# Task 1 (of 2)

There are four observed fundamental forces that form the basis of all known interactions in nature: gravitational, electromagnetic, strong nuclear, and weak nuclear forces. Electric forces are subcomponents of electromagnetic forces and are created by electric fields. Electric fields are present, for example, in coaxial cables used in cable television service, video, antennae, data control, and instrumentation. Under the configuration shown in Figure 1, the electric flux density (***D***) in a long coaxial cable of external radius ***b*** is given as a function of the internal radius

(***a***), the density of charge (*v*), and the radial distance, ***r***, from the center of the cable. The exact equation used to determine ***D*** is based on the position of ***r*** relative to ***a*** & ***b*** based on the conditions shown in the table below:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |

|  |  |
| --- | --- |
| ***r (cm)***  | ***D (nC/cm2)***  |
| 0 < r <= a  | 𝜌𝑣𝑟/2  |
| a < r < b  | 𝜌𝑣𝑎2/(2𝑟)  |
| b  r  | 0  |

 |

## Figure 1: A coaxial cable[[1]](#footnote-1)

Write a **MATLAB** script that will ask the user to input the charge

density (𝜌𝑣), the radial distance ***r***, the internal radius ***a***, and the external radius ***b,*** then determine the electric flux density ***D*** (nC/cm2), and finally output the value of ***D*** using a print statement to the MATLAB command window.

**Test case:** 𝜌𝑣 = 8 𝑛𝐶/𝑐𝑚3, *r* = 2 *cm*, *a =* 3.0 *cm* and *b =* 6.0 *cm*

Your MATLAB Command Window should look like this for the test case:

Enter the charge density (nC/cm^3) = 8

Enter the radial distance r(cm): 2

Enter the internal radius a(cm): 3

Enter the external radius b(cm): 6

The electric flux density, D, is 8.0 nC/cm^2

# Task 2 (of 2)

A common application for repetition flow is the simulation of some games. The following game
has two players. To start the game each player rolls a dice, the player with the highest number is
the Driver and the other player is the Follower. If there is a tie, both players roll the dice again
until there is a Driver and a Follower. Then, the Follower rolls their dice n-times. Each time the
Follower gets the same number as the Driver, the Follower scores a point, otherwise, the Driver
scores a point. At the end of the game, the player will the highest number of points is the winner.

Create a MATLAB script that will:
• Prompt the user to enter the number of rounds n, and the name of each player.
• Determine the Driver and the Follower and keep track of the score for each player
• Display the winner for each round
• Display the winner after n-rounds and the number of points for each player.

Use the following MATLAB function to simulate a dice: X = randi([1, 6], 1). The
function will randomly output an integer between 1 and 6.

**Test Case:**

**Note: the points will vary each time the code is executed since the rolling of the dice is
generated using a random generator.**

Enter Player 1 name: John Doe
Enter Player 2 name: Jane Doe
The number of attempts n = 6
The driver is Jane Doe, the follower is John Doe
The winner of the round 1 is Jane Doe
The winner of the round 2 is Jane Doe
The winner of the round 3 is John Doe
The winner of the round 4 is Jane Doe
The winner of the round 5 is John Doe
The winner of the round 6 is Jane Doe
The winner of the game is Jane Doe
The points for Jane Doe is 4
The points for John Doe is 2

1. *Fundamentals of Electromagnetics with Engineering Applications*. by Stuart M. Wentworth. 2005 John Willey and Sons (Pages 51-54). [↑](#footnote-ref-1)