DATA STRUCTURE VISUALIZATION

Preface

The data structure visualization application provides a user-friendly interface and clearly visual objects to display data structure for easy understanding how data structures store its data. Users can choose between many data structures according to their needs. Application also allows users to customize the appearance of visualization such as size, color of its components to suit their aesthetic.

Detailed description

Core:

The application uses C++ and SFML – a pure C++ with OpenGL library which is powerful for primarily developing 2D games- for implementation.

Primary content:

Data structure can be "visualized" in the application:

- Hash table: Linear probing, Quadratic probing, …
- Tree: AVL Tree, Min Heap, Max Heap, …
- Graph: Minimum Spanning Tree, Dijkstra, …

User interface (UI):

There are **TextBox** and **Button** wrapper class to contain characteristic of a text box or a button object to draw on the screen, which allow users to easily interact with the application.

Color and Size:

Users can change color and size of each components of the application by an option **Settings** at the top right corner of the application.

Frontend:

The most basic objects to draw or "appear" on the window are **CircleNode** and **Arrow** wrapper class, which represent Node and Edge of data structure.

Backend:

The most basic object to control and organize data in the behind is **Node** class, which contains an array of value, an array of pointers to others **Node**s and a pointer to a **CircleNode**, which is use for animation.

Animation:

There are **Status** and **Handle** classes to control step-by-step of each operator on data structure. **Status** contains a pointer to a **CircleNode** or an **Arrow** and all status of them at a certain step such as colors, positions, … **Handle** contains an array of an array of **Status** named **List** in private, **List[i]**

is status of data structure at step i of the operator. Handle has 3 primary functions are stepback, stepforw, run; which allow users to step back or forward step-by-step or keep doing the animation to the end of the operator. These function will be performed by an independent thread.

Operator:

The application provides these following operators, depending on which data structures they choose:

+ Initialize: Users can choose to create data structure with random data or from an external file provided for the application.

+ Add: Users can add new data to data structure by giving new entry data to the application.

+ Remove: Users can remove existing data by giving a suitable “key” to the application.

+ Search: Users can search for existing data by providing a suitable search “key” for the application.

+ Graph algorithm: Application will do some basis algorithms with the given graph.