**comsats university islambad**

Data Structures and Algorithms

Semester Project

The Mandalorian

A program for a security agent to take down criminals on the map

**Submitted By:**

Muhammad Umair

FA19-BCS-057

BCS-3B

**Submitted To:**

**Dr. Inayat Ur Rehman**

The Mandalorian is­­­­ a bounty hunter achieving its bounties, just like a security agent hunting for criminals. His job is to hunt more and more bounties (criminals) on the map using his weapons and he will receive award for that.

Our program will help any mandalorian to catch the criminals more efficiently. There is a store of weapons which shows all the available weapons that he can buy and put them in his list of weapons. Each travel and bounty will cause him lose some health. He can also heal himself by buying some heal if he is low on health. On achieving a bounty, the travel journey is displayed along with credits and health remaining.

# **Data Structures Used:**

# **Structs:**

Structs are used to define the entities and their properties. Structs are best way to define small entities containing more than one variables. Three structs are used:

### Weapons:

* + Name
  + Damage
  + Price

### Bounty

* + Name
  + Reward
  + Health
  + Location
  + Distance

### Mandalorian

* + Health
  + Weapon List
  + Bounties Queue
  + Distance Travelled
  + Achieved Bounties List

# **Linked Lists:**

Linked Lists are used to store weapon data and make achieved bounties list. We used linked list on these because these two things are dynamic. We are not aware about the size of the list as we do not know how many bounties the user will achieve or how many weapons will he buy. Inserting data in it is O (1) operation as we are inserting new data in the end of the list and displaying it which is O (n) operation.

# **Priority Queue:**

The Bounties are stored in a priority queue because we want to achieve more reward efficiently by placing the highest priority bounty in front. Priority is determined by reward to distance ratio. The high reward and low distance is the most preferred. We are using priority queue of fixed size (in array) because there is a limit of bounties that user can have at a time.

