EAZY ROOMS

NAME : NALLAVENGANNAGARI JAGADEESWAR REDDY

USN : 1NH18CS124

SECTION : 3B

REVIEWER : MR.GAGAN

CHAPTER -1 :

INTRODUCTION

1.1 PROBLEM DEFINITION

The word itself says that allocating vacant rooms to the customers. By entering the details of the customer in a book is comparatively difficult, so a software is created to implement this .This is coded by using data structures .In data structures linked lists , arrays are implemented . This is used to provide information on booking rooms .

This program will reduce the effort of the people who are visiting new places . They can easily book their room online and can save time from searching rooms to stay. The program contains operations like food menu, hotel bill, customer details. In this project I used single linked lists to show the customer details, food and hotel bill.

1.2 OBJECTIVES:

In this project I am implementing program on allotting rooms to the customers on their comfort and providing food directly to their room. I am using three linked lists in this project.

1. Adding a new customer
2. Food menu
3. Receiving food order
4. Total bill along with food

   The first ten rooms in the hotel are AC rooms and the remaining ten rooms are NON-AC rooms . The customer details consists of name of customers and phone number. The food list consists of two items and they can order any of them. By using switch case I am implementing adding customer ,removing customer , food order and displaying the details.

1.3 METHODOLOGY:

In this project I am using data structures as main concept to determine the vacancy of rooms.

By using food menu I am adding food items and by using add customer details I am adding customer details to the program and then by using room list I am allotting rooms to the customer and by using remove customer I am removing customer details when he is vacating the room. By using food order I am delivering the food to the ordered room.

By using switch case I am displaying adding customer , removing customer , ordered food and total bill of the customer.

1.4 EXPECTED OUT COMES

1. Add customer
2. Remove customer
3. Display customer details
4. Receive food order

1.5 HARDWARE REQUIRMENTS

Processor	: Any Processor above 500 MHz
RAM	512
Hard Disk	: 10 GB
Input device	: Standard Keyboard and Mouse
Output device	: VGA and High Resolution Monitor

