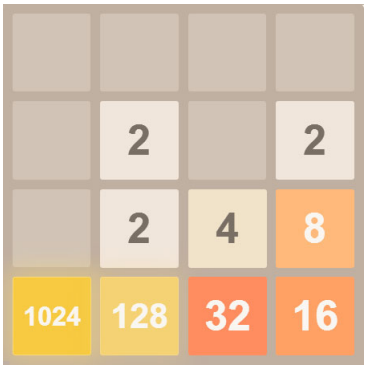
АЛГОРИТМИ ВЪРХУ ГРАФИ
ДОМАШНО № 2 ПО "ДИЗАЙН И АНАЛИЗ НА АЛГОРИТМИ"
ЗА СПЕЦИАЛНОСТ "КОМПЮТЪРНИ НАУКИ", 1. ПОТОК
(СУ, ФМИ, ЛЕТЕН СЕМЕСТЪР НА 2019 / 2020 УЧ. Г.)

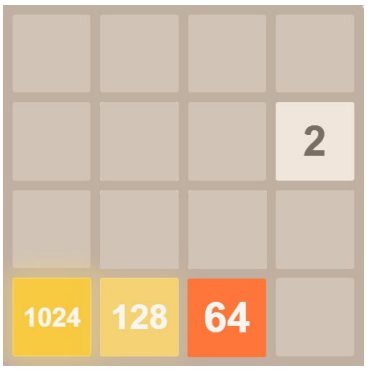
Задача 1 е вдъхновена от компютърната игра "2048". Тук използваме различни правила с цел опростяване на задачата.



Плочки с числа (степени на двойката) са разположени върху клетките на квадратно игрално табло. Някои клетки са заети с плочки, други клетки са свободни. Две клетки ще наричаме съседни, ако имат обща страна; всяка клетка, която не е по края на таблото, има четири съседни клетки. Играчът не може да мести старите плочки — тези, които вече са заели своите места. На всеки ход играчът получава по една нова плочка, която той има право да постави на произволно избрана от него свободна клетка от таблото.

Пример: на таблото, показано вдясно, играчът току-що е получил плочка с числото 2 и е избрал да я сложи върху главния диагонал.

После автоматично протича следната верижна реакция: ако в клетка, съседна на текущата (избраната от играча), има плочка със същото число, текущата клетка се опразва, като плочката от нея се придвижва към съседната клетка, където плочките се сливат (числата се събират), а клетката с промененото число става текуща и на свой ред се слива с някой от съседите си. Процесът продължава, докато това е възможно. Ако на някоя стъпка текущата клетка има две или повече подходящи съседни клетки, играчът избира с коя съседна клетка да се слее текущата.



В примера, показан по-горе, двете съседни клетки 2 се сливат, като горната (новата) плочка слиза. Получават се две съседни четворки. Те на свой ред се сливат, при което се получават две съседни осмици и тъй нататък. Накрая се стига до положението, показано на долната диаграма: числото 64 няма равен съсед и верижната реакция спира.

Ако играчът беше избрал да сложи новата плочка в горния десен ъгъл на таблото, тя щеше да се слее само с плочката под нея и верижната реакция щеше да спре до числото 4.

Целта на играча е да получи възможно най-голямо число в края на верижната реакция (и да опразни възможно най-много клетки на таблото). Помогнете на играча, като съставите алгоритъм, който определя в коя клетка да се постави новата плочка и накъде да продължи верижната реакция, ако на някоя стъпка има две или повече възможности. За целта обяснете как входните данни могат да се представят чрез граф и изберете някой изучен алгоритъм върху графи. Достатъчно е словесно описание на модела и алгоритъма, няма нужда от код. Алгоритъмът трябва да работи върху произволно голямо табло и да бъде възможно най-бърз.

(2 точки)

Демонстрирайте модела, който сте избрали, като начертаете графа, който съответства на по-горната диаграма. Изпълнете избрания алгоритъм върху този граф, за да покажете как алгоритъмът намира положението, показано върху по-долната диаграма.

(1 точка)

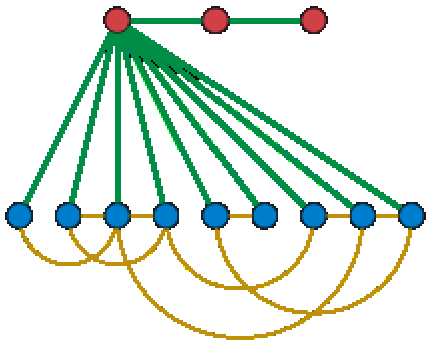
Задача 2. Дадени са тегловен граф G и едно негово минимално покриващо дърво T. (Дървото T е вече построено.) Към графа G се добавя ново ребро, чиито краища са измежду съществуващите върхове на G, тоест не се добавят върхове. Съставете възможно най-бърз алгоритъм, който да построява минимално покриващо дърво на новия граф. Такъв алгоритъм не е бил изучаван на лекции; очаква се да го съставите "от нулата" — без да разчитате на готови алгоритми.

а) Опишете алгоритъма с думи. **(1 точка)**

б) Демонстрирайте алгоритъма с пример. Демонстрацията трябва да бъде пълна — от входните данни до отговора на алгоритъма. **(1 точка)**

в) Докажете коректността на предложения алгоритъм. **(2 точки)**

Задача 3. Граф с $n \geq 5$ върха се нарича скорпион, ако притежава връх от първа степен (жилото), който е свързан с връх от втора степен (опашката), който е свързан с връх от степен $n - 2$ (тялото), който е свързан с останалите $n - 3$ върха (краката). Възможно е някои крака да са свързани помежду си.



Дадена е симетрична двоична матрица $n \times n$ със стойности *false* върху главния диагонал, която тълкуваме като матрица на съседствата на неориентиран граф с n върха (без примки).

Предложете алгоритъм, който за време $O(n)$ в най-лошия случай разпознава дали графът, зададен от матрицата, е скорпион. Обърнете внимание, че дължината на входа е $\Theta(n^2)$, тоест времевата сложност $O(n)$ е сублинейна и алгоритъмът не разполага с достатъчно време за прочитането на всички входни данни. Следователно алгоритъмът не може да провери дали дадената матрица е симетрична: тази проверка изисква време $\Omega(n^2)$. С други думи, алгоритъмът трябва да приеме наготово (без проверка), че дадената матрица е симетрична. Алгоритъмът може да приеме наготово и това, че матрицата съдържа само стойности *false* върху главния диагонал. Накратко, алгоритъмът приема без проверка, че дадената матрица е коректно представяне на неориентиран граф без примки. Единственото, което алгоритъмът трябва да провери, е дали графът представлява скорпион.

Опишете алгоритъма на програмния език Си във вид на функция със следния формат: `bool IsScorpion(Graph G)`. Типът `Graph` да бъде деклариран в програмата така: `typedef bool Graph[n][n]`, а идентификаторът `n` да бъде име на константа.

Ако в използваната версия на езика Си липсва тип `bool`, вместо него може да се ползва типът `int`, вместо константата *false* — числото 0, а вместо *true* — числото 1.

Програмата може да използва само типовете `int`, `bool` и масив. Да не се използват никакви готови функции (библиотеки).

(3 точки)