Assignment No-01(Group A)

Title: - Implement depth first search algorithm and Breadth First Search algorithm

Objectives:-

1. Understand the implementation of depth first search algorithm
2. Understand the implementation of Breadth First Search algorithm

Problem Statement:-

Implement depth first search algorithm and Breadth First Search algorithm, Use an undirected graph and develop a recursive algorithm for searching all the vertices of a graph or tree data structure

Software and Hardware requirements:-

1. **Operating system:** Linux- Ubuntu 16.04 to 17.10, or Windows 7 to 10,
2. **RAM-** 2GB RAM (4GB preferable)
3. You have to install **Python3** or higher version

Theory-

1. Depth First Search

What do we do once have to solve a maze? We tend to take a route, keep going until we discover a dead end. When touching the dead end, we again come back and keep coming back till we see a path we didn't attempt before. Take that new route. Once more keep going until we discover a dead end. Take a come back again… This is exactly how Depth-First Search works.

The Depth-First Search is a recursive algorithm that uses the concept of backtracking. It involves thorough searches of all the nodes by going ahead if potential, else by backtracking. Here, the word backtrack means once you are moving forward and there are not any more nodes along the present path, you progress backward on an equivalent path to seek out nodes to traverse. All the nodes are progressing to be visited on the current path until all the unvisited nodes are traversed after which subsequent paths are going to be selected.

DFS Algorithm

A standard Depth-First Search implementation puts every vertex of the graph into one in all 2 categories:

1) Visited 2) Not Visited.

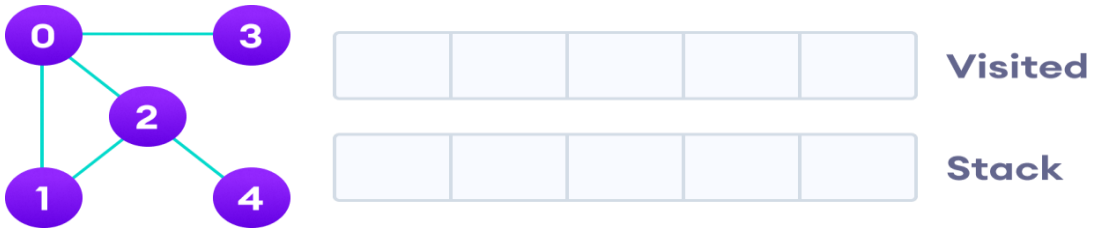
The only purpose of this algorithm is to visit all the vertex of the graph avoiding cycles.

The DSF algorithm follows as:

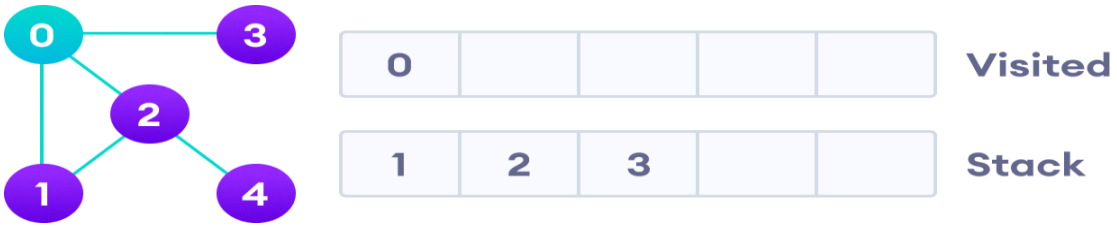
1. We will start by putting any one of the graph's vertex on top of the stack.
2. After that take the top item of the stack and add it to the visited list of the vertex.
3. Next, create a list of that adjacent node of the vertex. Add the ones which aren't in the visited list of vertexes to the top of the stack.
4. Lastly, keep repeating steps 2 and 3 until the stack is empty.

Depth First Search Example

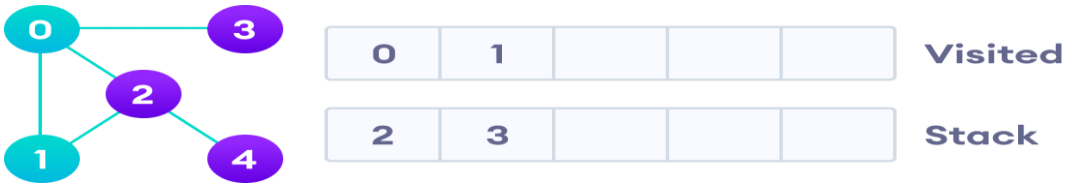
Let's see how the Depth First Search algorithm works with an example. We use an undirected graph with 5 vertices.



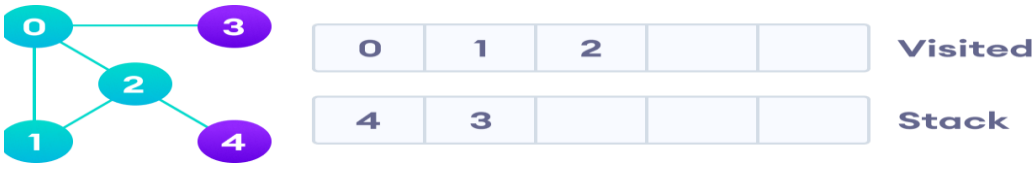
We start from vertex 0, the DFS algorithm starts by putting it in the Visited list and putting all its adjacent vertices in the stack.