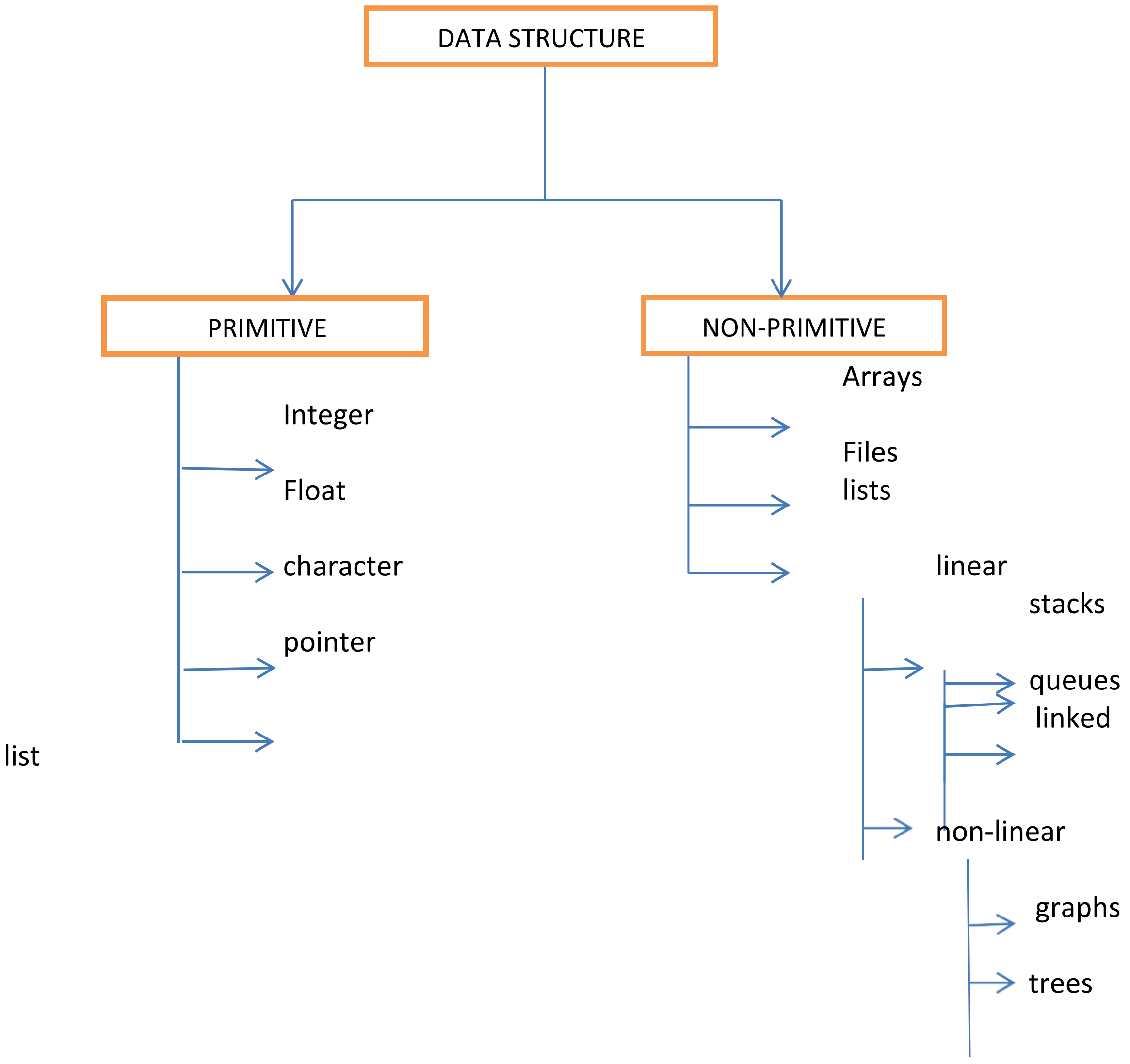
1.6 SOFTWARE REQUIREMENTS

• Operating system	: Windows XP
• Front End	: ASP.Net 2.0
• Server	: Internet Information Services

CHAPTER – 2

DATA STRUCTURES

Data structure is used for storing data in a format.



Data structures are of two types :primitive data structures and non-primitive data structures.

Primitive data structures can be directly manipulated by machine instructions.

examples: 1.int

2.float

3.char
4.pointer

Non-primitive data structures can not be directly manipulated using machine instructions.

1.Arrays
2.lists
3.files

Data structures is a main important concept of any programming language. It consists of so many linear and non linear data structures .

Linear data structures consists of static memory allocation and dynamic memory allocation. And dynamic memory allocation consists of linked lists , queue and stacks. Trees can used for any searching operations and graphs can be used for implement a any analysis and any survey conduction it will be used

2.1 STACKS

Stack is a data structure which is used for storing data. It follows last in first order (LIFO) Or stacks is first in last out (FILO) list.

Memory is allocated to the nodes using dynamic memory allocation functions such as malloc() , calloc(), realloc() and free().

1.malloc() : This function is used to allocate a complete single block of memory of the specified size. A pointer is used to store the address returned my malloc.

Syntax –

datatype*ptr=(datatype*)malloc(size)

2.calloc() : It is function which allocates a specified size of memory in multiple blocks of same size. Each block should be assigned to null. A pointer is used to store the address.

Syntax –

datatype*ptr=(datatype*)calloc(size, number of blocks)

40
30
20
10

3.realloc() : For reallocating the allocated memory this function is used. A pointer is used to store the address returned.

