Задача 1.

Играта започва с квадратна дъска, състояща се от плочки с номера от 1 до N и една празна плочка, представена с цифрата 0. Целта е да се наредят плочките в съответствие с техните номера. Местенето се извършва, като на мястото на празната плочка могат да се преместят плочки отгоре, отдолу, отляво и отдясно.

На входа се подава число N - броя на плочките с номера (8, 15, 24 и т.н.), число I - индексът на позицията на нулата в решението (при -1 се задава индекс по подразбиране - най-долу в дясно) и след това се въвежда подредбата на дъската. С помощта на алгоритъма IDA* и евристиката "разстояние на Манхатън" да се изведе:

1) На първият ред дължината на "оптималния" път от началото до целевото състояние.

2) Съответните стъпки (на нов ред за всяка една), които се извършват за да се стигне до крайното състояние. Стъпките са left, right, up и down

** Имайте предвид, че не всяка конфигурация на входен пъзел, която подадете, е решима. Дали пъзелът е решим може да се провери като обяснения как това може да се направи могат да бъдат намерени тук.*

*** Моля, принтирайте времето необходимо за намиране на пътя (без да включвате времето за принтиране на решението) в секунди с точност поне до 2-рия знак след нулата.*

Примерен вход:

8

-1

1 2 3
4 5 6
0 7 8

Примерен изход:

2

left

left

Задача 2
Разположете N царици на дъска NxN, така че да не се бият. Използвайте алгоритъма MinConflicts за решение на задачата.

Вход:

цяло число N - броя на цариците, които да се разположат.

** Изискване да работи за N=10 000 (за под секунда)*

Изход:

изведете на стандартния изход игралната дъска като обозначите царица със * а празна клетка със _

Примерен вход:

4

Примерен изход:

_ * _ _

_ _ _ *

* _ _ _

_ _ * _

Задача 3
Да се реши задачата за търговския пътник (Traveling Salesman Problem) чрез използване на генетичен алгоритъм.

За целта на програмата се задава: N - число (N <= 100) - брой точки в пространството (брой градове).

Програмата генерира N точки от двумерна координатна система на случаен принцип.

Търси се най-къс път, който да минава през всяка точка само по един път. За целта нека да се изведе на поне пет стъпки дължината на текущо най-добрия път в популацията.

- 1. на 1-та генерация
- 2, 3, 4 - по избор
- 5. След последната генерация.

Задача 4
Да се имплементира игра на морски шах срещу противник, като се използва алгоритъм min-max с alpha-beta отсичане.
Започва се с празна дъска.

На всяка стъпка:

- единият играч въвежда две числа ([1,3]) от клавиатура, които са неговия ход на дъската;
- използвайки алгоритъма, намираме и нашия ход;
- след това се показва конкретното състояние на играта.

Накрая се показва кой е спечелил играта.
* Направете играта така, че да може да се задава дали компютърът или играчът е първи.

** Направете алгоритъма оптимален.

Задача 5
Реализирайте Наивен Бейсов Класификатор, класифициращ индивидите като демократи или републиканци, използвайки 16-те атрибута и двата класа от следните данни:

http://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records

За тестване на алгоритъма приложете 10-fold крос-валидация (разделете данните по случаен начин на 10 множества и направете 10 обучения и тествания на модела като всеки път използвате едно от множествата за тестване, а останалите 9 за обучение).

Изведете метриката точност (Accuracy) за всяко от 10-те обучения, както и тяхното средно аритметично (за обобщена оценка на класификатора).

* Данните може да имат нужда от предварителна обработка.

Задача 6
Реализирайте алгоритъма за класификационно дърво ID3 и го приложете върху данните breast-cancer.arff (https://archive.ics.uci.edu/ml/datasets/breast+cancer). Използвайте кросвалидация за изчисляване на точността на модела върху обучаващото множество.

За избягване на преспецифициране (overfitting) на дървото използвайте константа K дефинираща минимален брой на обучаващи примери в множеството.

** Опитайте се да приложите друг подход за избягване на преспецифициране (overfitting). Сравнете резултата.*