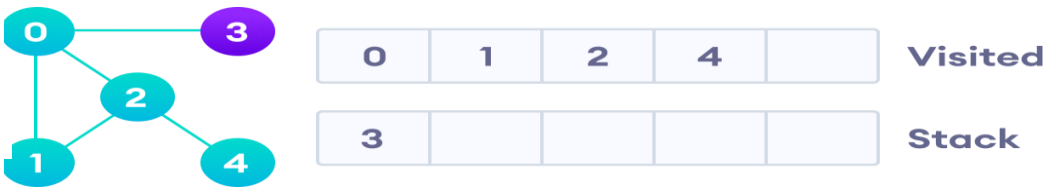
Next, we visit the element at the top of stack i.e. 1 and go to its adjacent nodes. Since 0 has already been visited, we visit 2 instead



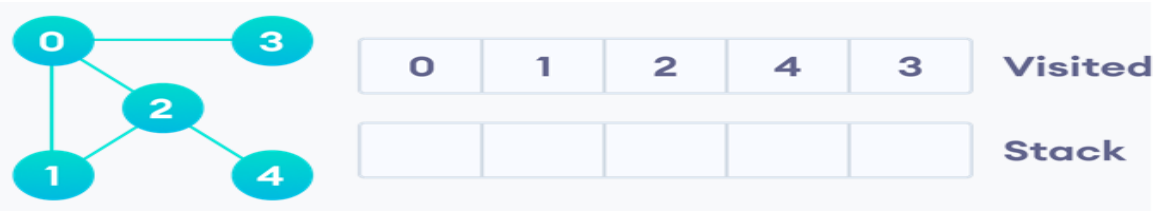
Vertex 2 has an unvisited adjacent vertex in 4, so we add that to the top of the stack and visit it.



Vertex 2 has an unvisited adjacent vertex in 4, so we add that to the top of the stack and visit it.4



After we visit the last element 3, it doesn't have any unvisited adjacent nodes, so we have completed the Depth First Traversal of the graph.



After we visit the last element 3, it doesn't have any unvisited adjacent nodes, so we have completed the Depth First Traversal of the graph.

Application of DFS Algorithm

1. For finding the path
2. To test if the graph is bipartite
3. For finding the strongly connected components of a graph
4. For detecting cycles in a graph.

Breadth-First Search

Breadth-First Search (BFS) is an algorithm used for traversing graphs or trees. Traversing means visiting each node of the graph. Breadth-First Search is a recursive algorithm to search all the vertices of a graph or a tree. BFS in python can be implemented by using data structures like a dictionary and lists. Breadth-First Search in tree and graph is almost the same. The only difference is that the graph may contain cycles, so we may traverse to the same node again.

BFS Algorithm

Breadth-first search is the process of traversing each node of the graph, a standard BFS algorithm traverses each vertex of the graph into two parts:

1) Visited 2) Not Visited.

So, the purpose of the algorithm is to visit all the vertex while avoiding cycles.

BFS starts from a node, then it checks all the nodes at distance one from the beginning node, then it checks all the nodes at distance two, and so on. So as to recollect the nodes to be visited, BFS uses a queue.

The steps of the algorithm work as follow:

1. Start by putting any one of the graph's vertices at the back of the queue.
2. Now take the front item of the queue and add it to the visited list.
3. Create a list of that vertex's adjacent nodes. Add those which are not within the visited list to the rear of the queue.
4. Keep continuing steps two and three till the queue is empty.

Many times, a graph may contain two different disconnected parts and therefore to make sure that we have visited every vertex, we can also run the BFS algorithm at every node.

Explanation:

1. Create a graph.
2. Initialize a starting node.
3. Send the graph and initial node as parameters to the bfs function.
4. Mark the initial node as visited and push it into the queue.
5. Explore the initial node and add its neighbours to the queue and remove the initial node from the queue.
6. Check if the neighbours node of a neighbouring node is already visited.
7. If not, visit the neighbouring node neighbours and mark them as visited.
8. Repeat this process until all the nodes in a graph are visited and the queue becomes empty.

Advantages of BFS

1. It can be useful in order to find whether the graph has connected components or not.
2. It always finds or returns the shortest path if there is more than one path between two vertices.

Disadvantages of BFS

1. The execution time of this algorithm is very slow because the time complexity of this algorithm is exponential.
2. This algorithm is not useful when large graphs are used.

Conclusion

Depth-First Search and Breadth-First Search (BFS) are used to traverse the graph or tree. We implemented Depth-First Search and Breadth-First Search (BFS) in python for searching all the vertices of a graph or tree data structure.

Assignment No-02(Group A)

Title: - Implement A star Algorithm for any game search problem.

Objectives:-

1. Understand the implementation of A star Algorithm

Problem Statement:-

Implement A star Algorithm for any game search problem.

Software and Hardware requirements:-

1. **Operating system:** Linux- Ubuntu 16.04 to 17.10, or Windows 7 to 10,
2. **RAM**- 2GB RAM (4GB preferable)
3. You have to install **Python3** or higher version

Theory-

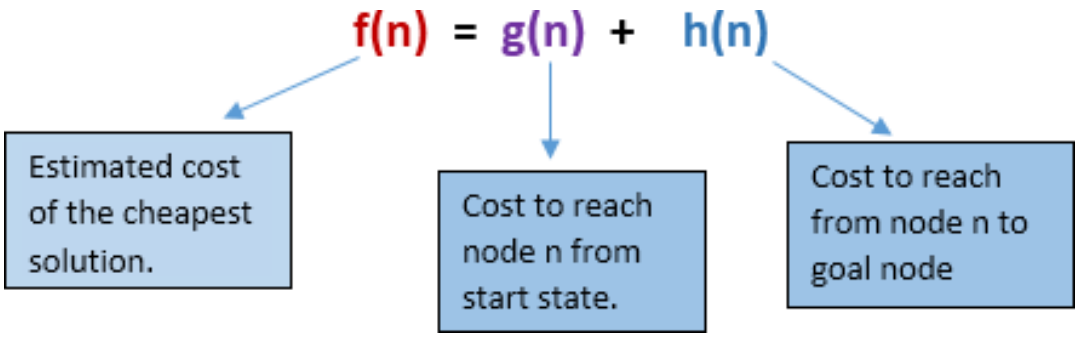
A* Search

A* search is the most commonly known form of best-first search. It uses heuristic function h(n), and cost to reach the node n from the start state g(n).

It has combined features of UCS and greedy best- first search, by which it solve the problem efficiently. A* search algorithm finds the shortest path through the search space using the heuristic function.