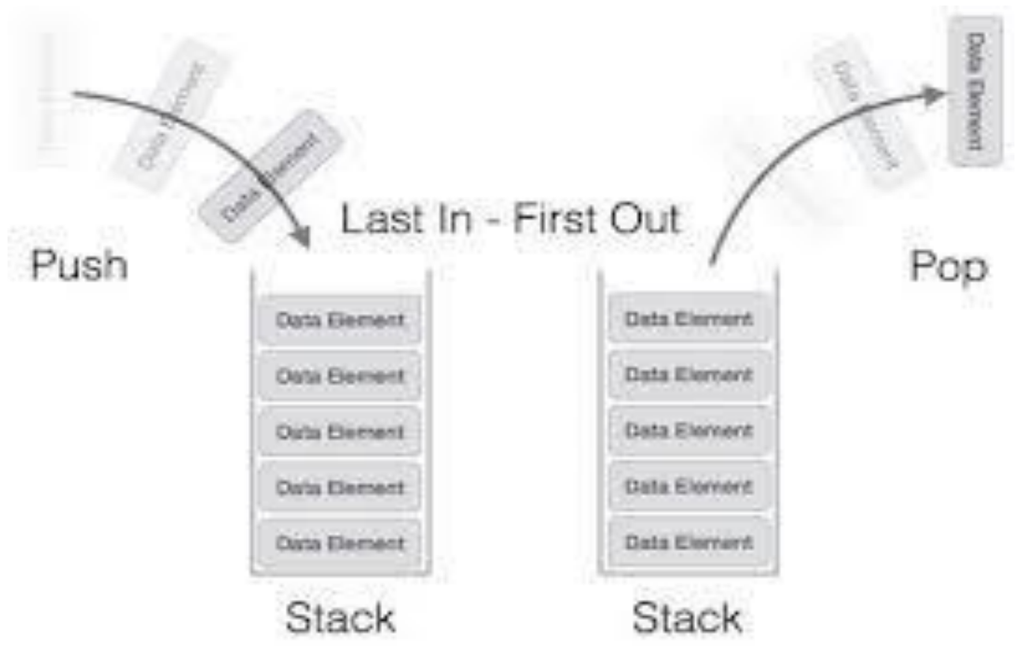
Syntax –

datatype*ptr=(datatype*)realloc(ptr,size)

4.free() : It is a function which is used to free the allocated memory.
Syntax –
free(pointer name)

When an element is inserted in a stack, the concept is called push and when an element is removed from the stack, these concept is called pop. Try to pop out an empty stack is called underflow. And trying to push an element into a full stack is known as overflow. Actually we will treat them as exceptions.

Exceptions are expected to be performed by an operation that cannot be evaluated. Operations like pop and top cannot be performed if the stack is empty.
Attempting the execution of pop on an empty stack throws an exception trying to push an element in a full stack throws an exception.



2.2 QUEUE :

deletion is done at beginning of the list. Generally, insertion is called as en-queue and deletion is called as de-queue. We will check whether the queue is empty or full by giving max-1. If this condition true we can say that queue is full.

Queues is a last in first out. So insertion is done at end of the list and the

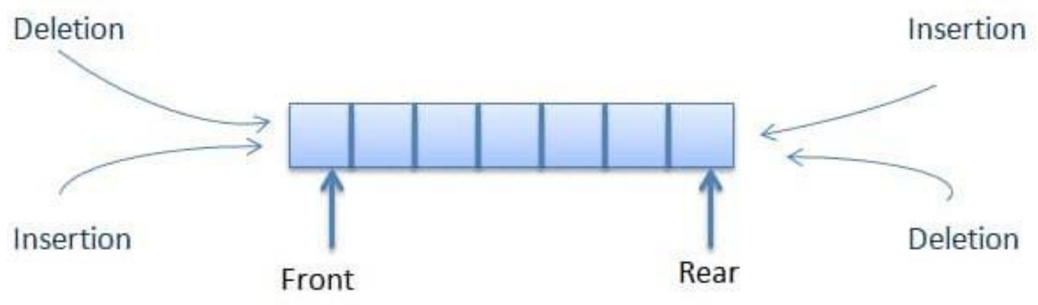
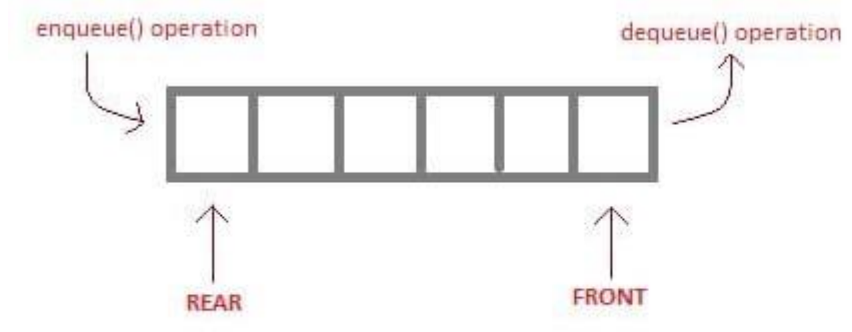


Fig-2.2:Working of a simple queue.



enqueue( ) is the operation for adding an element into Queue.
dequeue( ) is the operation for removing an element from Queue .

