Solution Description

Adi Album

The main Python script is *oskay.py* which consists solely of a main program to be ran as described in the question's description.

oskar.py consists of 3 main parts:

1. Input handling
2. Free-space graph construction and solution finding
3. Output handling

Each part is implemented in a separate Python module *input_handler.py*, *graph_handler.py*, *output_handler.py*.

- ***input_handler.py***
  This Python module oversees parsing the input arguments, running a basic validation for their correctness, and parsing the input obstacle file to three binary 2D matrices representing the obstacles: *xy_plane*, *yz_plane*, and *zx_plane* as given in the input file

- ***graph_handler.py***
  This Python Module is the main part of the solution. It constructs a graph *free_space_graph* consisting a node for each free 3D position, and an edge in between every two neighboring positions. The neighboring relation is naturally defined by the 3D grid.
  The *free_space_graph* is constructed in two parts:
  a. A 3D grid graph is constructed with dimensions parsed from input.
  b. Next, invalid nodes are removed from graph by viewing the planes *xy_plane*, *yz_plane*, and *zx_plane* as parsed from input.
  The shortest path is found using the Dijkstra method.

  In this module we use '*networkx*' Python module.

- ***output_handler.py***
  This python module oversees writing the solution to *solution.txt* as described in the question's description. If no path exists between source and target point the output will be an invalid command '-1'

The entire project is also available on Github: https://github.com/AdiAlbum1/oskar-cube, with an easy install and run guide.