This search algorithm expands less search tree and provides optimal result faster. A* algorithm is similar to UCS except that it uses g(n)+h(n) instead of g(n).

In A* search algorithm, we use search heuristic as well as the cost to reach the node. Hence we can combine both costs as following, and this sum is called as a fitness number.



Algorithm of A* search:

Step1: Place the starting node in the OPEN list.

Step 2: Check if the OPEN list is empty or not, if the list is empty then return failure and stops.

Step 3: Select the node from the OPEN list which has the smallest value of evaluation function (g+h), if node n is goal node then return success and stop, otherwise

Step 4: Expand node n and generate all of its successors, and put n into the closed list. For each successor n', check whether n' is already in the OPEN or CLOSED list, if not then compute evaluation function for n' and place into Open list.

Step 5: Else if node n' is already in OPEN and CLOSED, then it should be attached to the back pointer which reflects the lowest g(n') value.

Step 6: Return to Step 2.

Advantages:

1. A* search algorithm is the best algorithm than other search algorithms.
2. A* search algorithm is optimal and complete.
3. This algorithm can solve very complex problems.

Disadvantages:

1. It does not always produce the shortest path as it mostly based on heuristics and approximation.
2. A* search algorithm has some complexity issues.
3. The main drawback of A* is memory requirement as it keeps all generated nodes in the memory, so it is not practical for various large-scale problems.

Conclusion

Implement A star Algorithm for any game search problem in python for searching path.

Assignment No-03(Group A)

Title: - Implement Greedy search algorithm for Prim's Minimal Spanning Tree Algorithm

Objectives:-

1. Understand the concept of Greedy search algorithm.
2. Understand the implementation of Prim's Minimal Spanning Tree Algorithm

Problem Statement:-

Implement Greedy search algorithm for any of the following application:

I. Selection Sort
II. Minimum Spanning Tree
III. Single-Source Shortest Path Problem
IV. Job Scheduling Problem
V. Prim's Minimal Spanning Tree Algorithm
VI. Kruskal's Minimal Spanning Tree Algorithm
VII. Dijkstra's Minimal Spanning Tree Algorithm

Software and Hardware requirements:-

4. **Operating system:** Linux- Ubuntu 16.04 to 17.10, or Windows 7 to 10,
5. **RAM-** 2GB RAM (4GB preferable)
6. You have to install **Python3** or higher version

Theory-

Minimum Spanning Tree?

As we all know, the graph which does not have edges pointing to any direction in a graph is called an undirected graph and the graph always has a path from a vertex to any other vertex. A spanning tree is a subgraph of the undirected connected graph where it includes all the nodes of the graph with the minimum possible number of edges.

Remember, the subgraph should contain each and every node of the original graph. If any node is missed out then it is not a spanning tree and also, the spanning tree doesn't contain cycles. If the graph has n number of nodes, then the total number of spanning trees created from a complete graph is equal to n^(n- 2).

In a spanning tree, the edges may or may not have weights associated with them. Therefore, the spanning tree in which the sum of edges is minimum as possible then that spanning tree is called the minimum spanning tree. One graph can have multiple spanning-tree but it can have only one unique minimum spanning tree.

There are two different ways to find out the minimum spanning tree from the complete graph i.e Kruskal's algorithm and Prim's algorithm. Let us study prim's algorithm in detail below:

Prim's Algorithm?

Prim's algorithm is a minimum spanning tree algorithm which helps to find out the edges of the graph to form the tree including every node with the minimum sum of weights to form the minimum spanning tree.

Prim's algorithm starts with the single source node and later explore all the adjacent nodes of the source node with all the connecting edges. While we are exploring the graphs, we will choose the edges with the minimum weight and those which cannot cause the cycles in the graph.

Prim's Algorithm for Minimum Spanning Tree

Prim's algorithm basically follows the greedy algorithm approach to find the optimal solution. To find the minimum spanning tree using prim's algorithm, we will choose a source node and keep adding the edges with the lowest weight.

The algorithm is as given below:

•Initialize the algorithm by choosing the source vertex

•Find the minimum weight edge connected to the source node and another node and add it to the tree

•Keep repeating this process until we find the minimum spanning tree

Pseudocode

T = ∅;

M = { 1 };

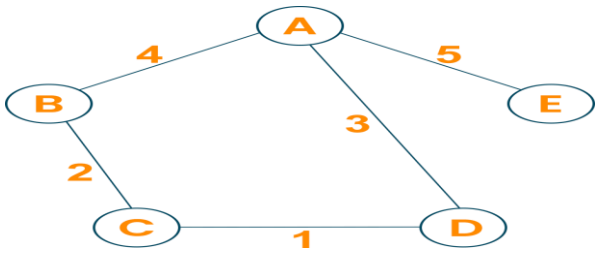
while (M ≠ N)

let (m, n) be the lowest cost edge such that m ∈ M and n ∈ N - M; T = T ∪ {(m, n)}

M = M ∪ {n}

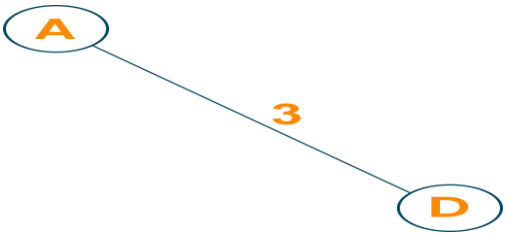
Here we create two sets of nodes i.e M and M-N. M set contains the list of nodes that have been visited and the M-N set contains the nodes that haven't been visited. Later, we will move each node from M to M-N after each step by connecting the least weight edge.

