**Quiz 03**

**Artificial Intelligence-AI3151**

**Instructor: Aqsa Kiran**

**Dated: 01-01-2021**

**Maximum Time: 40 Minutes**

****

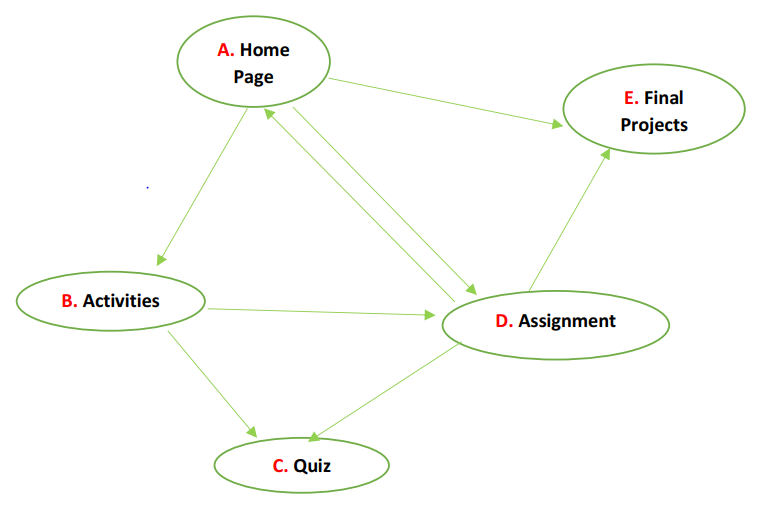
**Note:**

1. **All questions are mandatory.**
2. **Write short and descriptive answers or specific syntax where required.**
3. **Submit .docx file (word file) solution with subject AI\_y1\_Quiz\_03.**

**Q1: In what cases, we apply uninformed search and informed techniques? Give three example searching problems? 3M**

**Q2: What are the criteria to evaluate the search techniques? 2M**

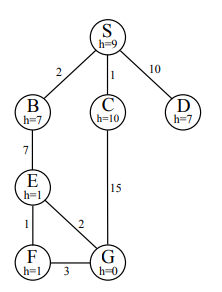
**Q3: A final year student is looking for the project material online, so following is the searching graph of an E-Learning website. Apply DFS and UCS (assume different path cost at each step) technique such as: 5M**



1. Write its algorithm and which data structure it follows?
2. Find out the goal path sequence by traversing depth wise, write its adjacency list starting from A and goal is E.
3. Evaluate its performance based on the criteria you defined in Q2.

**Q4: Write short and descriptive answers to the following:**

1. Difference between GBFS and Recursive GBFS? 2M
2. Consider the search graph shown below. S is the start state and G is the goal state. All edges are bidirectional. Give the path that would be returned using A\* search strategy by implementing proper Open and Closed queue. 3M



1. How would you find the Admissibility and Consistency of any state let’s say B or any other state to defend the optimality of A\* technique used?

aqsakiran60@gmail.com

**ANSWERS**

**Answer 1:**

In Uninformed searches we do not know about our goal state, we don’t have any information about anything, In un-informed searches the only information program have is only described in problem definition.

**For Example:**

1. Searching largest number in a huge data
2. Searching for a specific sequence in DNA’s
3. Finding largest amount of energy in chemical reactions

Whereas in informed searches we have our goal state given, we know what exactly we are searching, in informed searches our program ends as we reach the goal state

**For example:**

1. Searching the best possible route to a destination
2. Searching the shortest path for a destination
3. Solving a puzzle unless it is complete

**ANSWER 2:**

Criteria to evaluate the searching techniques depends on the problem itself, if we have a problem with no goal then we use uninformed search because that will be the best way to tackle that problem

Where as if we have a defined goal with its path cost at each step then we UCS or A\* or GBF because they all make their decisions depending on the cost and then some are still time consuming so it all depends on problem some of very common criterias are:

1. Cost checking criteria
2. Heuristic cost checking
3. Final cost (HC + C)
4. Time checking criteria
5. Less storage usage

**ANSWER 3**

|  |  |
| --- | --- |
| **OPEN LIST (DFS)** | **CLOSE LIST (DFS)** |
|  | **A** |
| **B,D,E** | **B** |
| **C,D,E** | **C** |
| **D,E** | **B** |
| **D** | **D** |
| **E,C** | **E (Final state )** |
|  |  |

**UCS: (**values in box are cost**)**

4

3

1

2

1

5

1

|  |  |
| --- | --- |
| **OPEN LIST** | **CLSOED LIST** |
|  | **A** |
| D[1],B[3],E[4] | D[1] |
| E[1],B[3],E[4] | E[4] FINAL GOAL REACHED |
|  |  |
|  |  |

**PART A:**  It follows Tree data Structure

Algorithm for Uninformed search :

* Create nodes from which we need to search
* Initialize 2 lists[open and closed]
* Pick 1 node and push it into closed list
* Discover all nodes of element of closed list and put them in open list
* Then push first element of open list into closed list
* Repeat this until goal is reached

**Part C (Uniform cost search)**

It is pretty much optimal because we know our cost at each step

We can find the best possible way by analyzing the nodes with least cost

And hence we reached to our GOAL with less time and less memory consumed because we only discovered 2 nodes and we reached to our goal

**ANSWER 4**

**Difference between GBFS and Recursive GBFS**

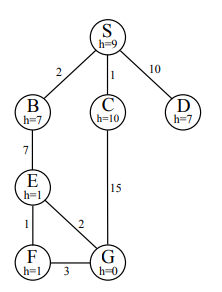
GBFS is Greedy Breadth first search and it only seeks the nodes with the least cost

It can stuck into a loop if it keeps finding least cost on a same point

This makes it more time consuming

RECURSIVE GBFS is a BFS but it recurse itself at each node and it doesn’t stuck into a loop

It finds least cost by recursing itself at each node it reaches.



|  |  |  |  |
| --- | --- | --- | --- |
| open | Close |  |  |
| S[9] |  |  |  |
| B[9],C[11],D[17] | S[9] |  |  |
| E[8] ,C[11],D[17] | B[9] |  |  |
| G[2],F[2] ,C[11],D[17] | E[8] |  |  |
|  | G[2] GOAL |  |  |
|  |  |  |  |
|  |  |  |  |