# Artificial Intelligence

## Assignment 2

Assignment due on: 12.11.2021

Notes: For all the programming tasks, you must use Python 3. In this assignment, you are not allowed to use any external modules (e.g *numpy* or *simpleai*) of Python unless otherwise stated.

### Question 1 Search spaces (2+4 points)

(a) Consider the $8$-queens problem from slide 11 of chapter 3.

- How large is the state space?
- How large is the state space if we consider a valid state to be exactly $4$ queens on the board while each row contains at most $1$ queen?

(b) Now consider the following scenario of PacMan, where we have a $3 \times 10$ board with some empty tile (black) and some with food (white dots) for PacMan to eat. Once PacMan enters a tile with food, it's automatically eaten. Your goal is to navigate PacMan to eat all food on the board. At any given position PacMan can have four possible orientations facing $North$, $East$, $West$, $South$.

- Give a description of the abstract problem in Figure 1, i.e. give the possible states, the allowed actions, and the goal test.
- Calculate the size of the state space for the board shown in Figure 1, assuming the food items are always placed at the board as shown but may be eaten.
- Now consider we have two ghosts (red & blue) roaming randomly on the given board (Figure 2). Now calculate the size of the state space with the presence of these ghosts. A ghost on a tile has no influence on the food present there.

Figure 1

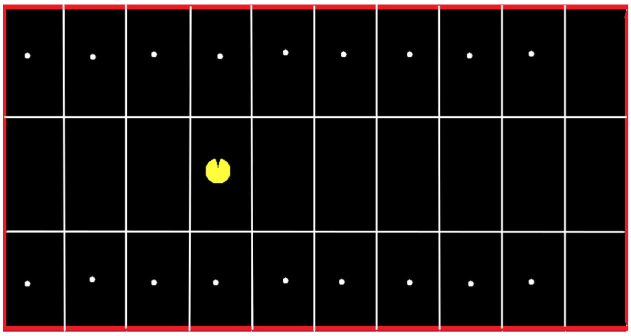
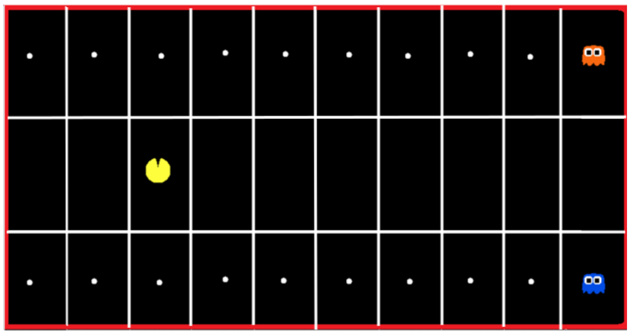
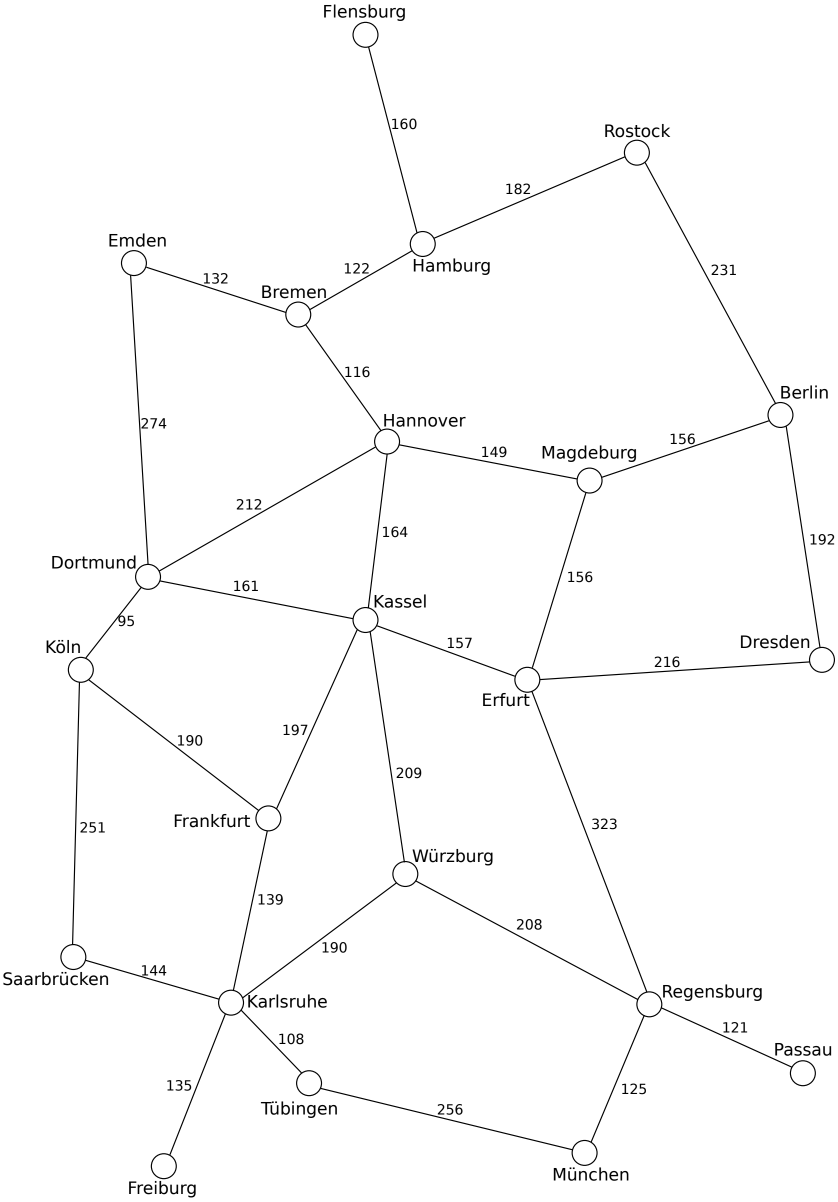


