Overview of data structures and algorithms computer science essay

Technology, Computer

As Robert Lafore has stated, data structures are a collection of structures that are used to store data in a memory location. A Data structure is designed to organize data to fulfill ones purpose of accessing and using them. Arrays, linked lists, stacks, binary trees, and hash tables etc…These are included as data structures. But there are data structures that can be thought of as Abstract data types as well, the given examples above can be considered as ADTs except arrays. All most all data structures have both advantages and disadvantages.

Nell B. Dale et al (1996) has stated that an abstract data type is one whose set of operations that are defined in a certain level without getting any restrictions from operational details.

According to Robert Lafore (2008) An Abstract data type is a way of looking at a data structure simply focusing on what it does instead of focusing on how it does it.

Both of these definitions give a similar meaning. If taken from Nell Dale’s point of view it’s a set of operations in a certain level that has no restrictions by operational details on it. When compared this definition with Robert Lafore’s definition, An Abstract data type is a way of looking at the operations of the data structure and as Nell Dale stated in the end “ without getting any restrictions by operational details”. By operational details Nell Dale has meant it does not focus on how on the details of those operations.

In conclusion to both these definitions, an abstract data type is a logical operation on certain level of data structures focusing on its process and not its method. Stacks and queues are examples of ADTs.

. Classification of data structures

Arrays

Robert lafore has explained Arrays to be known as the most commonly used data structure in programming. An array is a series of elements that is of the same type and same name. It is possible to create an array of any data type. In order to access an array the array has to have an index. An index is an integer that allows access to each of the elements of an array. Using an array we can name a group of elements instead of naming each of them individually.

A simple array will look like this see FIGURE 1. 2. 1

Data

Index

FIGURE 1. 2. 1 a simple array

The data within the array are the data entered to an array. And the numbers below outside the array are indexes used to access the particular data packet.

Advantages

Quick insertion

Very fast access if index is known

Disadvantages

Slow search

Slow deletion

Fixed size

Stacks

A stack is special a kind of a list in which both insertion and deletion takes place from the top of the stack. To add new elements to a stack a word called “ push” is used instead of insert and for removal or deleting of an element the word “ pop” is used.

Robert lafore(2008) has stated that a stack allows access to only one data item at a time. This is because a stack provides a “ LIFO” or Last in first out method. This means the last data item that is pushed into a stack is the first item that will be popped. Then it provides access to the next data item and so on.

By the name of this data structure itself it gives a picture of how the data structure would be. Given an example, a stack of data blocks, the last data block to be entered gets removed first. See FIGURE 1. 2. 2

Insertion of new data item takes place on top

Deletion of item takes place on top

Stack of data blocks

30

30

1

1

1

24

24

24

5

5

5

14

14

14

16

16

16

FIGURE 1. 2. 2 a stack of data blocks

Advantages

Provides last in first out access

Disadvantages

Slow access to other items

Queues

This is a data structure of ordered elements in which insertion takes place at the end and deletion takes place at the front. A queue is a known as a “ FIFO” or first in first out. Unlike stacks in queues the first data item to be entered is removed from the queue first. The process of a queue is easy to understand when modeled to the real world. For example think of the line ups in your local grocery stores. Each of the customer’s waits in line and one at the front of the line get to go first. Each of the new customers joins the line in the rear. See figure 1. 2. 3

New customers join at the end of the queue

First customer joins the queue, first to go

FIGURE 1. 2. 3 A queue of customers

Similar to the figure 1. 3 above in computer science the elements which are inserted first in a queue are removed first from the queue.

Advantages

Provides first in first out access

Disadvantages

Slow access to other items

Linked lists

Linked list has a similarity with arrays. D. Samanta (2004) has defined that linked lists as a ordered collection of finite, adjacent data elements called as nodes where linear order is maintained by links or pointers. Links or pointers are used to address to the other node. Simplify by pointing to the next location in the sequence. The pointers are maintained depending on the requirements. FIGURE 1. 2. 4 shows a simple example of a linked list.

Link

Link to the next node node

Data

FIGURE 1. 2. 4 a simple linked list

Advantages

Quick insertion

Quick deletion

Disadvantage

Slow search

1. 3. Importance of data structures in software industries

Importance of data structures

To understand the importance of data structures first we have to know what data structures are. It is already explained in the previous sections.

What is the importance of it? The importance is that a data structure can give an overview of the types of operations that can be performed and its processes.