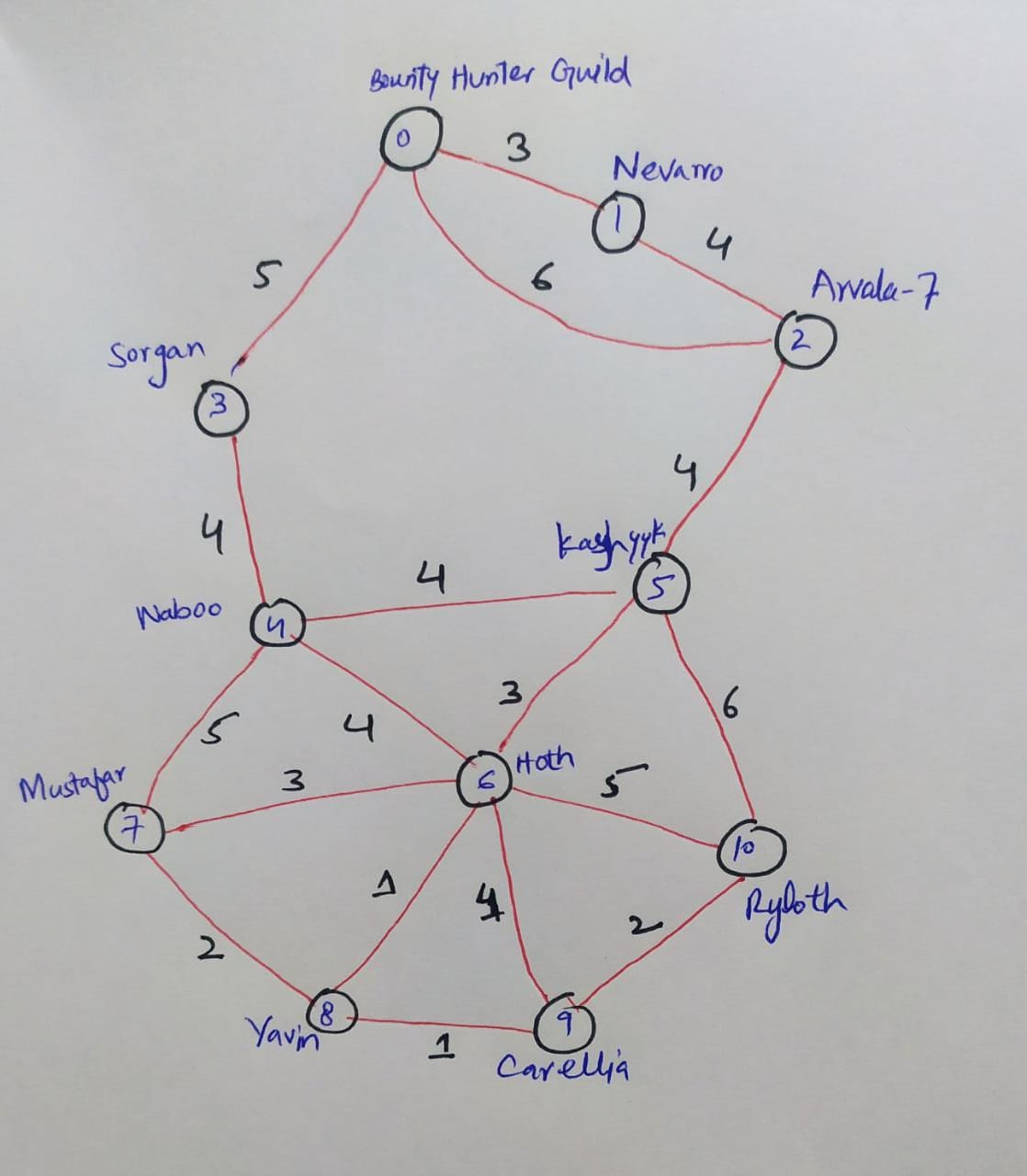
# **Hash Table:**

Hash table is used to store all the available weapons on the store. The reason for using hash table is because of its fast searching which on average case is O (1). The user when buys a weapon from store it instantly finds the key using hash function and gives us the weapon. The size of our hash table is fixed because the store can accommodate limited number of weapons. The collisions are handled by linear probing to keep the size of table fixed.

# **Graphs:**

Graph is used to store the map of all the planets. The graph is undirected which means its edges are bi-directional we can go forward and also come back. It is a weighted graph in the form of matrix keeping track of distances and paths between the planets. We used graph to locate our bounty.

**Graph used in the project:**



# **Array List:**

Array list is used to store the names of planets on map as our graph is in matrix form we needed some data structure to store the names of the planets. The number of planets in the map are fixed so we had no problem of fixed size of array list. Main reason for using array list is because of its fast access to elements which is just O (1) operation.

# **Algorithms Used:**

# **Linear Search:**

Linear search is used to get a weapon from the weapon linked list. It is also used to find the planet in planets array. We used linear search because we are not storing data in sorted form.

# **Insertion Sort:**

When new element is added into the priority queue, to maintain its order the new element is inserted according to its priority thus keeping the queue sorted according to our priority. Half part of insertion sort is being used.

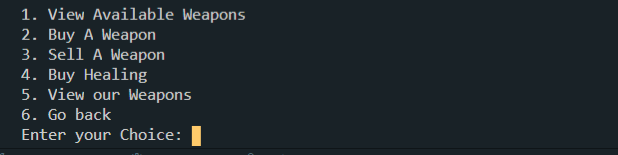
# **Dijkstra Algorithm:**

Dijkstra Algorithm is used to find the shortest distance from our source to all the destination planets and the path to the destination planet so we know how much distance will it take and we will later use that distance to set our highest priority bounty. We will travel using the shortest path and reach our destination.

# **Main Menu:**

## **Go to Store:**

Inside store we can:



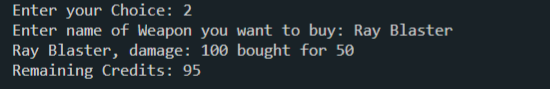
1. **View A­­vailable Weapons:**

When user enters this option, a list of all the available weapons are displayed to user. All available weapons are stored in a **Hash Table.**

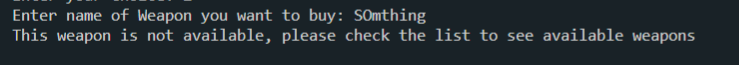


1. **But a weapon**

User can buy a weapon which is available in the list of weapons. When a weapon is bought it goes into list of weapons of Mandalorian stored in a **linked list.**

