|  |  |
| --- | --- |
| AIN SHAMS UNIVERSITYFACULTY OF ENGINEERINGi-CREDIT HOURS ENGINEERING PROGRAMS *Computer Engineering and Software Systems Program* | Logo  Description automatically generated |

|  |  |
| --- | --- |
| ***Spring 2021*** | **Course Code: *CSE 472*** |
| **Artificial Intelligence** | |

**Course Project**

Submitted by

|  |  |  |
| --- | --- | --- |
|  | **Name** | **ID** |
| **1** | **Omar Khaled Ossman Abdellatif Ghonemy** | **18P1498** |
| **2** | Shehab El Din Adel Nassef Attia | **18P3863** |
| **3** | **Mostafa Mohammad Mohammad Shams Eldeen** | **18P6781** |
| **4** | Zyad Ahmed Aly Hedar Yakan | **18P9538** |

# Introduction

In Artificial Intelligence, Search techniques are universal problem-solving methods. **Rational agents** or **Problem-solving agents** in AI mostly used these search strategies or algorithms to solve a specific problem and provide the best result. Problem-solving agents are the goal-based agents and use atomic representation. In this topic, we will learn various problem-solving search algorithms.

**Based on the search problems we can classify the search algorithms into uninformed (Blind search) search and informed search (Heuristic search) algorithms.**

1. **Uninformed Search (Blind Search)** will be implemented in our program into 3 search algorithms:
2. Breadth First Search (BFS)
3. Depth First Search (DFS)
4. Uniform Cost Search (UCS)
5. **Informed Search (Heuristic search) will be implemented in out program into 2 search algorithms :**
6. A\* search (Astar)
7. Greedy Best First Search (GBFS)

Graphical user interface, application

Description automatically generated

In the rest of the document we will show how to use our GUI to create graphs and draw them and then apply each of the 5 mentioned algorithms to show the optimal path using each algorithm.

# User Guide

In this outline we will discuss how to use the program to generate graphs and then apply all of the 5 search algorithms on it. The following are all the functionalities that can be done using our program and to use the program you must use the functionalities in the following sequence of steps:

1. Creating Directed or Undirected Graphs.
2. Adding nodes to the graph.
3. Adding heuristics to the nodes.
4. Creating edges between the nodes (Can be weighted or unweighted edges).
5. Drawing the Graph.
6. Adding the required Start and Goal Nodes.
7. Applying BFS , DFS , UCS, Astar , and GBFS algorithms.
8. Clearing the canvas.
9. Cleaning (Deleting) the graph.