<u>QUEUE DATA STRUCTURE</u>

Working of enqueue and dequeue.
FIG-2.2

Insertion is happened at rare part and deletion is happening at front part. So we can say when queue is empty we will give the command as front=-1 and along that front is greater than rear.
There are many ways to implement queue operations and some of the commonly used methods are listed below.

- Simple circular array based implementation.
- Dynamic circular array based implementation.
- Linked list implementation.
- Multiprogramming.

The simple and the circular queue has same operations and the only difference is the circular queue has the elements will go under the circular manner.

And a queue is set up in a circular array with front and rear defined as usual. Assume that queues has some specific manner to interrupt the any other programming which will be coming from the suitable process.

Queues can be implemented in linear way that we have been used easily for any important purpose or any thing which will be usefull. If the queues is main important topic in the data structures which have been seen in this project.

In queues there are so many types:

- Circular queue.
- Linear queue.
- Double ended queue.
- Priority queue.

Disadvantages of stacks:

-> First element cannot be first accessed.
-> Insertion and deletion of elements can only be done at one end.
-> Stacks follow contiguous dynamic memory allocation.

Disadvantages of queue:

-> Insertion of elements can only be done at rear end.
-> Deletion of elements can only be done at front end.
-> Queue follow contiguous memory allocation.

As mentioned above these are the disadvantages of stacks and queue. All the disadvantages of stacks and queue are the advantages of linked list, hence linked list is brought into picture.

Syntax –
datatype*ptr=(datatype*)realloc(ptr,size)
4.free() : It is a function which is used to free the allocated memory.
Syntax –
free(pointer name)

2.3 LINKED LIST :

A linked list is a linear data structure , in which the elements are not sorted at contiguous memory locations ,a single linked list is linked list which has only one link .in a structure with different type of members in which atleast one node is pointing to itself is called self referential structure.

There are many other data structures that will do the same things as linked list. It is important to understand the difference between linked list and arrays. Both linked list and arrays are used to store the collection of data and since both are used for same purpose we can differentiate their usage.

That means in which cases arrays are suitable and in which cases linked list are suitable.

MAIN ADVANTAGES OF LINKEDLIST OVER ARRAYS IS:

1. Size of array is fixed, we must know its upper limit in advance. But in linked list size is not fixed.
2. Insertion and deletion is easy compared to array.
3. No memory wastage will be there in linked list.

1. single linked list

```
Struct node

{


  Int data; Structslist *ptr;


}
```

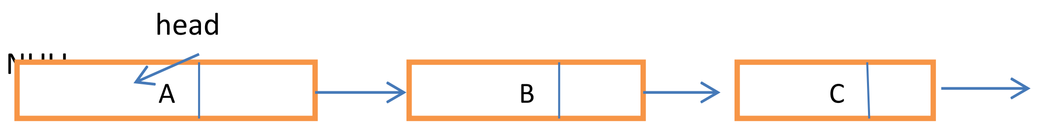
2. Double linked list:

```
  Structdlist
 {         int data;    structdlist *prev; structdlist *next;

}
```

Dynamic arrays is a variable size list data structures that enters the element to be added or removed in a single and simple way of inventing dynamic memory allocation arrays is to begining start with some fixed size of array.as soon as that array become full create the new array double the size of the original array similarly reduce the array size to half . If the elment in the array or less than half.

