**Project 6 Part 1 Coin detection Part 1**

[Project 6 Part 1 Coin detection Part 1](https://fcps.blackboard.com/webapps/blackboard/content/listContent.jsp?course_id=_1516674_1&content_id=_43601203_1&mode=reset)

Name of file l061.cpp

Your main should call only the folllowing method:

Create part1() method that does the following:

1) read the file image.ppm (image will contain an image of coins)

2) create the file imagef.ppm by applying the complete canny edge detection from previous lab (optimize the values to work best for the easy file I provided)

3) use the edge you found and gradient direction to implement voting and create the imagev.ppm that displays the result of the phase 1 voting for centers.

4) use the results of voting and a threshold value to pick good candidates for circle centers and create imageCC.ppm file that will display the original image and on the original image will do a filled circle of radius 5  of color red for each candidate for circle center. (a filled circle of radius 5 can be obtained by drawing 5 circles with that center of radius 1,2,3,4,5)

Hints: use besenham algorithm to vote on the direction of the radiant. Also a good improvement would be to calculate the intersection points on the extremes so the line i more close to reality compared to using 2 very close points in which case the line will be off.

Submit the following document after your code runs on the easy image I provided (which you will transform in p3 ppm file using teh convert tool on gnu linux:

[Project 6 Coin Detection Part 1.docx](https://fcps.blackboard.com/bbcswebdav/pid-44785820-dt-content-rid-50508918_2/xid-50508918_2) [Project 6 Coin Detection Part 1.docx - Alternative Formats](https://fcps.blackboard.com/webapps/blackboard/content/listContent.jsp?course_id=_1516674_1&content_id=_43601203_1&mode=reset)

Your code will be tested against any piece of the easy image provided not just against the entire image.