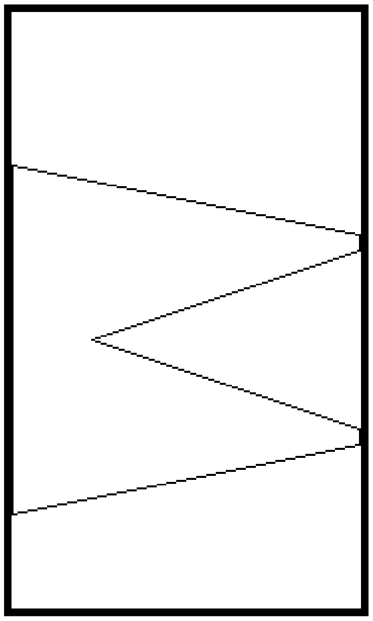
i. Input: ./CG_hw2 -f hw2_b.ps -s 2 > hw2_ex4.pbm ii. Output:

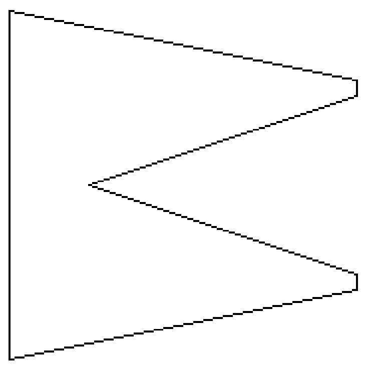




11. Input/Output Example: (Dark border is the window boundary)

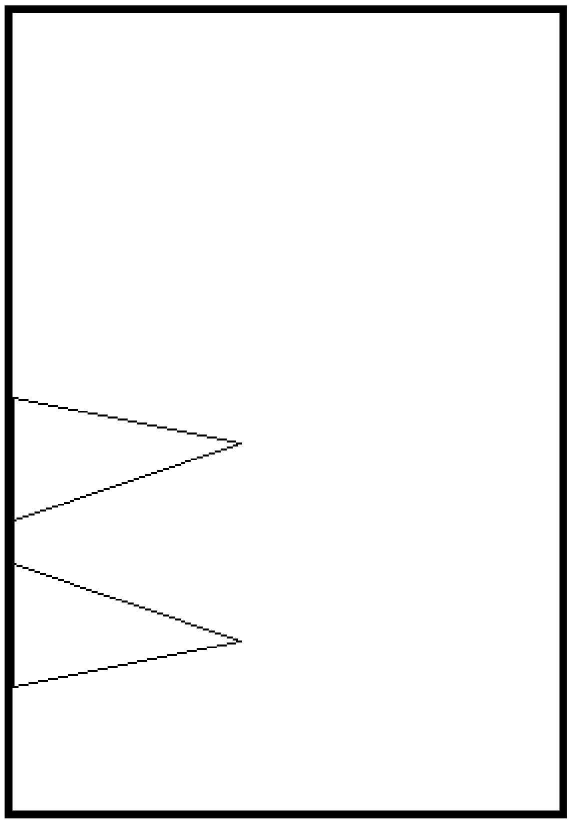
i. Input: ./CG_hw2 -f hw2_c.ps -a 200 -b 100 -c 375 -d 400 > hw2_ex5.pbm   ii. Output:

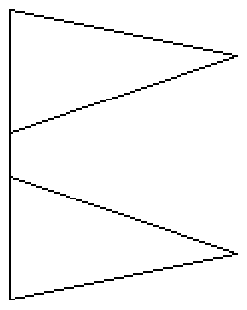




12. Input/Output Example: (Dark border is the window boundary)

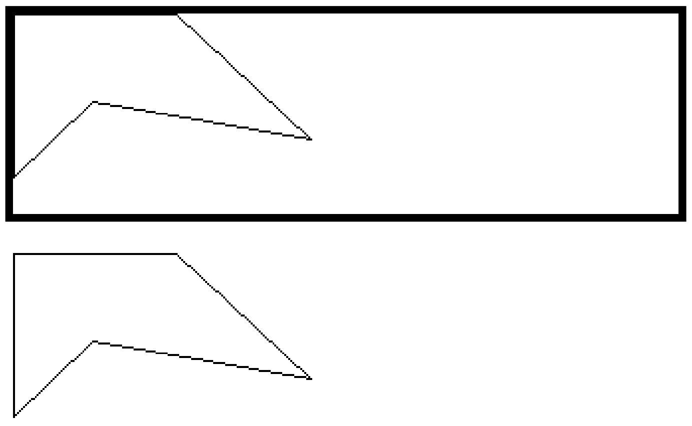
i. Input: ./CG_hw2 -f hw2_c.ps -a 275 -b 100 -c 550 -d 502 > hw2_ex6.pbm   ii. Output:





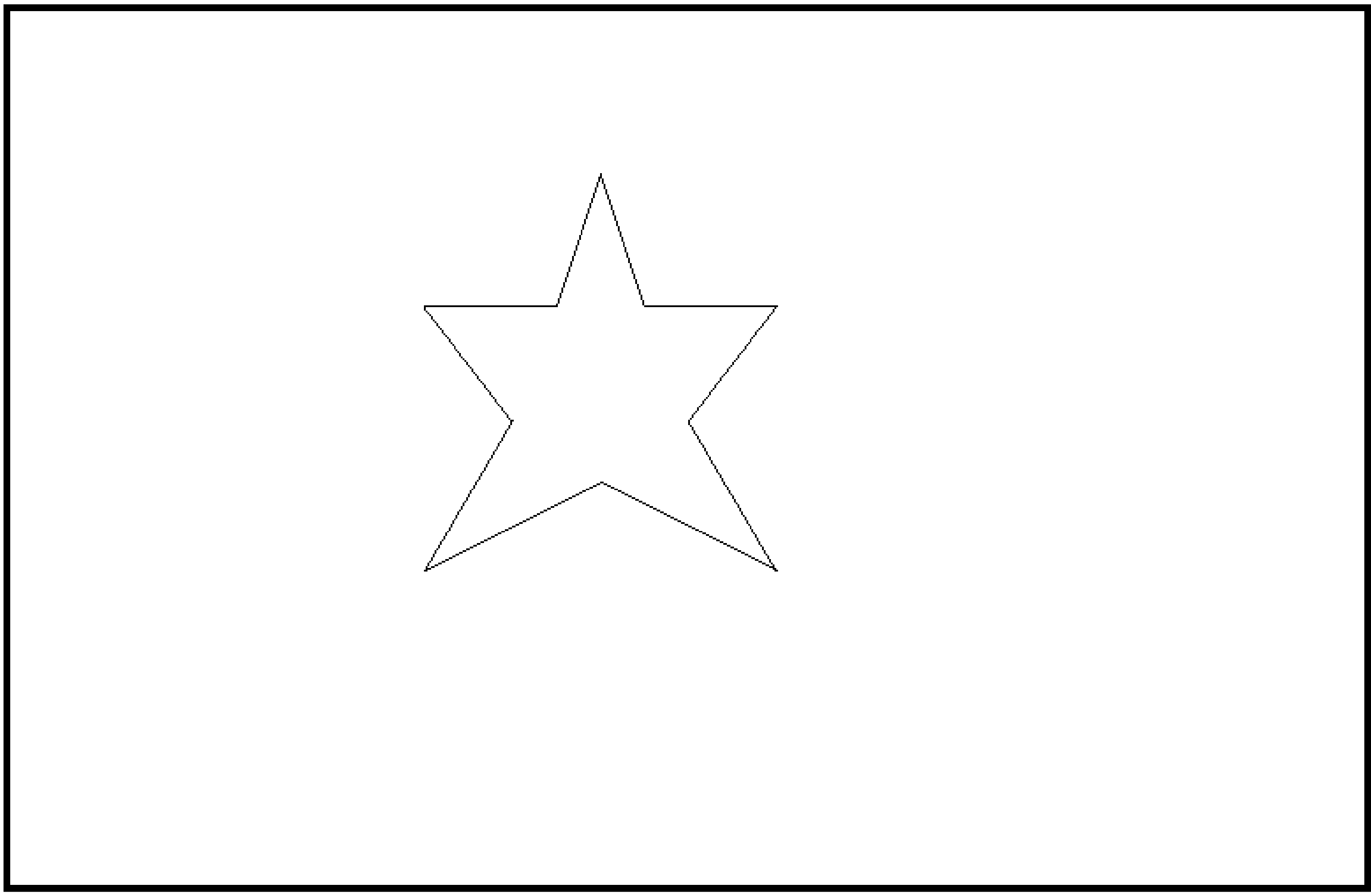
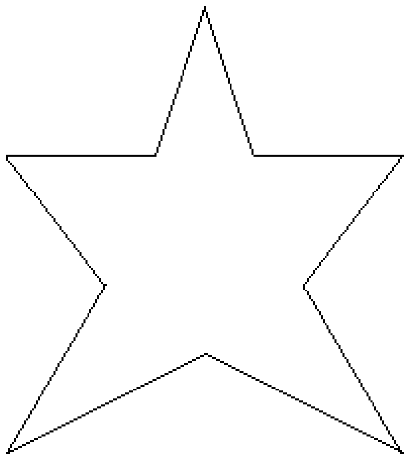
13. Input/Output Example: (Dark border is the window boundary)

i. Input: ./CG_hw2 -f hw2_b.ps -d 270 -c 435 -b 170 -a 100 -r 17 > hw2_ex7.pbm   ii. Output:

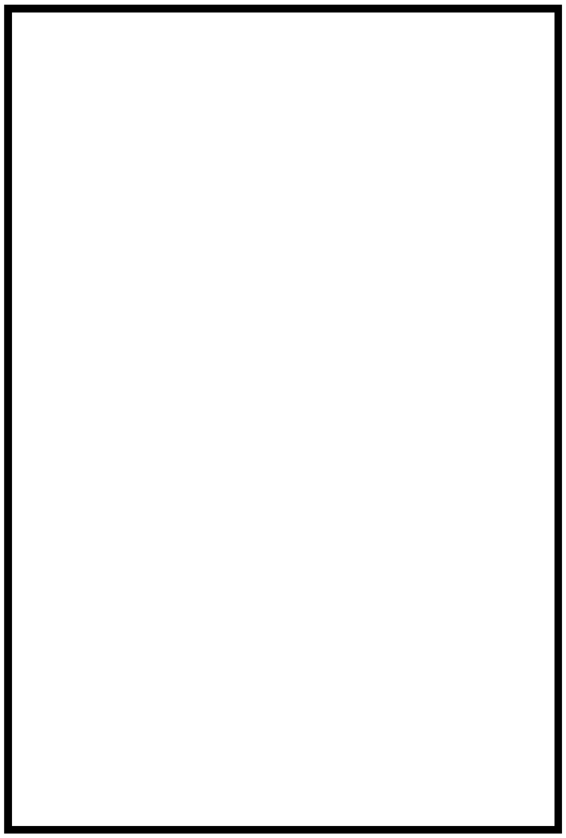


14. Input/Output Example: (Dark border is the window boundary)

i. Input: ./CG_hw2 -f hw2_b.ps -a -135 -b -53 -c 633 -d 442 > hw2_ex8.pbm   ii. Output:

15. Input/Output Example: (Dark border is the window boundary)
    i. Input: ./CG_hw2 -f hw2_c.ps -a -150 -b -475 -c 123 -d -65 > hw2_ex9.pbm   ii. Output:



3. Grading Scheme

    1. Assignment 1 features : 2 points
        - Comand-line argument reading
        - Image format
        - Line drawing
        - Transformations
        - Windowing
    2. Postscript reading : 1 point
    3. Polygon clipping : 7 points

4. Submission Guidelines:

    1. Assignments must be submitted via Bb Learn.
    2. README file: explain the features of your program, language and OS used, compiler or interpreter used, name of file containing main(), and how to compile/link your program. Text files only. Word and PDF documents will NOT be accepted.
    3. All source code. Your code must compile and run on tux (Linux).
    4. You may program in any language you like as long it can produce a usable executable named CG_hw2 on tux.
    5. Your program will be run by the TA. Please do NOT submit any image files, Visual C++ project files, or anything not requested in this section. Your program must run on tux without the installation of "special" libraries.
    6. Makefile: have the default rule compile your program.
    7. If you are using a language that doesn't produce an executable file, e.g. python, then be sure to include a script called CG_hw2 that accepts arguments and prints PBM to standard out.
    8. Points will be deducted if submission guidelines are not followed.
    9. Further details about Bb Learn
        1. You can reach Bb Learn through DrexelOne.
        2. Choose Computer Graphics among your list of courses. There is an "assignments" link in the left frame which will give you the list of assignments in the right frame.
        3. Click on the assignment you wish to submit.

4. Find your file and click Upload button.
5. Hit Submit button. DO NOT FORGET TO HIT THE SUBMIT BUTTON AFTER YOU UPLOAD ALL YOUR FILES.

NOTE: Your source code for all programming assignments will be run through a plagiarism detection system. This program uses compiler techniques which are invariant of syntax and style. If you are sharing code with other classmates, you will get caught. Please refer to the student handbook for actions that will be taken.

Last modified on September 24, 2020.