Structure of single linked list :



self referential code:

1.Single linked list

```
structslist
{
int data:
structslist *ptr;
}
```

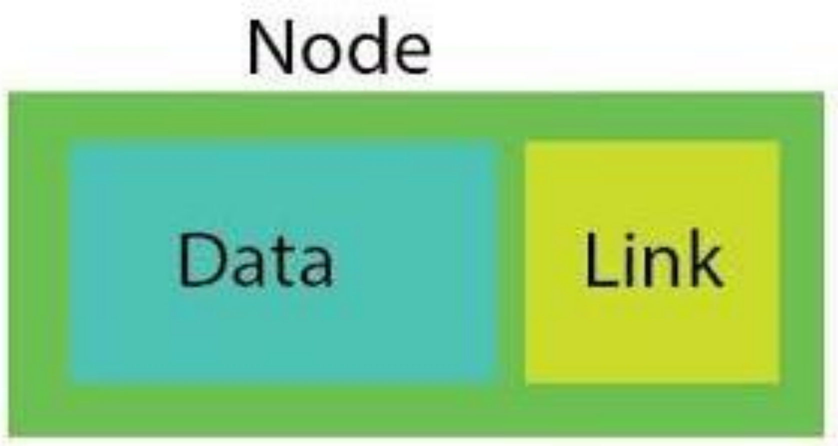
2.Double linked list

```
structdlist
{
        int data;
        structdlist *prev;
        structdlist *next;
}
```

An operations that I have used is creating, insertion at regular node, delete

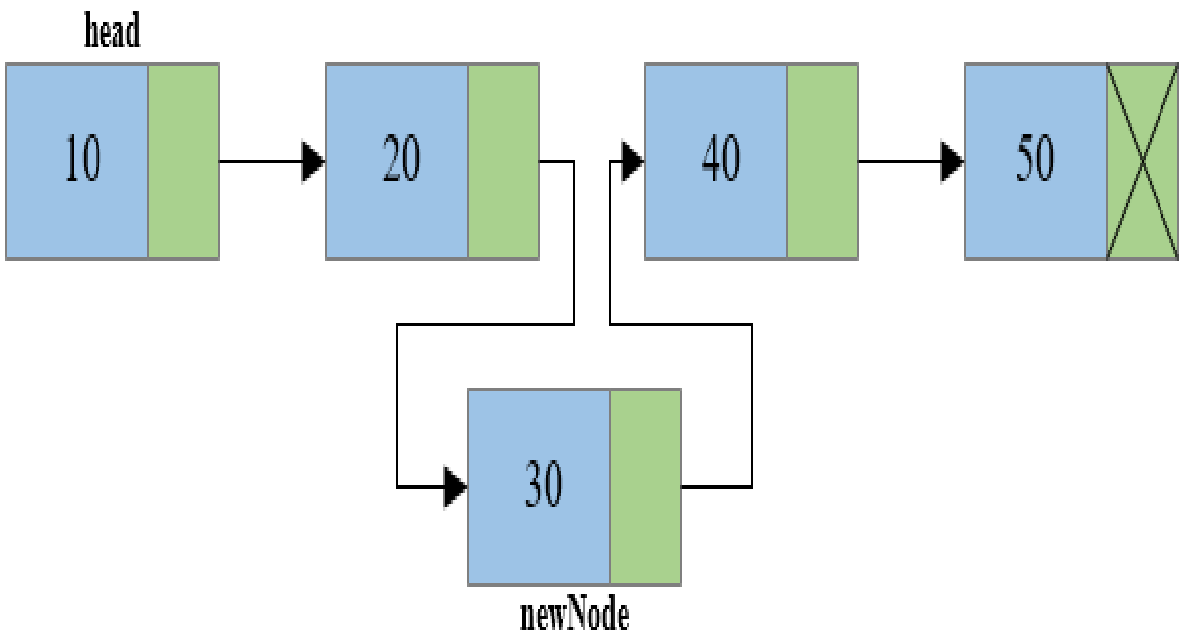
Creating a node:-

With creation we try to create the first node.



Inserting at regular node:-

Inserting a particular node into an existing single linked list.