Example Let us consider the below-weighted graph

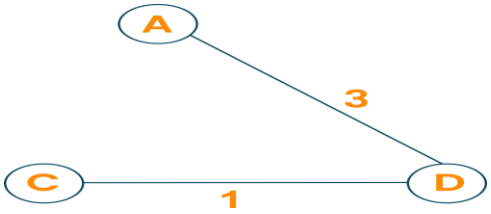


Later we will consider the source vertex to initialize the algorithm

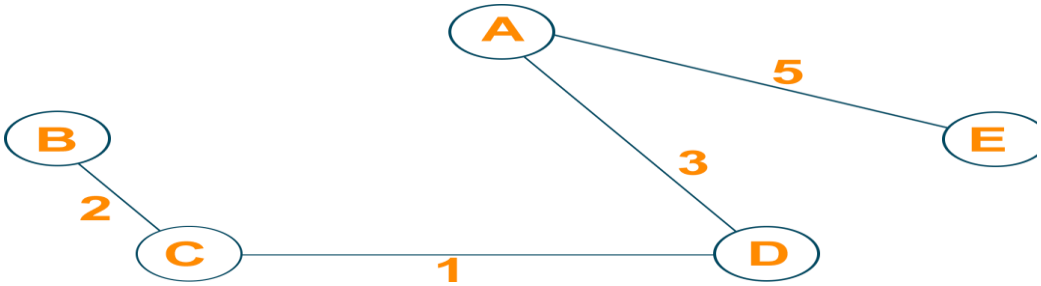
Now, we will choose the shortest weight edge from the source vertex and add it to finding the spanning tree.



Then, choose the next nearest node connected with the minimum edge and add it to the solution. If there are multiple choices then choose anyone.



Continue the steps until all nodes are included and we find the minimum spanning tree.



Time Complexity:

The running time for prim's algorithm is O(VlogV + ElogV) which is equal to O(ElogV) because every insertion of a node in the solution takes logarithmic time. Here, E is the number of edges and V is the number of vertices/nodes. However, we can improve the running time complexity to O(E + logV) of prim's algorithm using Fibonacci Heaps.

Applications

- Prim's algorithm is used in network design
- It is used in network cycles and rail tracks connecting all the cities
- Prim's algorithm is used in laying cables of electrical wiring
- Prim's algorithm is used in irrigation channels and placing microwave towers
- It is used in cluster analysis
- Prim's algorithm is used in gaming development and cognitive science
- Pathfinding algorithms in artificial intelligence and traveling salesman problems make use of prim's algorithm.

Conclusion

As we studied, the minimum spanning tree has its own importance in the real world, it is important to learn the prim's algorithm which leads us to find the solution to many problems. When it comes to finding the minimum spanning tree for the dense graphs, prim's algorithm is the first choice.

Assignment No-04(Group B)

Title: - Implement Branch and Bound and Backtracking for n-queens problem.

Objectives:-

1. Understand the concept and implementation of Branch and Bound for n-queens problem.
2. Understand the concept and implementation of Backtracking for n-queens problem.

Problem Statement:-

Implement a solution for a Constraint Satisfaction Problem using Branch and Bound and Backtracking for n-queens problem or a graph coloring problem.

Software and Hardware requirements:-

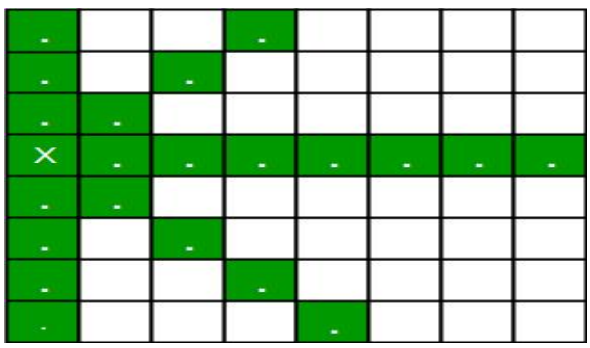
7. **Operating system:** Linux- Ubuntu 16.04 to 17.10, or Windows 7 to 10,
8. **RAM-** 2GB RAM (4GB preferable)
9. You have to install **Python3** or higher version

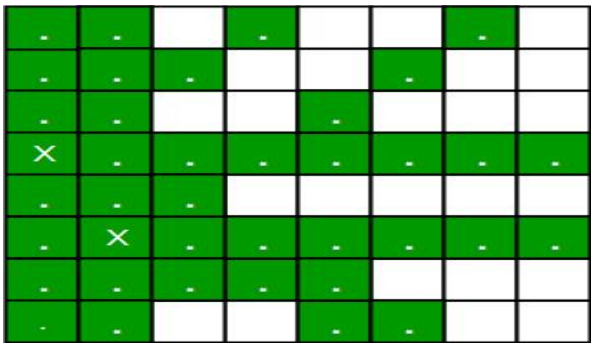
Theory-

The **N queens puzzle** is the problem of placing N chess queens on an N×N chessboard so that no two queens threaten each other. Thus, a solution requires that no two queens share the same row, column, or diagonal.

Backtracking Algorithm for N-Queen is already discussed here. In backtracking solution we backtrack when we hit a dead end. **In Branch and Bound solution, after building a partial solution, we figure out that there is no point going any deeper as we are going to hit a dead end.**

"The idea is to place queens one by one in different columns, starting from the leftmost column. When we place a queen in a column, we check for clashes with already placed queens. In the current column, if we find a row for which there is no clash, we mark this row and column as part of the solution. If we do not find such a row due to clashes, then we backtrack and return false."





1. For the 1st Queen, there are total 8 possibilities as we can place 1st Queen in any row of first column. Let's place Queen 1 on row 3.

2. After placing 1st Queen, there are 7 possibilities left for the 2nd Queen. But wait, we don't really have 7 possibilities. We cannot place Queen 2 on rows 2, 3 or 4 as those cells are under attack from Queen 1. So, Queen 2 has only 8 – 3 = 5 valid positions left.
3. After picking a position for Queen 2, Queen 3 has even fewer options as most of the cells in its column are under attack from the first 2 Queens.

We need to figure out an efficient way of keeping track of which cells are under attack. In previous solution we kept an 8-by-8 Boolean matrix and update it each time we placed a queen, but that required linear time to update as we need to check for safe cells. Basically, we have to ensure 4 things:

1. No two queens share a column.
2. No two queens share a row.
3. No two queens share a top-right to left-bottom diagonal.
4. No two queens share a top-left to bottom-right diagonal.

