DAILY ONLINE ACTIVITIES SUMMARY

Date:	**12 June 2020**	**Name:**	**Asha Rudrappa Totagi**
Sem& Sec	**6thsem& A sec**	**USN:**	**4AL17CS015**

Certification Course Summary

Course	**Ethical Hacking**

Certificate Provider	**Udemy**	**Duration**	**3 hours**

Coding Challenges

Problem Statement

Program 1: Write a Python program to implement Magic Square. A magic square of order n is an arrangement of n^2 numbers, usually distinct integers, in a square, such that the n numbers in all rows, all columns, and both diagonals sum to the same constant. A magic square contains the integers from 1 to n^2.

The constant sum in every row, column and diagonal is called the magic constant or magic sum, M. The magic constant of a normal magic square depends only on n and has the following value:

M = n(n^2+1)/2

example

Magic Square of size 5

```
9 3 22 16 15
2 21 20 14 8
25 19 13 7 1
18 12 6 5 24
11 10 4 23 17
```

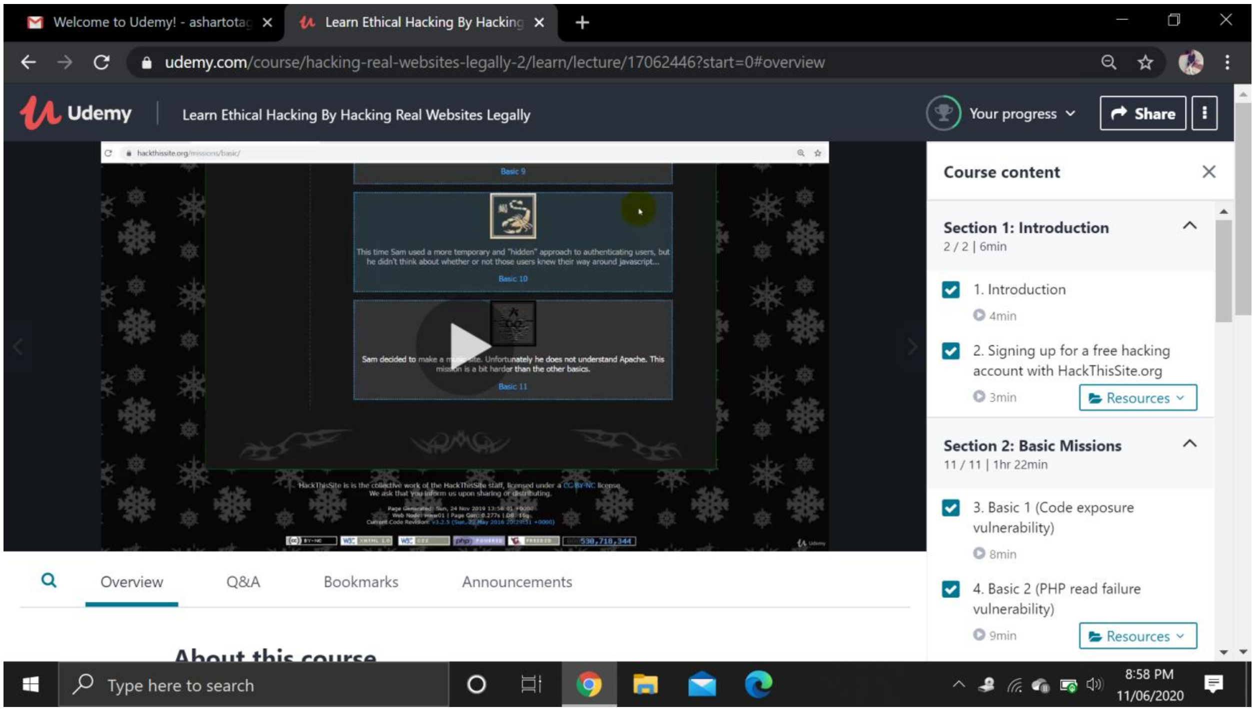
Sum in each row &amp; each column = 5*(5^2+1)/2 = 65

