CSE 2025 – DATA STRUCTURES
PROJECT 2 REPORT

A)Functions

```
Node *readInputFile();
```

Gets the file name as an input from the user , then reads and creates a linked list for Adj list represetation.

```
void dijkstra(Node *head , char src , char dst , Node* paths[]);
```

Dijkstra Algorithm method. *head is the Adj list pointer , src is source vertex taken as input from user , dst is destination vertex taken as input from user , paths array is the paths needs to be taken in order to reach corresponig verticices from source vertex.

```
void updateHeap(Heap *heap_ptr , char dest);
```

Updates the distance of the destination vertex within the heap. Then modifies the min heap to keep the heap structure. heap_ptr is the pointer to heap , dest is destination vertex.

```
int findIndex(Heap *heap_ptr,Node *node_ptr ,char v);
```

Finds the index of vertex v in the given list. heap_ptr is in heap type , node_ptr is in node type and searches in Adj list.

```
char extractMin(int final_dist[]);
```

Extracts the vertex with minimum distance to source vertex , that is root of min heap. Final_dist array is the distance array of the verticicies.

```
void heapify(int index);
```

Called after extractMin. Make sures that heap structure is still intact after root is extracted.

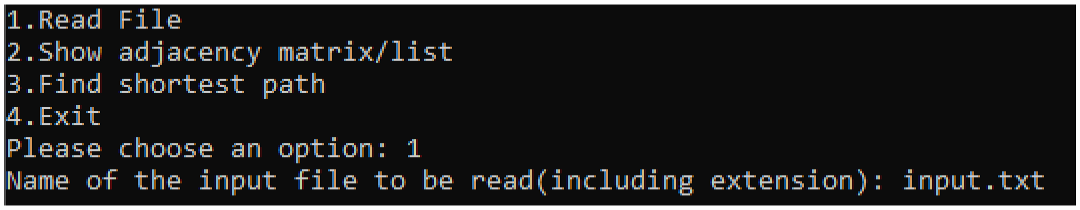
```
int isEmpty();
```

Checks if min_heap is empty or not.

```
Node *copyList(Node* path_ptr);
```

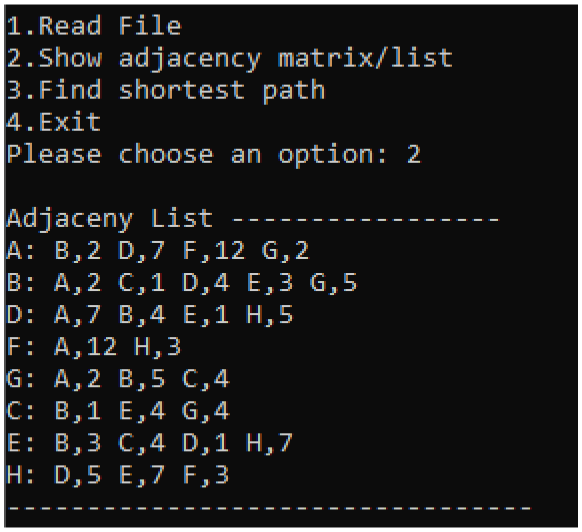
Makes a copy of the given linked list and returns a pointer to it.

1.)Read File



```
1.Read File
2.Show adjacency matrix/list
3.Find shortest path
4.Exit
Please choose an option: 1
Name of the input file to be read(including extension): input.txt
```

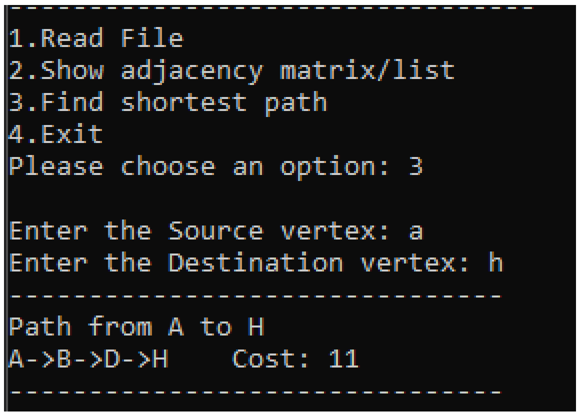
2.)Show Adj List



```
1.Read File
2.Show adjacency matrix/list
3.Find shortest path
4.Exit
Please choose an option: 2

Adjaceny List -----------------
A: B,2 D,7 F,12 G,2
B: A,2 C,1 D,4 E,3 G,5
D: A,7 B,4 E,1 H,5
F: A,12 H,3
G: A,2 B,5 C,4
C: B,1 E,4 G,4
E: B,3 C,4 D,1 H,7
H: D,5 E,7 F,3
---------------------------------
```

3.)Find Shortest Path



```
--------------------------------
1.Read File
2.Show adjacency matrix/list
3.Find shortest path
4.Exit
Please choose an option: 3

Enter the Source vertex: a
Enter the Destination vertex: h
-------------------------------
Path from A to H
A->B->D->H    Cost: 11
-------------------------------
```

References

https://www.geeksforgeeks.org/dijkstras-algorithm-for-adjacency-list-representation-greedy-algo-8/

https://www.codingame.com/playgrounds/1608/shortest-paths-with-dijkstras-algorithm/keeping-track-of-paths