Figure 2



### Question 2 Search strategies (4 + 1 = 5 points)

(a) Given is a graph which models the connections and travel costs between German cities. You start at the node 'Hamburg' and your goal location is 'Tübingen'. If an algorithm does not follow a specific order to store the frontier nodes, save them counterclockwise, starting at 12 o'clock (north). Use the following search algorithms and write down all visited (expanded) nodes in correct order:



i Breadth-first search

ii Depth-first search

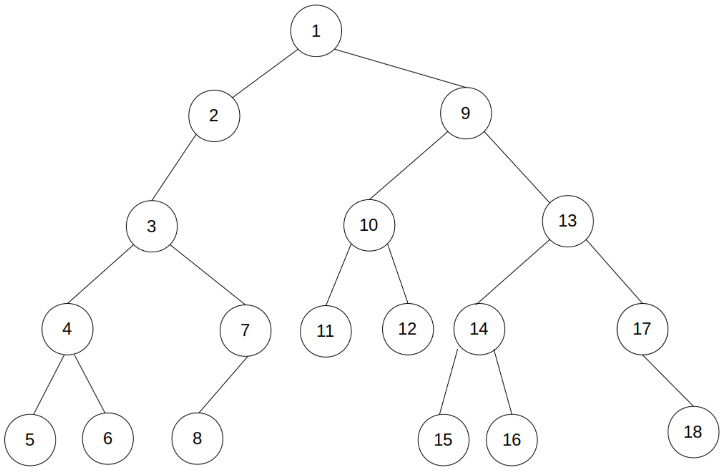
iii Uniform-cost search

(b) State an advantage and disadvantage of the iterative deepening search

## Question 3 Programming in Python (1+3+3+2 = 9 points)

In this task, you have to define a binary tree and functions that search for a value in the given tree. Use the template *assignment_02.py* where we have provided a class named *BinaryTree* which contains the instance variables: *value, left, right. Value* refers to the value of that node and *left* and *right* refers to the left and right branch of that node respectively.

(a) Using the *BinaryTree* class, define a tree which resembles the structure shown below. The values of the nodes are inside the circles. Define each branch (left/right) of a node by another instance of the *BinaryTree* class recursively until the tree is fully defined. For the nodes which do not have left and/or right branches, keep the left and/or right variables value as *None. (1 points)*



(b) Implement a function *depth_first_search(tree, val, search_order)* that looks for the value *val* in the binary *tree* defined in part (a) using the depth-first search algorithm. *search_order* is a list to store order of the nodes, in which *tree* has been searched for value *val.* The function returns *true* if *val* is found in the *tree* otherwise it returns *false. (3 points)*

(c) Implement a function *breadth_first_search(tree, val, search_order)* that looks for the value *val* in the binary *tree* defined above using the breadth-first search algorithm. *(3 points)*

(d) Implement a function *depth_limited_search(tree, val, search_order, limit)* that searches for the value *val* using the depth-limited search algorithm. *(2 points)*
*Note: You are allowed to use collections module from python. Make sure each function can be called by the same name and input arguments as defined in the template file.*