Hitender Prakash
hprakash@iu.edu
812-361-9187

Draw a circle on console without using floating point operations

https://github.com/hitenderprakash/drawCircle

Algorithm:

Circle can be represented with equation $x^2 + y^2 = r^2$. We find all the circle points (points that lie on circumference) only in the first quadrant and calculate the other points by applying transformations(mirroring and reflection, along all the remaining three quadrants). For finding the points in first quadrant, we loop for x=0 to x=r and find the corresponding y from the above equation i.e $y_i = (r^2 - x^2)^{½}$. Since we are not allowed to use sqrt(), therefor we can find approximate value for y as given below:

1. If we can find y_i such that : $(y_i^2 + x_i^2) == r^2$, then select the y_i
2. Otherwise we find two values for y_i (y_{i1}, y_{i2}) such that $(y_{i1}^2 + x_i^2) > r^2$ and $(y_{i2}^2 + x_i^2) < r^2$, in this case we finally choose y_i which gives us minimum loss for $[r^2 - (y_i^2 + x_i^2)]$

Design:

Storing the actual/absolute points of our circle (x,y) would make operations like shifting the center or resizing the outer grid complex. I am storing the relative difference (dx,dy) from origin (O_x , O_y) so that actual point position can be calculated linearly using (O_x+dx, O_y+dy). This calculation will take constant time and will be handled by the display function itself, providing a layer of abstraction. It will not change the asymptotic complexity of display operation and hence ensures that we can make raster resizing and center shifting operations in constant time.

One point to be noted is that, we represent the Raster grid as two dimensional character vector, thus in first quadrant, difference of any point (dx,dy) would have to be transformed as (x +dx, y-dy) because in matrices y increases downward, therefor we stores:

1. First quadrant: (dx, -dy)
2. Second quadrant (-dx, -dy)
3. Third quadrant (-dx, dy)
4. Fourth quadrant (dx, dy)

How to run:

Clone the repository : https://github.com/hitenderprakash/drawCircle
On Linux machine, run “make”

In the main.cpp I have included code for few test cases, please uncomment the code for desired test case and compile again with “make” and execute the code by running “ ./drawCircle” command or you can also use “make run” command, it will do both - compile and execute.
NOTE: Each Test case is numbered and provided with a brief explanation of what it does.

Functions supported:

1. Draw circle such that it automatically fits within console
    a. Console width and length can be read dynamically and then create the circle with it’s center in the middle of the console where [radius = min(length,width)/2]. I have tested the code on Linux and Windows OS.

Hitender Prakash
hprakash@iu.edu
812-361-9187

2. Else draw circle with user provided inputs assuming they are valid
3. Resize the grid. It will not recalculate the circle points again only call the display function again
4. Relocate the center. Will not recalculate the circle points

Limitations: We cannot change the radius of the circle as it will cause the recalculation of all circle points.
Workaround: Create new object with new radius parameters

General Flow of the program:

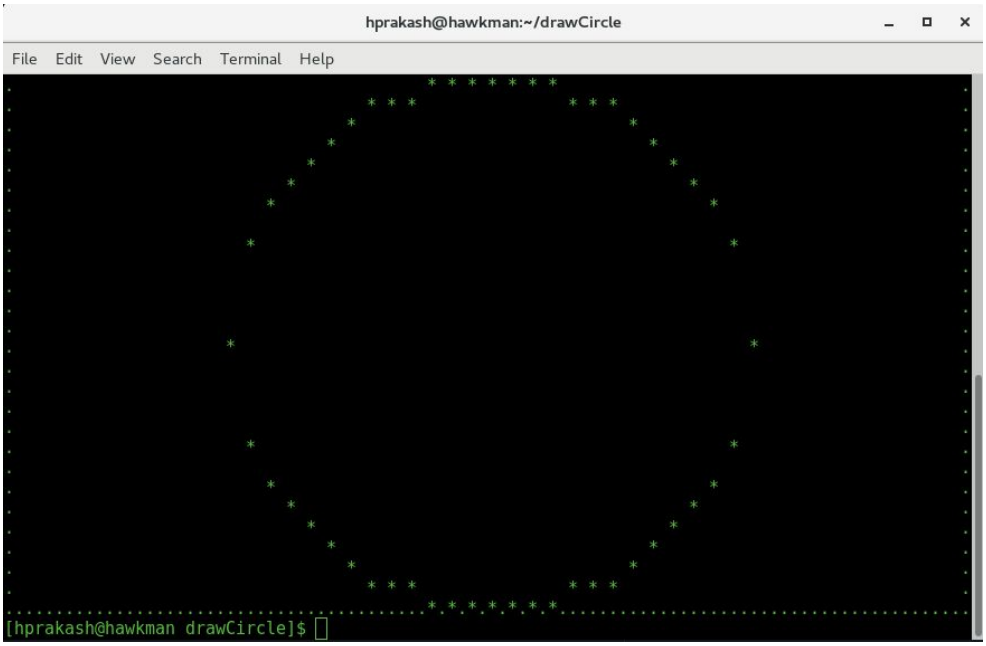
1. Create Raster Object by calling constructor **Raster()**
2. Set the origin and radius of the circle by calling **set_circle_param()** function
3. Call **drawCircle()** function (it will not display the circle on console but only calculate and store circle points
4. Call **display_Circle()** function to print the circle on console
5. Now you can call **resize_RasterDimensions()** or **relocate_Center()** to change raster grid size or shift origin of the center and call **display_Circle()** again to see the output after transformation