Program 2: Write a Java program to find maximum width of a binary tree

Status: DONE

Uploaded the report in Github	**YES**
If yes Repository name	**Daily Status**
Uploaded the report in slack	**YES**

Certification Course Details: (Attach the snapshot and briefly write the report for the same


Welcome to Udemy! - ashartotag
Learn Ethical Hacking By Hacking
udemy.com/course/hacking-real-websites-legally-2/learn/lecture/17062446?start=0#overview
Udemy
Learn Ethical Hacking By Hacking Real Websites Legally
Your progress
Share
hackthissite.org/missions/basic/
Basic 9
This time Sam used a more temporary and "hidden" approach to authenticating users, but he didn't think about whether or not those users knew their way around javascript...
Basic 10
Sam decided to make a music site. Unfortunately he does not understand Apache. This mission is a bit harder than the other basics.
Basic 11
Overview
Q&A
Bookmarks
Announcements
About this course
Type here to search
8:58 PM
11/06/2020

Course content

Section 1: Introduction
2 / 2 | 6min

- ☑ 1. Introduction — 4min
- ☑ 2. Signing up for a free hacking account with HackThisSite.org — 3min — Resources

Section 2: Basic Missions
11 / 11 | 1hr 22min

- ☑ 3. Basic 1 (Code exposure vulnerability) — 8min
- ☑ 4. Basic 2 (PHP read failure vulnerability) — 9min — Resources

DAY 2 (12-06-2020) - Introduction to ethical hacking and basic missions to hack.

Coding Challenges Details: (Attach the snapshot and briefly write the report for the same)

Program 1:

```
def generateSquare(n):
    magicSquare = [[0 for x in range(n)]
                    for y in range(n)]
    i = n / 2
    j = n - 1
    num = 1
    while num <= (n * n):
        if i == -1 and j == n:
            j = n - 2
            i = 0
        else:
            if j == n:
                j = 0
            if i < 0:
                i = n - 1
        if magicSquare[int(i)][int(j)]:
            j = j - 2
            i = i + 1
            continue
        else:
            magicSquare[int(i)][int(j)] = num
            num = num + 1
        j = j + 1

        i = i - 1

    print ("Magic Square for n =", n)

    print ("Sum of each row or column",n * (n * n + 1) / 2, "\n")

    for i in range(0, n):

        for j in range(0, n):

            print('%2d ' % (magicSquare[i][j]),end = '')

            if j == n - 1:

                print()

n=int(input("Number of rows of the Magic Square:"))

generateSquare(n)
```

Program 2:

```
import java.util.LinkedList;
import java.util.Queue;

public class Main {
      public static class Node{
        int data;
        Node left;
        Node right;

        public Node(int data){
          this.data = data;
          this.left = null;
          this.right = null;
        }
      }
      public Node root;

      public Main(){
        root = null;
```

```
}
public int findMaximumWidth() {
    int maxWidth = 0;
    int nodesInLevel = 0;
    Queue<Node> queue = new LinkedList<Node>();
    if(root == null) {
        System.out.println("Tree is empty");
        return 0;
    }
    else {
        queue.add(root);

        while(queue.size() != 0) {
            nodesInLevel = queue.size();
            maxWidth = Math.max(maxWidth, nodesInLevel);
            while(nodesInLevel > 0) {
                Node current = queue.remove();
                if(current.left != null)
                    queue.add(current.left);
                if(current.right != null)
                    queue.add(current.right);
                nodesInLevel--;
            }
        }
    }
    return maxWidth;
}

public static void main(String[] args) {

    Main bt = new Main();
    bt.root = new Node(1);
    bt.root.left = new Node(2);
    bt.root.right = new Node(3);
    bt.root.left.left = new Node(4);
    bt.root.left.right = new Node(5);
    bt.root.right.left = new Node(6);
    bt.root.right.right = new Node(7);
    bt.root.left.left.left = new Node(8);
    System.out.println("Maximum width of the binary tree: " + bt.findMaximumWidth());
    }
}
```