## Creating Directed or Undirected Graphs

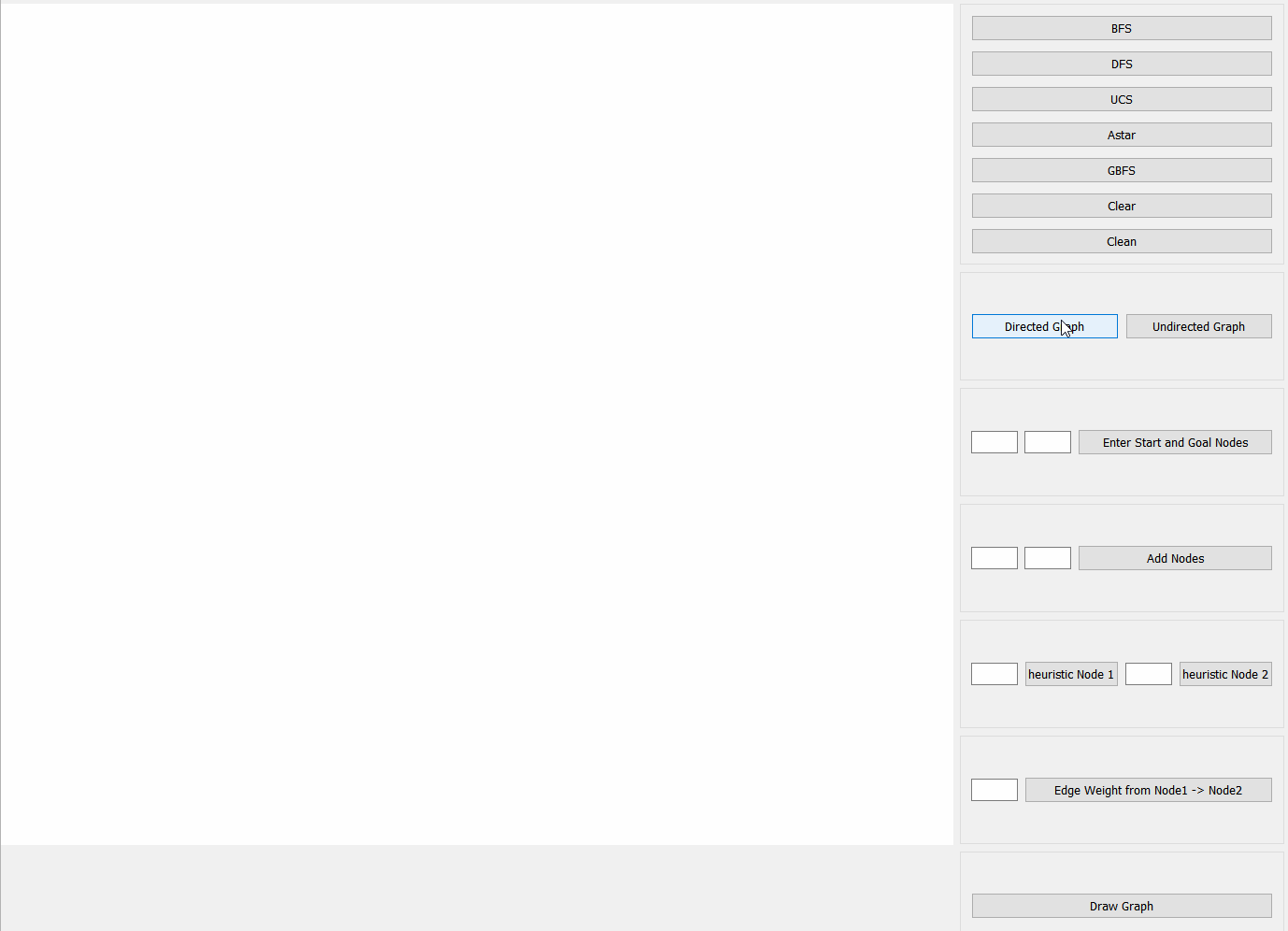


Figure 1 First step is creating the graph

## Adding nodes to the created graph

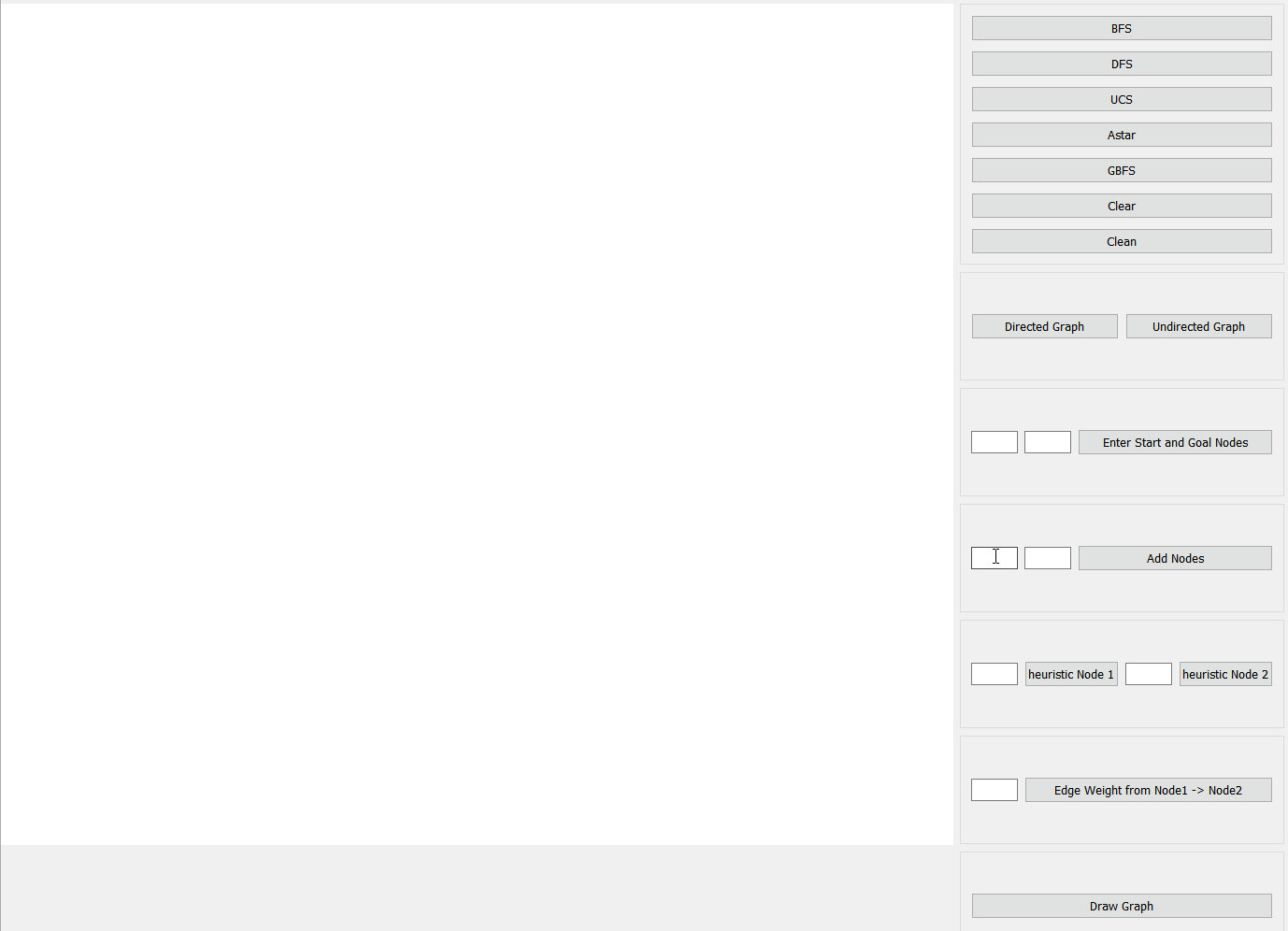


Figure 2 Adding the required nodes to the graph

**Precondition:** You must select the type of graph you want to create either directed or undirected.

## Adding heuristics to the node

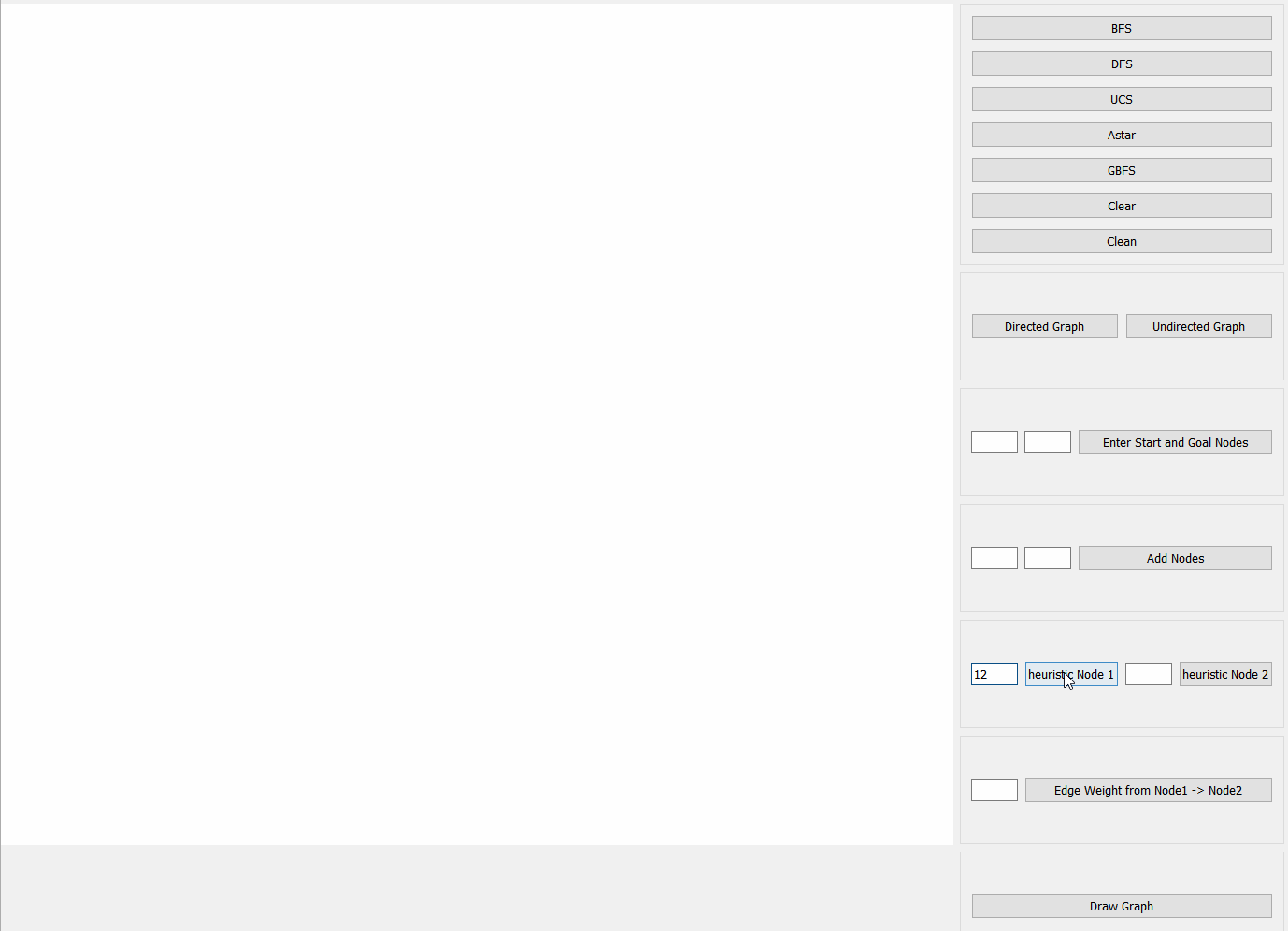


Figure 3 Adding heuristic to each node

**Precondition:** You created a graph and added two nodes to the graph, and you add **positive** heuristic values

## Adding edges between the nodes

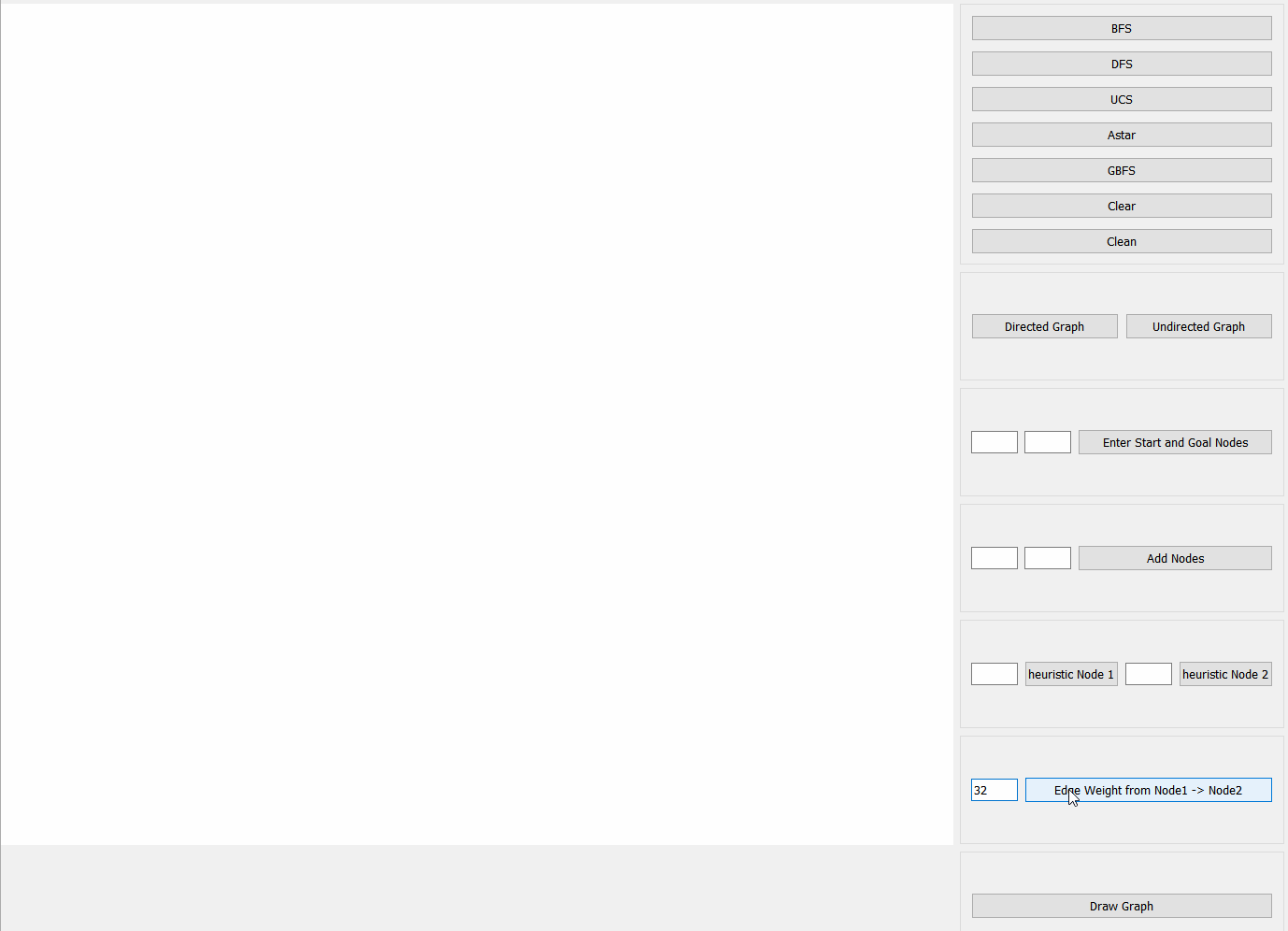


Figure 4 Adding the edge between nodes

IF YOU LEAVE THE TEXT BOX EMPTY THE EDGE WILL BE AN UNWEIGHTED EDGE**!!** (you wont be able to use astar or ucs algorithms)

**Precondition:** You created a graph and added two nodes to the graph.

*Optional 🡺 adding weight to the edge*