For example

```
Raster circle(30,20);
circle.set_circle_param(28,10,6);   //center(28,10) and radius=6
circle.drawCircle();
circle.dispay_Circle();
circle.resize_RasterDimensions(40,20); //increase the raster dimensions
circle.dispay_Circle();
```

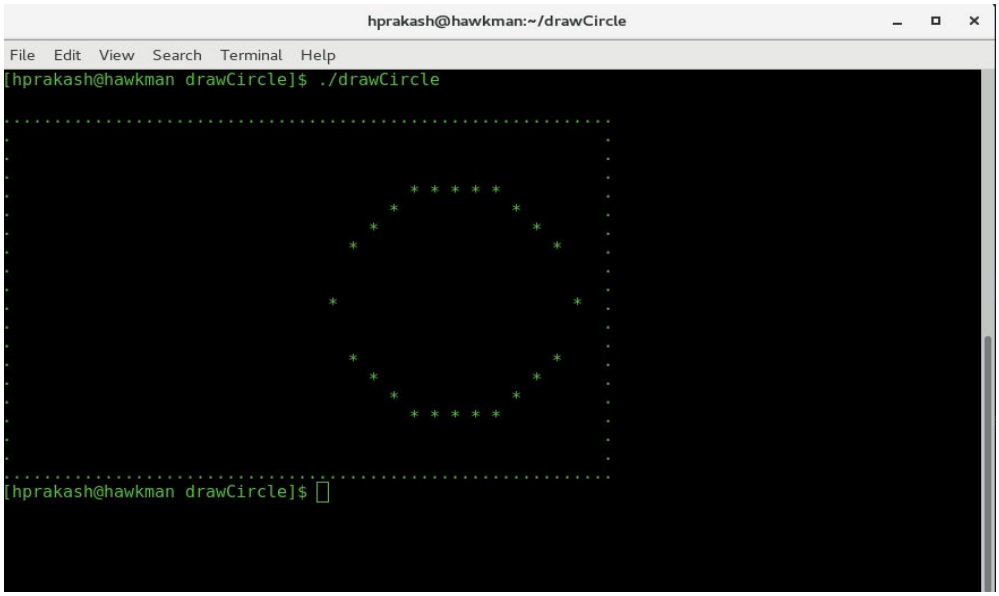
Test Runs:
Given below are the screenshots for few of the test runs I did.