Number 1 is automatic because of the way we store the solution. For number 2, 3 and 4, we can perform updates in O(1) time. The idea is to keep **three Boolean arrays that tell us which rows and which diagonals are occupied**.

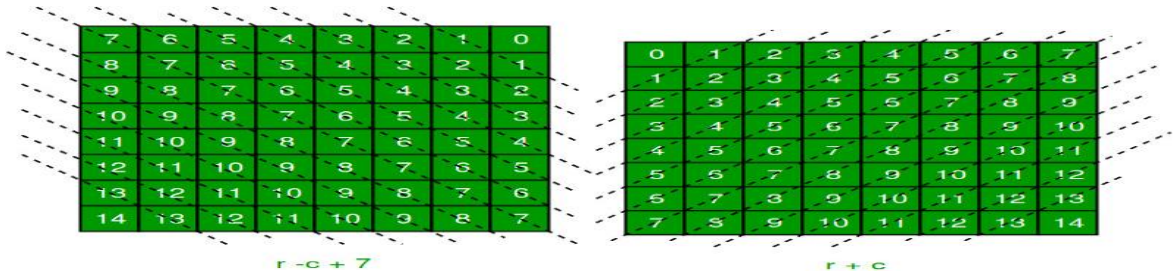
Lets do some pre-processing first. Let's create two N x N matrix one for / diagonal and other one for \ diagonal. Let's call them slashCode and backslashCode respectively. The trick is to fill them in such a way that two queens sharing a same /diagonal will have the same value in matrix slashCode, and if they share same diagonal, they will have the same value in backslashCode matrix.

For an N x N matrix, fill slashCode and backslashCode matrix using below formula

slashCode[row][col] = row + col

backslashCode[row][col] = row – col + (N-1)

Using above formula will result in below matrices



The 'N – 1' in the backslash code is there to ensure that the codes are never negative because we will be using the codes as indices in an array.

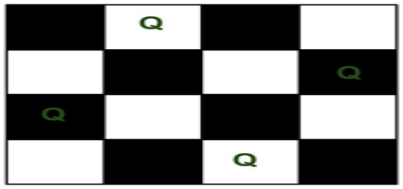
Now before we place queen i on row j, we first check whether row j is used (use an array to store row info). Then we check whether slash code (j + i) or backslash code (j – i + 7) are used (keep two arrays that will tell us which diagonals are occupied). If yes, then we have to try a different location for queen i. If not, then we mark the row and the two diagonals as used and recurse on queen i + 1. After the recursive call returns and before we

try another position for queen i, we need to reset the row, slash code and backslash code as unused again

Backtracking;-

The N Queen is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other. For example, following is a solution for 4 Queen problem.

The expected output is a binary matrix which has 1s for the blocks where queens are placed. For example, following is the output matrix for above 4 queen solution.



{ 0, 1, 0, 0}
{ 0, 0, 0, 1}
{ 1, 0, 0, 0}
{ 0, 0, 1, 0}

Naive Algorithm

Generate all possible configurations of queens on board and print a configuration that satisfies the given constraints.

```
 while there are untried configurations
{   generate the next configuration
   if queens don't attack in this configuration then
   {       print this configuration;
   }
}
```

Backtracking Algorithm

The idea is to place queens one by one in different columns, starting from the leftmost column. When we place a queen in a column, we check for clashes with already placed queens. In the current column, if we find a row for which there is no clash, we mark this row and column as part of the solution. If we do not find such a row due to clashes then we backtrack and return false.

1) Start in the leftmost column

2) If all queens are placed return true

3) Try all rows in the current column.

Do following for every tried row.

a) If the queen can be placed safely in this row then mark this [row, column] as part of the solution and recursively check if placing queen here leads to a solution.

b) If placing the queen in [row, column] leads to a solution then return true.

c) If placing queen doesn't lead to a solution then unmark this [row, column] (Backtrack) and go to step (a) to try other rows.

4) If all rows have been tried and nothing worked, return false to trigger backtracking.

Conclusion:- In these way we have implemented a solution for a Constraint Satisfaction Problem using Branch and Bound and Backtracking for n-queens problem.

Assignment No-05(Group B)

Title: - Develop an elementary catboat.

Objectives:-

1. Understand the concept of catboat.

Problem Statement:-

Develop an elementary catboat for any suitable customer interaction application.

Software and Hardware requirements:-

10. **Operating system:** Linux- Ubuntu 16.04 to 17.10, or Windows 7 to 10,
11. **RAM-** 2GB RAM (4GB preferable)
12. You have to install **Python3** or higher version or Turbo C++ or JDK.

Theory-

What is a chatbot?

A chatbot is a computer program designed to have a conversation with human beings over the internet. It's also known as conversational agents, which communicate and collaborate with human users, through text messaging, in order to accomplish a specific task. Basically, there are two types of chatbots. The one that uses Artificial Intelligence, and another one is based on multiple choice scripts.

Both types of chatbots aim to create a more personalized content experience for the users, whether that's while watching a video, reading articles or buying new shoes.

These Chatbots hold the promise of being the next generation of technology that people use to interact online with business enterprises. These Chatbots offer a lot of advantages, one of which is that, because Chatbots communicate using a natural language, users don't need to learn yet another new website interface, to get comfortable with the unavoidable quirks.

Chatbots are capable to interpret human speech, and decide which information is being sought. Artificial intelligence is getting smarter each day, and brands that are integrating Chatbots with the artificial intelligence, can deliver one-to-one individualized experiences to consumers.

Why chatbot?

Chatbots can be useful in many aspects of the customer experience, including providing customer service, presenting product recommendations and engaging customers through targeted marketing campaigns. If a customer has an issue with a product, she can connect with a chatbot to explain the situation and the chatbot can input that information to provide a recommendation of how to fix the product. On the recommendation side, chatbots can be used to share popular products with customers that they might find useful and can act

as a sort of personal shopper or concierge service to find the perfect gift, meal or night out for a customer with just a few basic questions. Brands are also using chatbots to connect their customers with thought leaders and add personality to their products. In all cases, brands seem to be having great success and experiencing increased engagement and revenue.