## Drawing the graph

Graphical user interface

Description automatically generated

Figure 5 Drawing the graph

**Precondition:** At least you have to create a graph and added two nodes to the graph.

*Optional 🡺 Added heuristic and/or added weight to edges*

## Adding required start and goal nodes

Chart, line chart

Description automatically generated

Figure 6 Adding Start and Search nodes

**Precondition:** At least you have to create a graph and added two nodes to the graph, the start and goal must be nodes in the graph.

*Optional 🡺 Added heuristic and/or added weight to edges*

## Applying the 5 Search algorithms

1. Chart, line chart

   Description automatically generatedBreadth First Search

Figure 7 BFS algorithm

### Precondition: You have to create a graph, added at least two nodes to the graph, and have specified the start and goal nodes.

*Optional 🡺 Added heuristic and/or added weight to edges*

1. Chart, line chart

   Description automatically generatedDepth First Search

Figure 8 DFS algorithm

### Precondition: You have to create a graph, added at least two nodes to the graph, and have specified the start and goal nodes.

*Optional 🡺 Added heuristic and/or added weight to edges*

1. Graphical user interface, application

   Description automatically generatedUniform Cost Search

Figure 9 UCS algorithm

### Precondition: You have to create a graph, added at least two nodes to the graph, and have specified the start and goal nodes. MUST: specify all the weights of the edges

*Optional 🡺 Added heuristic*

1. Graphical user interface, application

   Description automatically generatedA\* Search

Figure 10 A\* algorithm

### Precondition: You must create a graph, added at least two nodes to the graph, and have specified the start and goal nodes. MUST: specify all the weights of the edges, and specify the heuristics of each node

1. Graphical user interface, application

   Description automatically generatedGreedy Best First Search

Figure 11 GBFS algorithm

### Precondition: You must create a graph, added at least two nodes to the graph, and have specified the start and goal nodes. MUST: specify the heuristics of each node

*Optional 🡺 added the weights of each edge*

1. Chart, line chart

   Description automatically generatedClearing the canvas

Figure 12 Using Clear Function

### Precondition: You must create a graph, added at least two nodes to the graph.

*Optional* 🡺 *added edges and their weights of each edge, specify the heuristics of each node, specified the start and goal nodes.*

1. Chart, line chart

   Description automatically generatedUsing Clean/Deleting the graph

### Precondition: You must create a graph, added at least two nodes to the graph.

*Optional* 🡺 *added edges and their weights of each edge, specify the heuristics of each node, specified the start and goal nodes.*