Test case 1: Demo of drawing circle which automatically fits the screen

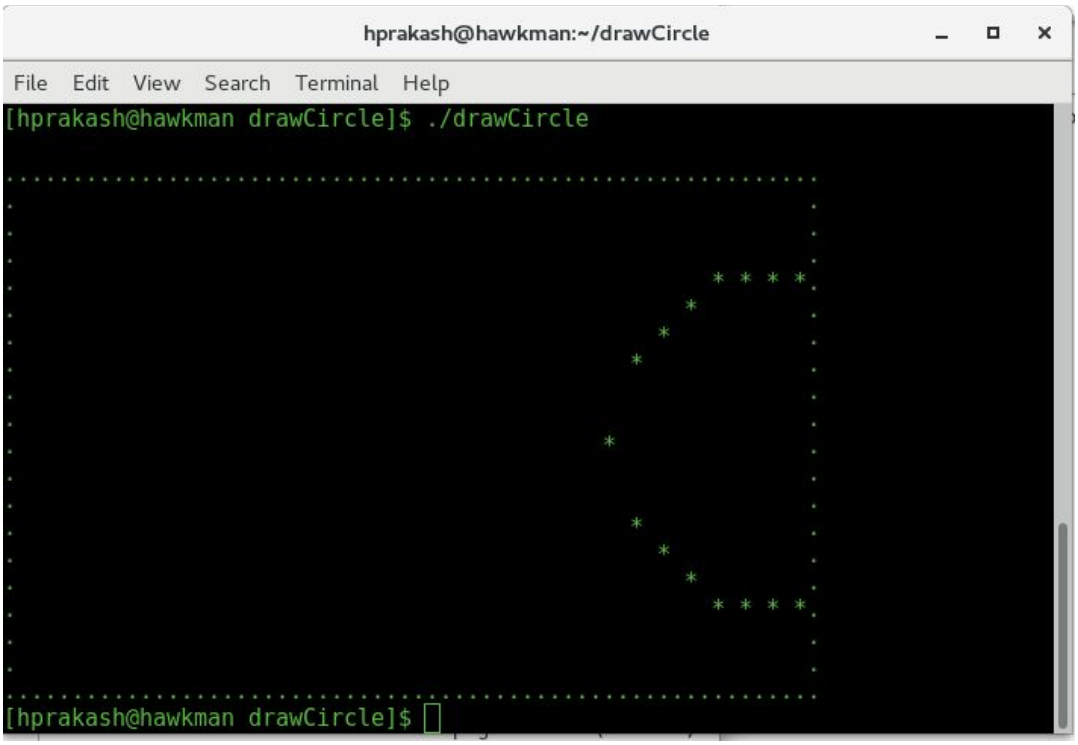


Hitender Prakash
hprakash@iu.edu
812-361-9187

Test Case 2: Drawing circle with user provided inputs for raster dimensions, center and radius

