Assignment 3 - Control Flow Graphs

Team 16

I Progress bar

1. Function:

First step is to determine whether the point is even within the bounds of the circle at all.

We know that radius r = 50

We also know that the center of the circle lies at 50,50

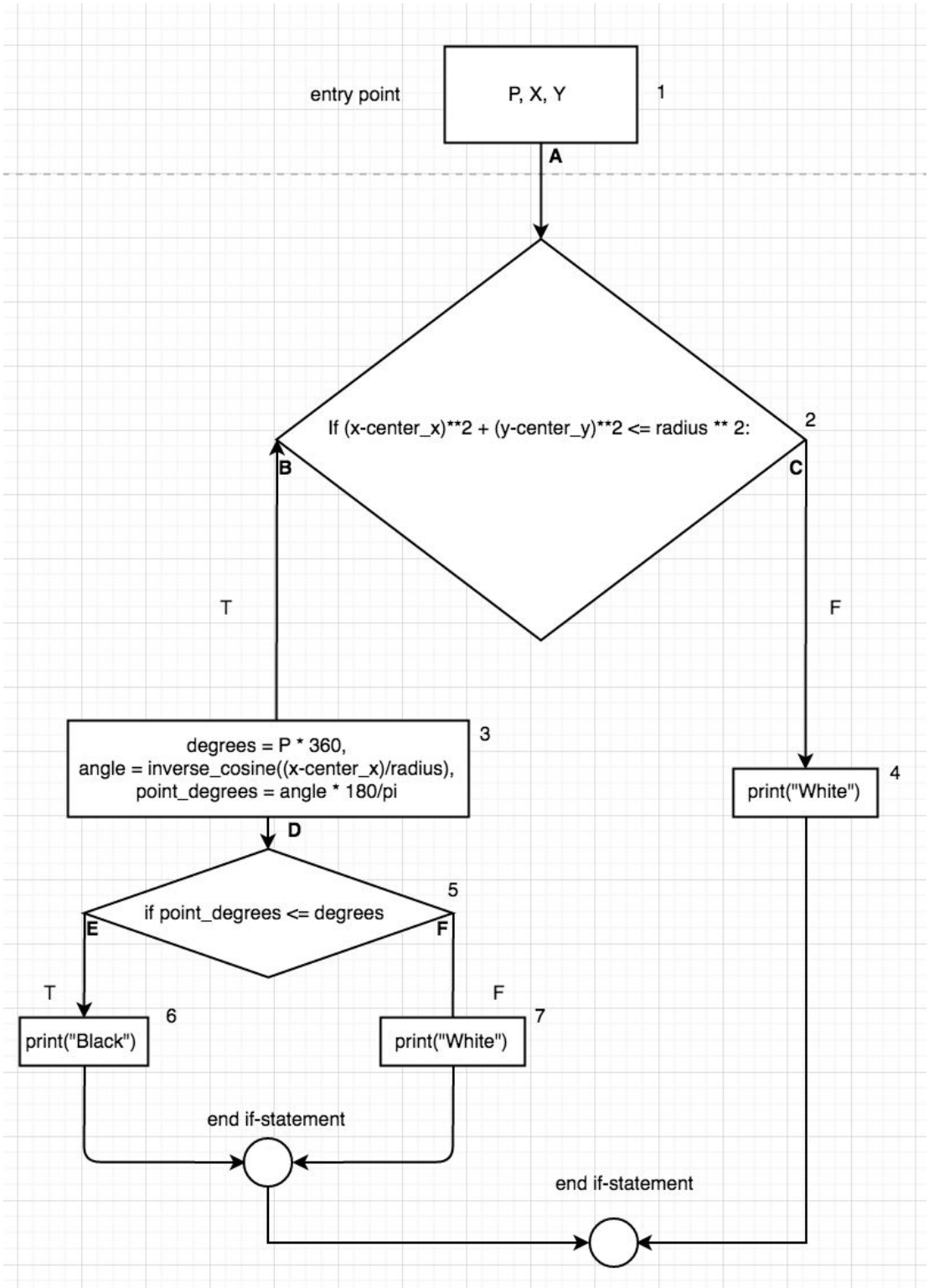
We can use a simple distance formula to verify that the passed point is less than the radius of 50

to convert from radians to degrees we multiply by 180/pi

first step is to convert from percentage to degrees then from degrees to

```
center_x = 50
center_y = 50
radius = 50
def determine_color(P, X, Y):
        If (x-center_x)**2 + (y-center_y)**2 <= radius ** 2:
                # within bounds of circle continue consideration
                degrees = P * 360 * .01 //93.6
                #x = Cos(angle) * radius + CenterX;
        #       angle = inverse_cosine((x - center_x)/radius)
                point_degrees = angle * 180/Pi
                If point_degress <= degrees:
                        print("Black")
                else:
                        print("White")
        else:
                print("White")
```

2. Control Flow Diagram:



3. Test cases

Input (P, X, Y)	Output	Statement Coverage	Coverage percentage
(0,0,0)	“White”	1A → 2C → 4	~42,8%
(12,55, 55)	“White”	1A → 2B → 3D → 5F → 7	~71%
(13,55,55)	“White”	1A → 2B → 3D → 5F → 7	~71%
(60,90,90)	“White”	1A → 2C	~28,5%
(26,54,19)	“Black”	1A → 2B → 3D → 5E → 6	~71%

Branch Coverage will be 3, because we cannot cover all possible true/false decisions without 3 paths according to our code. **Statement coverage** will also be 3 paths, this is the shortest amount of paths according to our code above.

4. Test case achievement

The test cases do not achieve 100% statement, branch or loop coverage in our case, because our code is divided with if statements so you can only cover one of the statements with one path and not all. There is no test case that can give our diagram/code 100% statement, branch or loop coverage. The only way to get 100% coverage is to change our code so that we can cover all branches/paths with 1 path/try.

II. Tower Of Hanoi

Tower of Hanoi Code:

```
public class TowerOfHanoi
{
	public void steps(int diskNum, String position, String destination, String extra)
	{
		if(diskNum==1)
		{
			//move the disk from it's position to the destination
			//base case
			System.out.println(position + " to " + destination);
		}
		else
		{
			steps(diskNum-1, position, extra, destination);
			System.out.println(diskNum + " to " + destination);
			steps(diskNum-1, extra, position, destination);
		}

	}
}
```

Control Flow Graph For Tower Of Hanoi: