משימה מספר 1 ניקוד 1

כתוב תוכנית הקולטת שני מספרים שלמים לתוך משתנים בשם x ו-y.

על התוכנית להחליף את התוכן שלהם **מבלי** להשתמש במשתנים נוספים כלשהם.

משימה מספר 2 ניקוד 4

הגדר פונקציה המקבלת מערך של ציונים ואת גודלו, ומחזירה את השונות הסטטיסטית של אוסף הציונים. המשוואה לחישוב השונות הסטטיסטית היא:  כאשר N הוא מספר הערכים ו..µ הינו ממוצע הערכים.

**לדוגמא**: לערכי בוחן 65, 76, 93 החישוב יהיה:

ממוצע: (93+76+65)/3=78

השונות הסטטיסטית: ((65-78)2+(76-78)2+(93-78)2)/3 = 132.66667

ניתן להניח ש- n > 0.

יש להשתמש בכותרת הפונקציה הבאה:

double variance (int\* marks, int n)

משימה מספר 3 ניקוד 6

הגדר פונקציה המקבלת מספר שלם, rowIndex ומחזירה את שורת ה- rowIndex של משולש פסקל (כאשר השורות הן 0-indexed, כלומר ממוספרים מ-0)

דוגמאות:

עבור rowIndex = 3 יוחזר {1,3,3,1}

עבור rowIndex = 0 יוחזר {1}

עבור rowIndex = 1 יוחזר {1, 1}

עבור rowIndex = 4 יוחזר {1,4,6,4,1}

ניתן להניח 0 <= rowIndex <= 32

יש להשתמש בכותרת הפונקציה הבאה:

int\* pascal (int rowIndex)

משימה מספר 4 ניקוד 8

הגדר פונקציה המקבלת מספר שלם *, colNum*, ומחזירה את הכותרת של העמודה המתאימה בגיליון אקסל.

כלומר, 1 🡪 A, 2🡪 B, 3🡪C….26🡪 Z, 27🡪AA, 28🡪AB…,AZ🡪52,53🡪BA,….

ניתן להניח ש- 212 ≥ *colNum* ≥ 1

יש להשתמש בכותרת הפונקציה הבאה:

string excelTitle (int colNum)

משימה מספר 5 ניקוד 8

הגדר פונקציה המקבלת מערך *nums*, המכיל מספרים שלמים ,וגודלו *n*. על הפונקציה להזיז את כל האפסים לסוף המערך תוך שמירה על הסדר היחסי של האלמנטים שאינם אפס.

הערה: חייבים לבצע את המשימה in-place מבלי ליצור עותק של המערך.

דוגמאות:

עבור nums = [0,42,0,14,28] יוחזר [42,14,28,0,0]

עבור nums = [0] יוחזר [0]

יש להשתמש בכותרת הפונקציה הבאה:

void moveZeroes( int\* nums, int n)

משימה מספר 6 ניקוד 6

להלן הגדרה של סדרת קומרצ'י Kn .

K0 = 0, K1 = 1, K2 = 1, and Kn = Kn-1 + Kn-2 + Kn-3 for n >= 3

הגדר פונקציה המקבלת מספר שלם n (32 => n => 0) ומחזירה את האיבר ה-n בסדרה.

יש להשתמש בכותרת הפונקציה הבאה:

int komarachi(int n)

משימה מספר 7 ניקוד 4

הגדר פונקציה powerOf3 המקבלת מספר שלם n ומחזירה true אם n הוא חזקה של 3, אחרת מחזירה false.

מספר שלם n היא חזקה של 3 אם קיים מספר שלם x המקיים n = 3x.

דוגמאות:

עבור n = 81 יוחזר true כי 34 = 81

עבור n = 0 יוחזר false כי לא קיים ערך עבור x המקיים 3x = 0

עבור n = -1 יוחזר false כי לא קיים ערך עבור x המקיים 3x = (-1)

ניתן להניח ש- 214 ≥ *n* ≥ 214-

יש להשתמש בכותרת הפונקציה הבאה:

bool powerOf3(int n)

משימה מספר 8 ניקוד 3

הגדר פונקציה המקבלת כפרמטר מספר שלם חיובי, n, ומחזירה את מספר המחלקים שלו (כולל 1 והוא עצמו).   
לדוגמא, עבור n=100 יוחזר 9 כיון שלמספר 100 יש 9 מחלקים.

יש להשתמש בכותרת הפונקציה הבאה:

int divisors(int n)

משימה מספר 9 ניקוד 14

הגדר פונקציה המקבלת מערך arr המכיל מספרים שלמים ומחזירה true אם כמות ההופעות של כל ערך במערך הוא ייחודי, אחרת מחזירה false.

דוגמאות

עבור arr = [12,28,28,12,12,32] יוחזר true

הסבר: לערך 21 יש 3 מופעים, ל-28 יש 2 מופעים ולערך 32 יש 1.

עבור arr = [1,2] יוחזר false

עבור arr = [-14,0,1,-14,1,1,0,1,28,1,0,0] יוחזר true

יש להשתמש בכותרת הפונקציה הבאה:

bool uniqueOccurrences(vector<int> arr)

משימה מספר 10 ניקוד 15

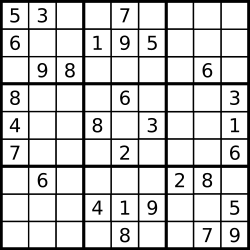
Your goal is to determine if a 9 x 9 Sudoku board is valid. Only the filled cells need to be validated **according to the following rules**:

1. Each row must contain the digits 1-9 without repetition.
2. Each column must contain the digits 1-9 without repetition.
3. Each of the nine 3 x 3 sub-boxes of the grid must contain the digits 1-9 without repetition.

**Note:**

* A Sudoku board (partially filled) could be valid but is not necessarily solvable.
* Only the filled cells need to be validated according to the mentioned rules.
* Empty cells contain the value 0.

**Example 1:**



Output = true

Write a function that accepts a two-dimensional array of size 9x9 containing integers and returns true if the array represents a valid sudoku board otherwise returns false.

Use the following function definition for your solution.

bool sudoku(int board [ ] [9])

משימה מספר 11 ניקוד 15

You are given three strings: s1, s2, and s3. In one operation you can choose one of these strings and delete its **rightmost** character.

Note that you **cannot** completely empty a string.

Define a function that return the *minimum number of operations* required to make the strings equal*.*If it is impossible to make them equal, return -1.

Example 1:  s1 = "abc", s2 = "abb", s3 = "ab" returns 2

Explanation: Deleting the rightmost character from both s1 and s2 will result in three equal strings.

Example 2: s1 = "dac", s2 = "bac", s3 = "cac" returns -1

Explanation: Since the first letters of s1 and s2 differ, they cannot be made equal.

 Use the following function definition for your solution.

int findMinimumOperations(string s1, string s2, string s3)

משימה מספר 12 ניקוד 8

הגדר פונקציה המקבת ה-head של רשימה מקושרת חד-כיוונית ומספר שלם n. על הפעולה להוציא החולייה ה-*n* מסוף הרשימה ולהחזירה את ה-head של הרשימה.

יש להשתמש בחתימת הפונקציה - ListNode\* removeNthFromEnd(ListNode\* head, int n)

וההגדרה הבאה עבור הרשימה

// Definition for a node in a single - linked list.

struct ListNode {

int val;

ListNode\* next;

ListNode() : val(0), next(nullptr) {}

ListNode(int x) : val(x), next(nullptr) {}

ListNode(int x, ListNode\* next) : val(x), next(next) {}

};

דוגמאות:

For head = [1,2,3,4,5], n = 2

[1,2,3,5] is returned

A diagram of a diagram

Description automatically generated

For head = [1], n = 1

[ ] is returned

For head = [1,2], n = 1

[1] is returned

משימה מספר 13 ניקוד 14

Suppose an array, *nums,* containing integers of length n sorted in ascending order is **rotated** between 1 and n times.

For example, the array nums = [0,1,2,4,5,6,7] might become:

* [4,5,6,7,0,1,2] if it was rotated 4 times.
* [0,1,2,4,5,6,7] if it was rotated 7 times.

Define a function that accepts a sorted rotated array nums of **unique** elements, and returns *the minimum element of this array*.

**You must write an algorithm that runs in O(log n) time.**

Example 1: nums = [14,28,32,7,12] returns 7

Explanation: The original array was [7,12,14,28,32] rotated 3 times.

Example 2: nums = [4,5,6,7,0,1,2] returns 0

Explanation: The original array was [0,1,2,4,5,6,7] and it was rotated 4 times.