Chatbots are easy to use and many customers prefer them over calling a representative on the phone because it tends to be faster and less invasive. They can also save money for companies and are easy to set up.

Chatbots are relatively new and most companies haven't implemented them yet, it's only natural that users are interested in them. Hence, people want to discover what chatbots can and cannot do.

The number of businesses using chatbots has grown exponentially. Chatbots have increased from 30,000 in 2016 to over 100,000 today. Every major company has announced their own chatbot and 60% of the youth population uses them daily.

These statistics prove that chatbots are the new-gen tech. No more waiting for the right time to incorporate them into your business. The time is now. By the year 2020, nearly 80% of businesses will have their own chatbot.

Billions of people are already using chatbots, so it's time your business did too.

Benefits of chatbot?

1. Available 24*7:

I'm sure most of you have experienced listening to the boring music playing while you're kept on hold by a customer care agent. On an average people spend 7 minutes until they are assigned to an agent. Gone are the days of waiting for the next available operative. Bots are replacing live chat and other forms of contact such as emails and phone calls.

Since chat bots are basically virtual robots they never get tired and continue to obey your command. They will continue to operate every day throughout the year without requiring to take a break. This improves your customer satisfaction and helps you rank highly in your sector.

2. Handling Customers:

We humans are restricted to the number of things we can do at the same time. A study suggests that humans can only concentrate on 3–4 things at the same time. If it goes beyond that you are bound to meet errors.

Chatbots on the other hand can simultaneously have conversations with thousands of people. No matter what time of the day it is or how many people are contacting you, every single one of them will be answered instantly. Companies like Taco Bell and Domino's are already using chatbots to arrange delivery of parcels.

3. Helps you Save Money:

If you are a business owner you are bound have a lot of employees who need to be paid for the work they do. And these expenses just keep adding up as business grows. Chatbots are a one time investment which helps businesses reduce down on staff required.

You could integrate a customer support chatbot in your business to cater to simple queries of customers and pass on only the complex queries to customer support agents.

4. Provides 100% satisfaction to customers:

Humans react to others based on their mood and emotions. If a agent is having a good attitude or is in good mood he will most probably talk to customers in a good way. In contrary to this the customer will not be satisfied.

Whereas chatbots are bound by some rules and obey them as long as they're programmed to. They always treat a customer in the most polite and perfect way no matter how rough the person is. Also, in the travel and hospitality industry where travelers do not speak the same language, a bot can be trained to communicate in the language of the traveler.

5. Automation of repetitive work:

Lets be honest, no one likes doing the same work again and again over brief period of time. In the case of humans, such tasks are prone to errors. Chatbots now help automate tasks which are to be done frequently and at the right time.

Also, now there are numerous slack bots which automate repetitive tasks. This helps people save time and increase productivity. For example, there are new items bought from your eCommerce site or there is a bug reported then it sends a short summary to a slack channel.

6. Personal Assistant:

People could use Bots as a fashion advisor for clothing recommendations, or ask trading tips from a finance bot, suggest places to visit from a travel bot and so forth. This would help the users get a more personal touch from the chatbot. Also, the chatbot will remember all your choices and provide you with relevant choices the next time you visit it.

How chatbot can drive revenue for you?

Below we have compiled reasons why chatbots are important for your business and how can they help in increasing revenues:

a. Higher user customer engagement

Most businesses these days have a web presence. But with being on the internet, boundaries of day and night, availability and unavailability have changed, so have user expectations. This is probably the biggest reason to use them. Bots give the user an interactive experience. It makes customers feel they are working with someone to help resolve their issue. If done right, bots can help customers find what they are looking for and make them more likely to return.

Customer Engagement

• **Clearance Sale :** Notify users about on-going clearance sale of products relevant to the users at their nearest outlets.

• **Product Finder :** Enable consultative selling without the need of a call center

• **It offer Notification :** Notify users about offers, product launches on products/ services they've shown interest in, and products that's back in stock

b. Mobile-ready and immediate availability

Along with a web presence, it has also become increasingly important for brands to have a mobile presence - mobile apps, mobile-optimized websites. Considering how chat has been around on the mobile for ages, most chatbot implementations don't need you to work on tweaking their UI, they are ready to implement and so available to your customers immediately

You might argue that you have an app for that. Having an app for your brand is great, but having users discover that app, download it and use it to stay engaged is not an easy deal. Instead, implementing a chatbot - which works on the mobile browser or a messaging-app which the user regularly uses - makes it all the more reason for a customer to be engaged with the brand

c. It can drive sales

Chatbots can be intelligent. Depending on a user's preferences or purchases, it can send products to customers which are more likely to convert into sales. Or it can send coupons to users for in-store purchases/discounts. Bots can also be used to link the user to your mCommerce site/app so they can buy the product directly from the convenience of their phones

Sell Intelligently

• **Product Recommendations :** Push proactive recommendations to users based on their preferences and search and order history.

• Enable order booking over chat.

d. Minimal cost - Maximum return

The best part about bots is they are cheap. Chatbot provide the necessary infrastructure and APIs for creating these bots. They require minimal maintenance and since it is automated, there is no labor-intensive work that goes in there.

e. Customer Service

• **Track Order :** Keep users up to date with order status. Schedule or reschedule delivery to a provided address or request to pick it up at any other Best Buy outlet.

• **Stock outs :** Notify users when desired product is available and place order over a chat.

• **Returns and Replacements :** No waiting time to reach customer care. Customers can instantly place request to replace or return an order.