In a software industry

Software industries keep changing and growing continuously, which requires a great deal of innovative ways to solve the problems they face. Use of data structures can bring down the stress level down to a point. Because uses of data structures are most effective if the programmers know what he wants to do with the data, whether he wants random access or the ability to move back and forth through the data. Use of data structures helps the programmer to determine which data structure has to be used in which situations. It is very important that the data are stored in a way that it can be retrieved and accessed easily in the future. Data structures fulfill this importance.

According to Robert Lafore (2008) data structure is nominated among three categories where it is useful.

Real- world data storage

Programmer’s tools

Real-world Modeling

These three categories help understand the usefulness of data structures.

By real-world data storage it means data is described based on physical entities like personnel records that describe a human, here the data is personnel records and entity is human.

Programmer’s tools are meant to be accessed only by the program itself. Because this is entirely based on the data storage structures that are not meant to be accessed by the user. Stacks, queues, and priority queues are tools that a programmer uses in such a way to facilitate some other operations.

Last but not the least its real-world modeling. It is basically structures that model real world situations. The most important data structure of this type is graph. Graphs can be used to represent airlines routes between cities etcA? a,¬A¦

Based on these three categories alone, we can point out that data structures provide a vast area in which data structures can be helpful for development of software industries.

1. 4. Complexity in relevant to data structures

When using data structures there is an important aspect related to them called as algorithms. Robert Lafore (2008) has defined that algorithms are used to manipulate the data within the data structures. Algorithms can be used in various ways, for searching particular data items and sorting the data.

Shi kuo chang has stated that the concept of an algorithm is one of the most important elements of computer science. Because it’s useful to frame a problem and get solutions without any errors.

When it comes to algorithms there are two complexities that arise from it, Time complexity and space complexity.

Time complexity

As Alfred V Aho et al (2009) has explained the time needed by an algorithm given as a function of the problems size is called as the time complexity.

There are three cases in time complexity best case, average case and worst case this can be explained separately in relevance to data structures.

Best case: – This is when an algorithm takes minimum amount of time to search a desired set of inputs.

For an example you can take a stack when searching for an element in a stack with a Linear search if u come across the element you are looking for at the top of the stack then it can be called as best time complexity.

Average case: – Average time complexity is the mean number of operations assuming the probability of the input.

The binary search method in algorithm can be identified as an average case time complexity method.

Worst case: – If an algorithm takes maximum amount of time to find the desired set of inputs then it’s called a worst case time complexity.

For an example you can take a stack again when searching for a desired element using a linear search if it is at the bottom of the stack then it is a worst case time complexity.

Space complexity

According to Alfred V Aho et al (2009) space complexity determines the size of the problems that can be solved by an algorithm. This simply means this is a function that describes the amount of memory space taken by the algorithm.

1. 5 Influence of object oriented programming concepts on data structures

According to Matt A. Weisfeld,(2004) Object oriented programming has been around in software development from 1960’s. The fact that Object oriented programming helps encapsulate the data and the operations that manipulate them in the object is considered as an advantage that arises from object oriented programming.

Tim Patrick et al has explained that when it comes to object oriented programming, there are four main concepts we need to be aware of, as follows.

Abstraction

Encapsulation

Inheritance

Polymorphism

.

Abstraction:-

This is a view of an entity that includes only those aspects that are relevant to a situation.

Encapsulation:-

This is the process of converting abstraction into a usable software component. A simple explanation is restricting access to the attributes and methods of an object directly.

Encapsulation helps encapsulate a process of a data structure. Example can be an array because of the encapsulation an output of an array would be the elements within the array it won’t show how it was processed.

Inheritance:-

This makes it possible for OOP code to build classes that extend or restrict features in other existing classes. Without the need to fully re write the code. Simply speaking, inheritance can receive attributes and methods from super classes.

When you take data structures there are special kinds of data structures identified as abstract data types. The inheritance concept can create new abstract data types from old. An example can be a stack. Use of inheritance allows a user to create a new stack from an old one.

Polymorphism:-

A simple explanation of polymorphism is that a recognised action can be performed differently in different situations by an object.

The influence of polymorphism concept in data structures is that it creates operations that can be applicable for data structures which store more than one type of data.

An example for this can be linked lists. A linked list can be implemented with operations of stacks and queues using polymorphism concept.

These four concepts are included among the other principles of object oriented programming. The last three concepts from the four alone give an idea, how oriented programming object can influence data structures.

Task 2

K

2. 2 A stack and its procedure for various operations.

According to V. rajaraman et al (0000) a stack is a memory location where the data are stored and retrieved in a location called top of the stack. When the data is entered the firstly entered data are pushed there for the first data to be entered will be retrieved last. Unlike a RAM where each of the data can be addressed separately in a stack the only data that can be addressed is the top of the stack.

There are two main Operations associated with a stack, PUSH and POP.