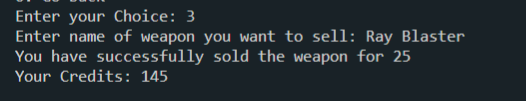


**Invalid Input:**

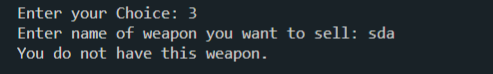


1. **Sell a weapon:**

User can sell a weapon for half of its bought price. When a weapon is sold, it is removed from user’s list of weapons and sold credits are added to user credits.

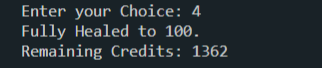


**Invalid Input:**



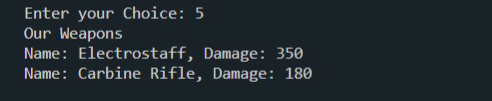
1. **Buy Healing:**

User can buy healing, if he has enough credits he will be healed to full else he will be healed for the credits he has.

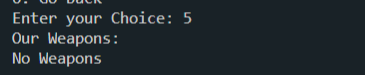


1. **View our list of weapons:**

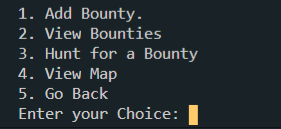
It simply displays the list of weapons user has.



**No Weapons:**

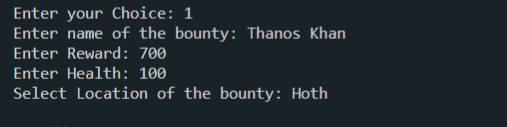


## **Take down Bounties:**

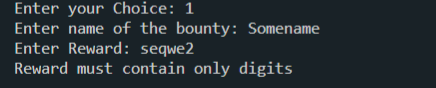


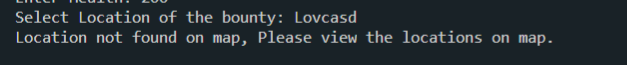
1. **Add a bounty:**

User can add a bounty providing its name, reward, health and location. On the basis of that it is added into Mandalorian’s bounties which is a **priority queue.**



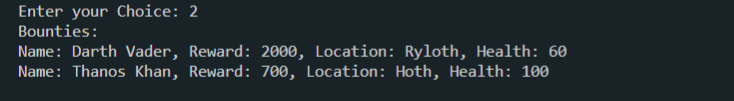
**Invalid Input:**





1. **View Bounties:**

This view all the bounties that have been added to the bounties queue.



1. **Hunt for a bounty:**

This option hunts down the highest priority bounty first if there is any.

If we do not have enough damage we cannot hunt for that bounty. We then have to get more damage from the store.

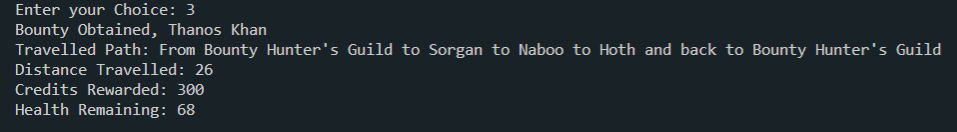


For every hunt we lose our health. If we do not have enough health then we cannot take down any bounty.



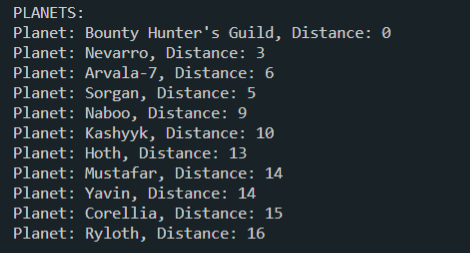
We can heal back to full from store if we have enough credits.

When we have everything to get to the bounty, then we can just travel to that place and hunt it down. The travelled path and distance will be displayed. And reward of the bounty will be added to the credits.



1. **View Map:**

This view all the planets on the map and their distances.



## **View Our Status:**

This views our status, our current health, distance travelled, achieved bounties if any, weapons if any and total credits.