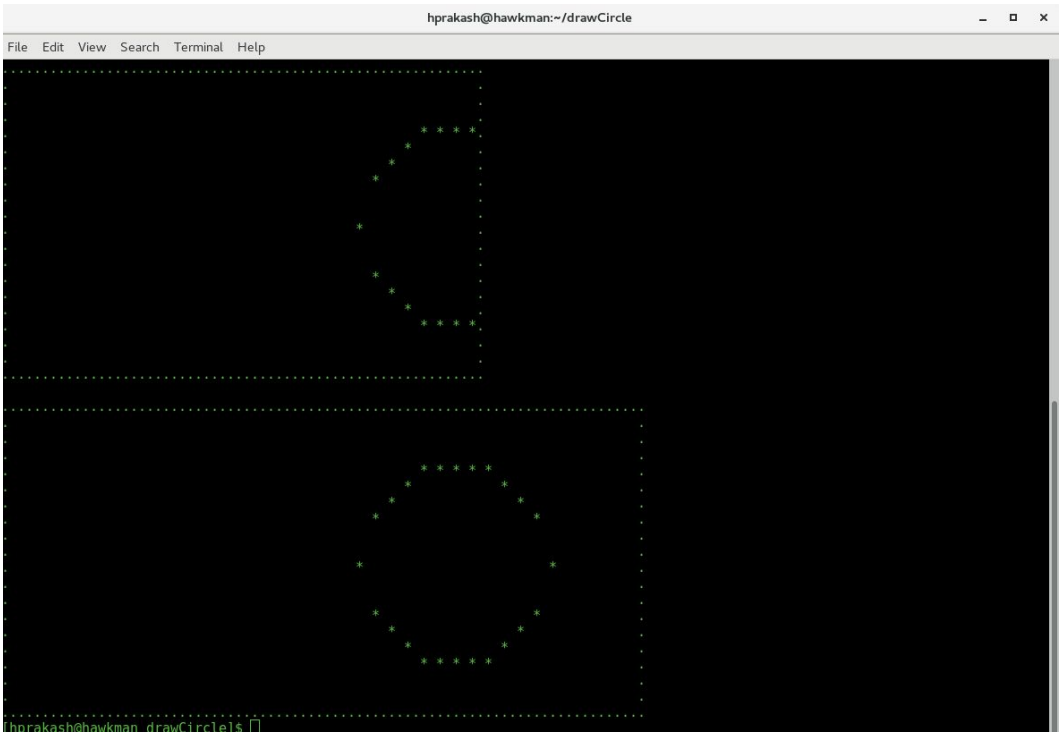


Test case 3: Draw a partial circle if it does not fit inside raster

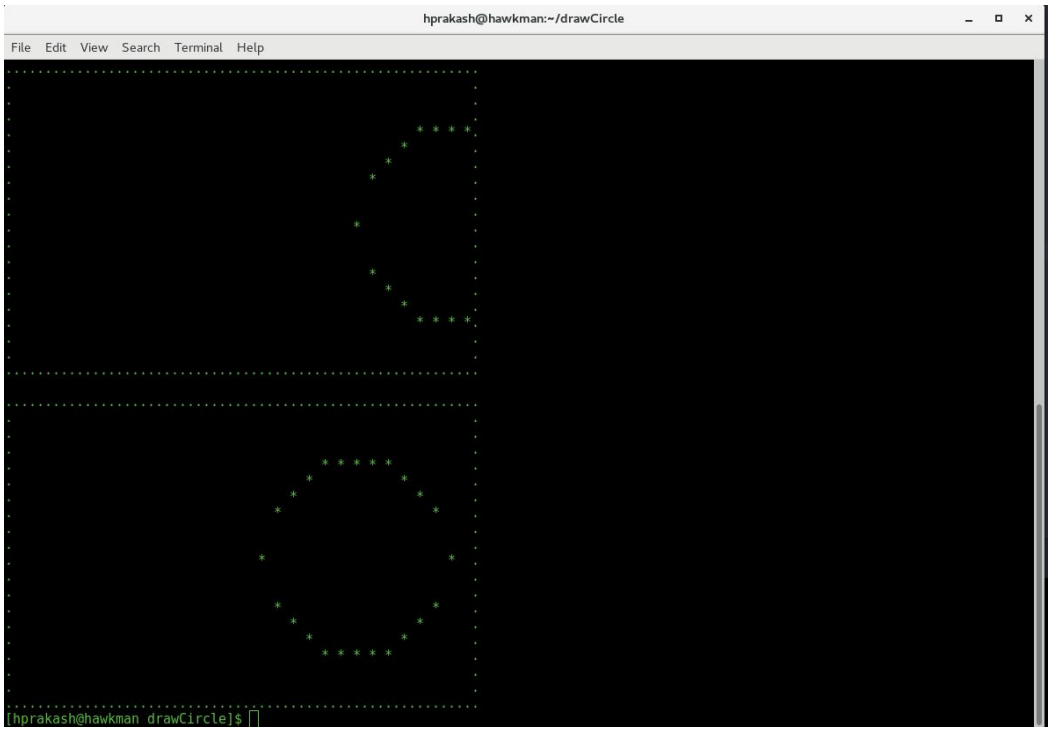


Test case 4: Raster resize. Initially circle can be partially fitted inside the raster but after raster size increase it can be fit inside the raster

Hitender Prakash
hprakash@iu.edu
812-361-9187



Test case 5: Relocate center. Initially the circle can be partially fit inside the raster. Then we move the center towards left and circle now falls inside the raster



Hitender Prakash
hprakash@iu.edu
812-361-9187

NOTE:
In current design, as per your suggestions we have enforced that the origin of the circle should lie within the raster. However we may remove this restriction because even if the center lies outside of the raster, circle may partially lie inside the raster. Making this change is easy we just need to disable “return error” if centerpoint lies outside the raster.

Each function return integer values, 0 on success and -ve on error, it makes error handling more easy. In driver code in main.cpp, however I have ignored checking these return values.

General Test Cases:

1. Provide small circle radius (shape may get distorted)
2. Give invalid raster dimensions
3. Give invalid origin coordinates
4. Give invalid radius, like 0 or -ve value
5. Increase raster size and then display the circle again
6. Decrease raster size and then display the circle again
7. Move the origin to different points within the raster (moving outside the raster is not allowed in current design as mentioned in the note above)
8. Give input such that circle partially fits inside the raster
9. Run Test Case 1 (in the main.cpp) with different window size of the console. Circle should always fit in the raster (in this case radius and origin is automatically decided by program)
10. Try displaying circle without creating it i.e call dispay_Circle() without calling drawCircle(). It should print only broder.
11. Try displaying circle without setting origin and radius i.e call drawCircle() and dispay_Circle() without calling set_circle_param()
12. Input large value for radius such that each circle point falls out of the raster. Should see only empty border

Known Bugs:

1. In current program, the origin of the circle can become invalid if we resize the raster such that origin lies outside of the new raster. Setting origin checks the raster size to ensure that origin should be within the raster boundary but the raster resizing function does not check if the current origin point will become invalid after resizing. However display function will handle this situation before printing the circle on console. I am deferring this bug for now. In next version we can enable the origin to be outside of the raster grid.
2. Circle can be distorted if the raster grid size provided is more than the maximum capacity of the computer screen. We are not checking for max width and length of monitor. It depends on device. Providing invalid length will look more ugly as consoles are not scrollable along x-axis.
3. When radius is small like 1 or 2, shape of circle is not so round. For larger radius it should be fine.