Example 3: nums = [14,17,28,32] returns 14

Explanation: The original array was [14,17,28,32] and it was rotated 4 times.

To implement your solution, use the following function definition:

int findMin(vector<int>& nums)

.

משימה מספר 14 ניקוד 18

כזכור, בכל צומת בעץ בינארי יש או 0, 1 או 2 בנים לא ריקים.

כתבו פונקציה רקורסיבית nTrees המקבלת מספר שלם n הגדול או שווה לאפס, ומחזירה את מספר העצים הבינאריים השונים האפשריים עם n צמתים.

לדוגמא, בעץ עם בדיוק 3 צמתים יש 5 עצים בינאריים שונים, כדלהלן:

יש להשתמש בכותרת הפונקציה הבאה:

int nTrees(int n)

משימה מספר 15 ניקוד 25

נתון 2 מערכים , preorder ו-inorder המכילים מספרים שלמים. המערך preorder מכיל את הסריקה התחילית (preorder) של עץ בינרי, והמערך inorder מכיל את הסריקה התוכית (inorder) של אותו עץ.

הגדר פונקציה המקבלת את שני המערכים , בונה ומחזירה את העץ הבינרי לפי הסריקות.

יש להשתמש בהגדרת הפונקציה הבאה:

TreeNode\* buildTree(vector<int>& preorder, vector<int>& inorder)

אפשר להניח שהערכים בעץ הם שונים זו מזו, וידוע שמספר הצמתים בעץ > 248.

דוגמאות:

עבור preorder = [3,9,20,15,7], inorder = [9,3,15,20,7]יוחזר [3,9,20,null,null,15,7]

(זה העץ המתקבל על מנת להבין את השאלה)

עבור: preorder = [-1], inorder = [-1] יוחזר [1-]

לרשותכם ההגדרה הבאה עבור צומת בעץ בינארי:

struct TreeNode {

int val;

TreeNode\* left;

TreeNode\* right;

TreeNode() : val(0), left(nullptr), right(nullptr) {}

TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}

TreeNode(int x, TreeNode\* left, TreeNode\* right) : val(x), left(left), right(right) {}

};

משימה מספר 16 ניקוד 25

There are n rooms labeled from 0 to n - 1 and all the rooms are locked except for room 0. Your goal is to visit all the rooms. However, you cannot enter a locked room without having its key.

When you visit a room, you may find a set of **distinct keys** in it. Each key has a number on it, denoting which room it unlocks, and you can take all of them with you to unlock the other rooms.

Define a function that accepts an array, *rooms* where rooms[i] is the set of keys that you can obtain if you visited room i, and returns true *if you can visit****all****the rooms, or* false *otherwise*.

Example 1:

rooms = [[1],[2],[3],[]] returns true

Explanation:

We visit room 0 and pick up key 1.

We then visit room 1 and pick up key 2.

We then visit room 2 and pick up key 3.

We then visit room 3.

Since we were able to visit every room, we return true.

Example 2:

rooms = [[1,3],[3,0,1],[2],[0]] returns false

Explanation: We can not enter room number 2 since the only key that unlocks it is in that room.

To implement your solution, use the following function definition:

bool canVisitAllRooms(vector<vector<int>>& rooms)

משימה מספר 17 ניקוד 10

הגדר פונקציה המקבלת מערך לא ממוין array וגודלו, המכיל מספרים שלמים ומחזירה מה הערך השכיח ביותר במערך (איזו מספר מופיע מספר מקסימלי של פעמים). אין למיין את המערך ואין לבנות מערך עזר.

ניתן להניח שקיים ערך שכיח אחד במערך.

יש להשתמש בכותרת הפונקציה הבאה:

int frequent ( const int a [], int n)

משימה מספר 18 ניקוד rand()%13+1

Define a function that accepts an integer, *n*, greater than or equal to zero and returns the count of all numbers with unique digits, x,   
where 0 <= x < 10n.

Example 1: For n = 2 the returned value will be 91

Explanation: The answer should be the total numbers in the range of

0 ≤ x < 100, excluding 11,22,33,44,55,66,77,88,99

Example 2: For n = 0 the returned value will be 1

You can assume 0 <= n <= 5

To implement your solution, use the following function definition:

int countSingle(int n)