***NAME: MAHMOUD NAGUIB RASHAD ELSAYED***

***LEVEL: TWO***

***CODE: 2016262***

***(DATA STRUCTURE)***

***(REPORT ABOUT APPLYING DATA STRUCTURES IN PROBLEMS SOLVING)***

***-*** ***DATA STRUCTURES and Algorithms are the steps to solve a problem.***

***(Sort Algorithms)***

**-The algorithm of ranking is one of the most studied algorithms and research in the field of computer science, the goal of which is to arrange several elements in a particular order.**

**-Some of the most popular ranking algorithms are:**

* **Merge Sort**
* **Quick Sort**
* **Heap Sort**
* **Bubble Sort**

**-Example: Sort by price or popularity in commercial websites.**

***(Search Algorithms)***

**-This algorithm searches for a particular element in a place and the most famous example is what is known as binary research.**

**-It divides the elements in half a number of times until the desired element is found.**

**-Its applications:**

* **When you search for a file name, for example, a video clip, it uses the previous method until it is found.**

***(Depth / breadth of first search)***

**-Used in research, but in the data structure which is a useful and very important function.**

**-The most important applications:**

1. **Search for information in search crawling.**
2. **In Artificial intelligence in building robots like a robot who plays chess.**
3. **Find the shortest way between two points and solve the labyrinths (Mazes).**

***(Hashing)***

**-** **Currently one of the most commonly used algorithms is to find an element via its key / value pair.**

**Also known as the dictionary and binds all its values you want to save with a particular key so search for its values.**

**-All you need is to use the key and it is much faster than Sort + Search to find an item.**

**-The most important applications:**

* **In the router to register IPs.**
* **To make sure that there is a specific value in the List where the linear search will take a large time to match the number of elements.**

***(Dynamic Programming)***

**-This algorithm is used to solve complex problems by cracking them into smaller parts, the small problems bees remember, solve it well then solve the bigger problems quickly.**

**-Example:**

**what is the sum of 1 + 1 + 1 + 1 + 1, will prepare it and say 5, but if I told you add one, what is the number will respond immediately 6 Without counting again.**

**How did you do it?**

**-You just added one to the number you remember in the previous problem without counting the time again.**

**(Dynamic Programming does the same to solve large problems).**

***(Exponentiation by squaring)***

**-** **If you want to calculate all you will do is do a loop for 32 times to find a solution, what if I told you that you are getting The same answer is done with a loop for only 5 times. There is no problem with this solution.**

**-** **This algorithm is used to find a solution for a number raised to large forces in a very small time.**

***(String Matching and Parsing)***

**-** **One of the most important algorithms used to search for a certain section in a huge text, when you use the ctrl + F, you search for a huge text**

**In the file using this algorithm, the most we can talk about that this algorithm also uses Hashing algorithm to speed up the search process.**

***(Stack)***

**-** **It is a waiting line for a set of data, which distinguishes this line as open from one direction only That is data Enters and exits from a single gateway. A stack is called a LIFO and this symbol means that the last element Entering the stack is the first element that restricts it, and this is obvious since we do not have a single slot to insert and remove items.**

**-Stack: The last disc you place is the first disc that you take out, or as a group.**

**-** **What jobs do we need in the stack?**

**1. Push: This process is used to add items to the stack.**

Public void push (int i) {

Stack List . Add (i);

}.

**2.** **Pop: used to output the first item (last added element) from the stack.**

public Integer pop(){

If ( ! Is empty ( ))

{

return stackList.removeLast();

}else

throw new java.util.EmptyStackException();

}.

1. **Is Empty: used to check if the stack is empty or not.**

Public Boolean is Empty () {

Return stackList.is Empty ();

}.

1. **Clear: Used to delete all items in the stack.**

public void clear(){

Stack list.clear ();

}.

1. **Top El: Used to query the first element (last added element) in the stack.**

***(Queues)***

**-** **Queues Is the waiting queue of this queue, which increases in size by adding data at the end and reduces the output of data from the front.**

**-The general meaning is similar to regular queues where the first comes first and the last one stands at the end of the row. If the reverse is stuck.**

**-The Queues has an end and an introduction, the end to enter the data and the intro to exit.**

**-** **What do we need to create Queues?**

* **Add a new item to the queue.**

Public void enqueue(int a) {

queue.add(a);

}.

* **Delete the first element in the queue.**

Public int dequeue(){

return queue.removeFirst();

}.

* **Query the beginning of the queue.**

Public int firstEl(){

return queue.getFirst();

}.

* **Clear: Filter Queues (Delete All Items).**

public void clear(){

queue.clear();

}.

* **Is Empty: Check if the queue is free.**

public boolean isEmpty(){

return queue.isEmpty();

}.

***(Linked List)***

**-Is a type of data structures consisting of a group of cells interconnected between them and each element in it called node and this node where the two fields, the first field is used to record values, the second field is a pointer indicates the title of the next node or the previous or NULL in case this episode is the last link.**

struct node{

int num;

char name[10];

node \* next;

};

typedef node \*node\_ptr; .

**-The most important kinds:**

* **Singly Linked List**
* **Doubly Linked List**

**-It has four ways:**

**1. Add the first ring.**

**2. Add from right (end)**

**3. Add from left (Start)**

**4. Addition of the center.**

***(Array)***

**-It is a way to store data of one type, of a specific size that is specified from the beginning of the matrix configuration.**

**-Types of array:**

**- One-dimensional array.**

int a[] = {1 , 2, 3, 4, 5};

char b[] = {'a', 'b'};

**1.Find a specific element within the array:**

// Array Search Algorithm

// input: array, size of array, count, key

LOOP I = first element index UNTIL last element index

IF current element = key:

RETURN current index

RETURN -1

**2.Deleting an item using its value for both ordered and subordinated array:**

// Array Delete Algorithm

// input: array, size of array, count, key

1. IF count = 0 :

2. PRINT “ Array is empty” AND Exit

3. ELSE continue to step 2.

4. i = Search( array, count, size, key)

5. IF i = -1 :

6. PRINT “ Not found “ AND Exit

7. ELSE // shift in

8. LOOP j = i UNTIL before last element index:

9. A[ j ] = A[ j+1 ]

10. Count - -

11. RETURN count

**-** **Passed by reference:** **When a parameter is passed by reference, the caller and the callee use the same variable for the parameter. If the callee modifies the parameter variable, the effect is visible to the caller's variable.**

**-** **Passed by value:** **When a parameter is passed by value, the caller and callee have two independent variables with the same value. If the callee modifies the parameter variable, the effect is not visible to the caller.**

***(Binary Trees)***

**-** **Is a type of data representation and consists of two main components, Nodes and Arcs.**

**-When we represent this tree, we change its concept and the root is in the top and bottom of the bottom.**

**-It is called the bilateral trees because each contract has two pillars connected to them and they are his sons and he has one father.**

**-Node:** **Each element is in the tree, and is the place where the data is stored.**

**-** **Special types of (Nodes):**

**- Root: An element that does not have any parent that is the highest element in the tree. Each tree contains only one root.**

**-Leaves: This is the group of the contract that does not have any children. Ie they are at the end of the tree.**

**-** **Arc: A link that connects the nodes between them and is one-way from top to bottom.**

**(Root)**

**(Arc)**

**(Leaves)**

**(I Hope You Liked The Report)**