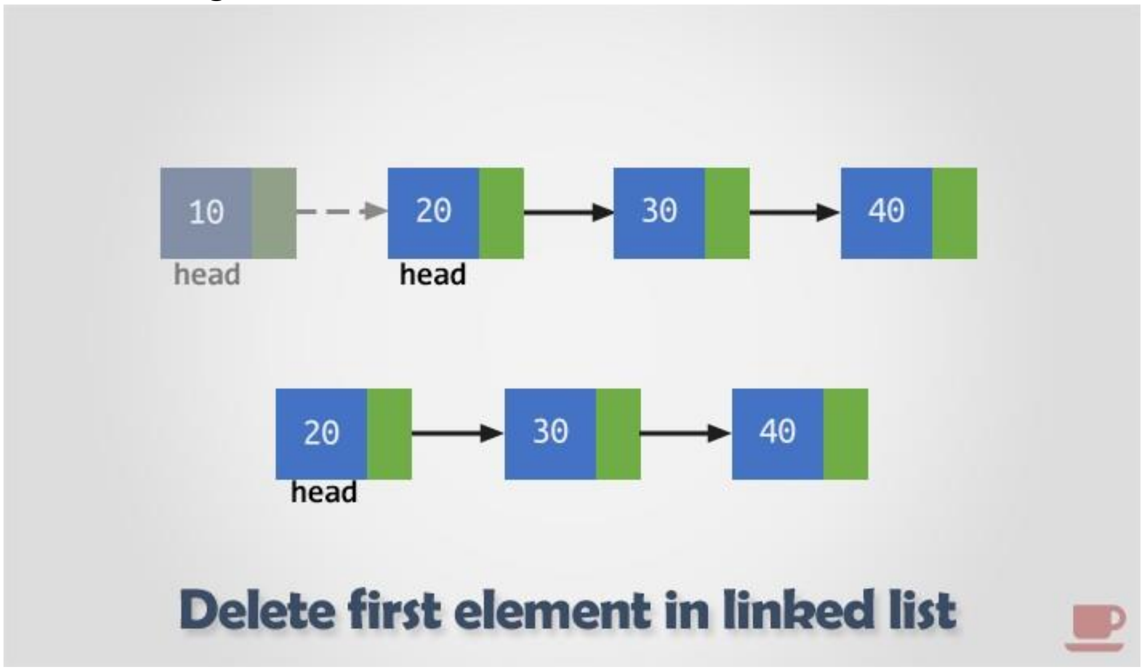


Deleting at specific node:-

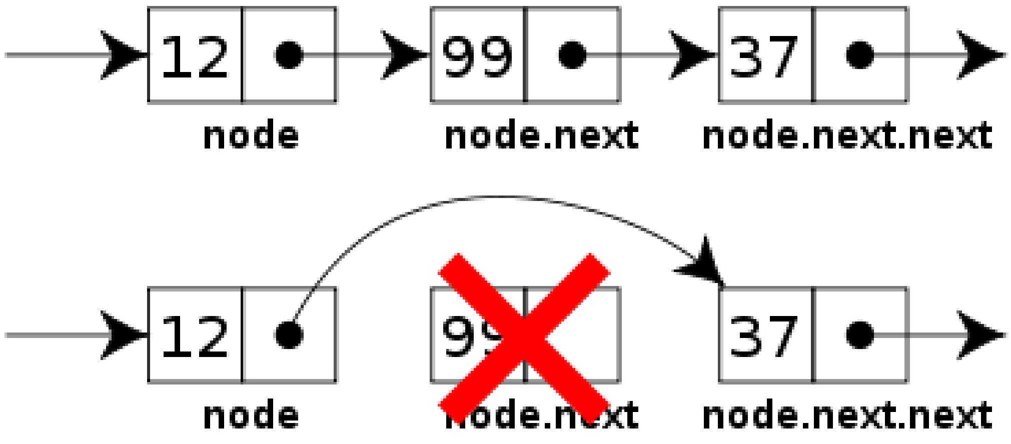
Deleting a specific node from existing single linked list is possible at three positions

1. deleting the first node
2. deleting the middle node
3. deleting the last node.