• **Seek Reviews :** Reach out to users to seek reviews on the products recently bought

Application across Industries

According to a new survey, 80% of businesses want to integrate chatbots in their business model by 2020. So which industries can reap the greatest benefits by implementing consumer-facing chatbots? According to a chatbot, these major areas of direct-to-consumer engagement are prime:

Chatbots in Restaurant and Retail Industries

Famous restaurant chains like Burger King and Taco bell has introduced their Chatbots to stand out of competitors of the Industry as well as treat their customers quickly. Customers of these restaurants are greeted by the resident Chatbots, and are offered the menu options- like a counter order, the Buyer chooses their pickup location, pays, and gets told when they can head over to grab their food. Chatbots also works to accept table reservations, take special requests and go take the extra step to make the evening special for your guests.

Chatbots are not only good for the restaurant staff in reducing work and pain but can provide a better user experience for the customers.

Chatbots in Hospitality and Travel

For hoteliers, automation has been held up as a solution for all difficulties related to productivity issues, labour costs, a way to ensure consistently, streamlined production processes across the system. Accurate and immediate delivery of information to customers is a major factor in running a successful online Business, especially in the price sensitive and competitive Travel and Hospitality industry. Chatbots particularly have gotten a lot of attention from the hospitality industry in recent months.

Chatbots can help hotels in a number of areas, including time management, guest services and cost reduction. They can assist guests with elementary questions and requests.

Thus, freeing up hotel staff to devote more of their time and attention to time-sensitive, critical, and complicated tasks. They are often more cost effective and faster than their human counterparts. They can be programmed to speak to guests in different languages, making it easier for the guests to speak in their local language to communicate.

Chatbots in Health Industry

Chatbots are a much better fit for patient engagement than Standalone apps. Through these Health-Bots, users can ask health related questions and receive immediate responses. These responses are either original or based on responses to similar questions in the database. The impersonal nature of a bot could act as a benefit in certain situations, where an actual Doctor is not needed.

Chatbots ease the access to healthcare and industry has favourable chances to serve their customers with personalised health tips. It can be a good example of the success of Chatbots and Service Industry combo.

Chatbots in E-Commerce

Mobile messengers- connected with Chatbots and the E-commerce business can open a new channel for selling the products online. E-commerce Shopping destination "Spring" was the early adopter. E-commerce future is where brands have their own Chatbots which can interact with their customers through their apps.

Chatbots in Fashion Industry

Chatbots, AI and Machine Learning pave a new domain of possibilities in the Fashion industry, from Data Analytics to Personal Chatbot Stylists. Fashion is such an industry where luxury goods can only be bought in a few physical boutiques and one to one customer service is essential. The Internet changed this dramatically, by giving the customers a seamless but a very impersonal experience of shopping. This particular problem can be solved by Chatbots. Customers can be treated personally with bots, which can exchange messages, give required suggestions and information. Famous fashion brands like Burberry, Tommy Hilfiger have recently launched Chatbots for the London and New York Fashion Week respectively. Sephora a famous cosmetics brand and H&M– a fashion clothing brand have also launched their Chatbots.

Chatbots in Finance

Chatbots have already stepped in Finance Industry. Chatbots can be programmed to assists the customers as Financial Advisor, Expense Saving Bot, Banking Bots, Tax bots, etc. Banks and Fintech have ample opportunities in developing bots for reducing their costs as well as human errors. Chatbots can work for customer's convenience, managing multiple

accounts, directly checking their bank balance and expenses on particular things. Further about Finance and Chatbots have been discussed in our earlier blog: Chatbots as your Personal Finance Assistant.

Chatbots in Fitness Industry

Chat based health and fitness companies using Chatbot, to help their customers get personalised health and fitness tips. Tech based fitness companies can have a huge opportunity by developing their own Chatbots offering huge customer base with personalised services. Engage with your fans like never before with news, highlights, game-day info, roster and more.

Chatbots and Service Industry together have a wide range of opportunities and small to big all size of companies using chatbots to reduce their work and help their customers better.

Chatbots in Media

Big publisher or small agency, our suite of tools can help your audience chatbot experience rich and frictionless. Famous News and Media companies like The Wall Street Journal, CNN, Fox news, etc have launched their bots to help you receive the latest news on the go.

Conclusion

In this way we implemented an elementary catboat for any suitable customer interaction application.

Assignment No-06(Group C)

Title: - Implement Expert System.

Objectives:-

1. Understand the concept of Expert System

Problem Statement:-

Implement any one of the following Expert System

I. Information management
II. Hospitals and medical facilities
III. Help desks management
IV. Employee performance evaluation
V. Stock market trading
VI. Airline scheduling and cargo schedules

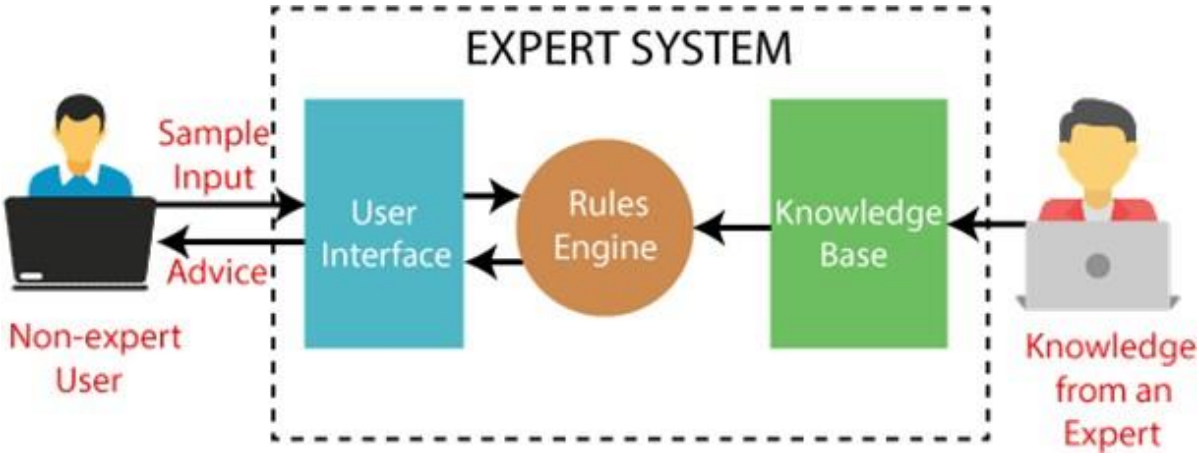
Software and Hardware requirements:-

2. **Operating system:** Linux- Ubuntu 16.04 to 17.10, or Windows 7 to 10,
3. **RAM-** 2GB RAM (4GB preferable)
4. You have to install **Python3** or higher version or Turbo C++ or JDK

Theory-

What is Expert System?

Expert System is an interactive and reliable computer-based decision-making system which uses both facts and heuristics to solve complex decision-making problems. It is considered at the highest level of human intelligence and expertise. The purpose of an expert system is to solve the most complex issues in a specific domain.



Expert Systems in Artificial Intelligence

The Expert System in AI can resolve many issues which generally would require a human expert. It is based on knowledge acquired from an expert. Artificial Intelligence and Expert Systems are capable of expressing and reasoning about some domain of knowledge. Expert systems were the predecessor of the current day artificial intelligence, deep learning and machine learning systems.

Examples of Expert Systems

Following are the Expert System Examples:

MYCIN: It was based on backward chaining and could identify various bacteria that could cause acute infections. It could also recommend drugs based on the patient's weight. It is one of the best Expert System Example.

DENDRAL: Expert system used for chemical analysis to predict molecular structure.

PXDES: An Example of Expert System used to predict the degree and type of lung cancer

CaDet: One of the best Expert System Example that can identify cancer at early stages

Characteristics of Expert System

The Highest Level of Expertise: The Expert system in AI offers the highest level of expertise. It provides efficiency, accuracy and imaginative problem-solving.

Right on Time Reaction: An Expert System in Artificial Intelligence interacts in a very reasonable period of time with the user. The total time must be less than the time taken by an expert to get the most accurate solution for the same problem.

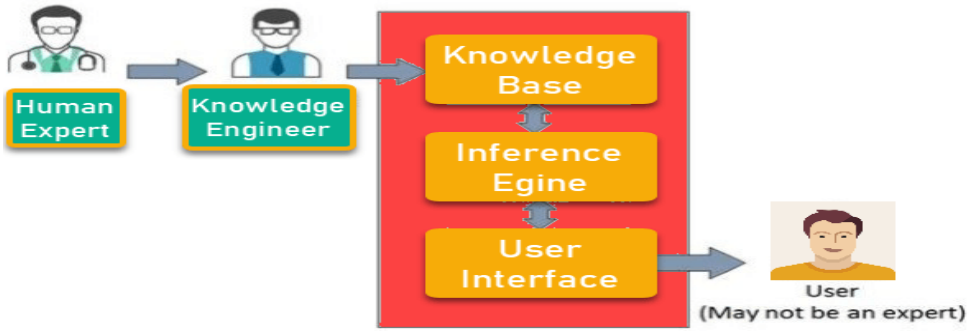
Good Reliability: The Expert system in AI needs to be reliable, and it must not make any a mistake.

Flexible: It is vital that it remains flexible as it the is possessed by an Expert system.

Effective Mechanism: Expert System in Artificial Intelligence must have an efficient mechanism to administer the compilation of the existing knowledge in it.

Capable of handling challenging decision & problems: An expert system is capable of handling challenging decision problems and delivering solutions.

Components of Expert System



The Expert System in AI consists of the following given components:

User Interface

The user interface is the most crucial part of the Expert System Software. This component takes the user's query in a readable form and passes it to the inference engine. After that, it displays the results to the user. In other words, it's an interface that helps the user communicate with the expert system.

Inference Engine

The inference engine is the brain of the expert system. Inference engine contains rules to solve a specific problem. It refers the knowledge from the Knowledge Base. It selects facts and rules to apply when trying to answer the user's query. It provides reasoning about the information in the knowledge base. It also helps in deducting the problem to find the solution. This component is also helpful for formulating conclusions.

Knowledge Base

The knowledge base is a repository of facts. It stores all the knowledge about the problem domain. It is like a large container of knowledge which is obtained from different experts of a specific field.

Thus we can say that the success of the Expert System Software mainly depends on the highly accurate and precise knowledge.

Other Key terms used in Expert Systems

Facts and Rules

A fact is a small portion of important information. Facts on their own are of very limited use. The rules are essential to select and apply facts to a user problem.

Knowledge Acquisition

The term knowledge acquisition means how to get required domain knowledge by the expert system. The entire process starts by extracting knowledge from a human expert, converting the acquired knowledge into rules and injecting the developed rules into the knowledge base.

Advantages of Expert System

1. These systems are highly reproducible.
2. They can be used for risky places where the human presence is not safe.
3. Error possibilities are less if the KB contains correct knowledge.
4. The performance of these systems remains steady as it is not affected by emotions, tension, or fatigue.
5. They provide a very high speed to respond to a particular query.

Limitations of Expert System

1. The response of the expert system may get wrong if the knowledge base contains the wrong information.
2. Like a human being, it cannot produce a creative output for different scenarios.
3. Its maintenance and development costs are very high.
4. Knowledge acquisition for designing is much difficult.
5. For each domain, we require a specific ES, which is one of the big limitations.
6. It cannot learn from itself and hence requires manual updates.

Applications of Expert System

1. **In designing and manufacturing domain**

   It can be broadly used for designing and manufacturing physical devices such as camera lenses and automobiles.

2. **In the knowledge domain**

   These systems are primarily used for publishing the relevant knowledge to the users. The two popular ES used for this domain is an advisor and a tax advisor.

3. **In the finance domain**

   In the finance industries, it is used to detect any type of possible fraud, suspicious activity, and advise bankers that if they should provide loans for business or not.

4. **In the diagnosis and troubleshooting of devices**

   In medical diagnosis, the ES system is used, and it was the first area where these systems were used.

5. **Planning and Scheduling**

   The expert systems can also be used for planning and scheduling some particular tasks for achieving the goal of that task.