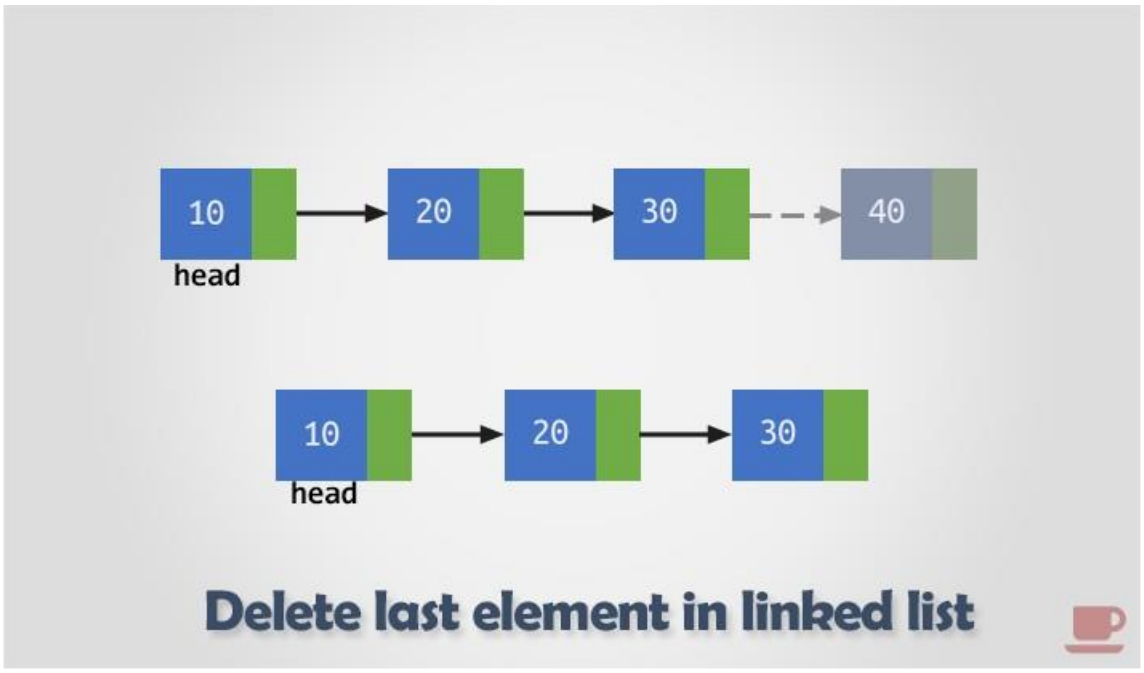
- **Deleting first node:-**



- **Deleting middle node:-**

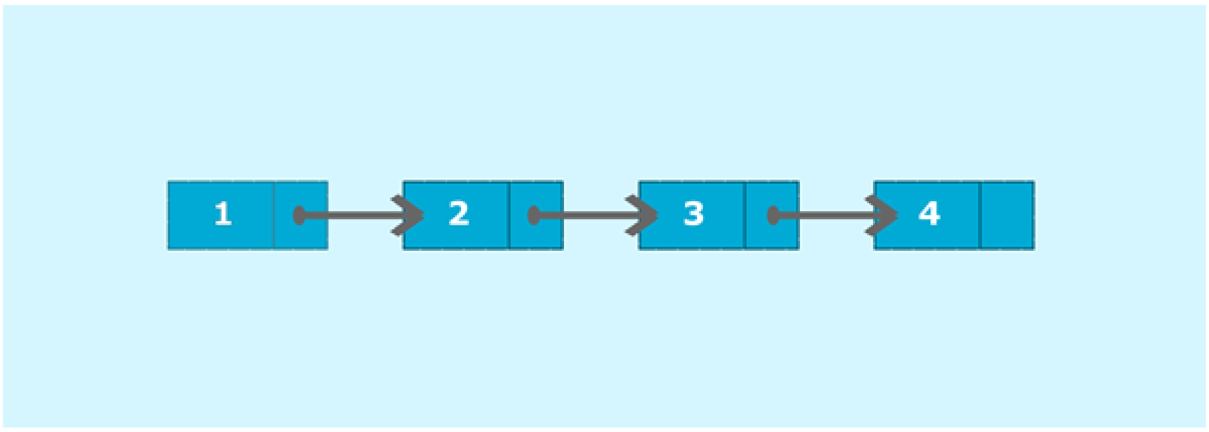


- **Deleting last node:-**



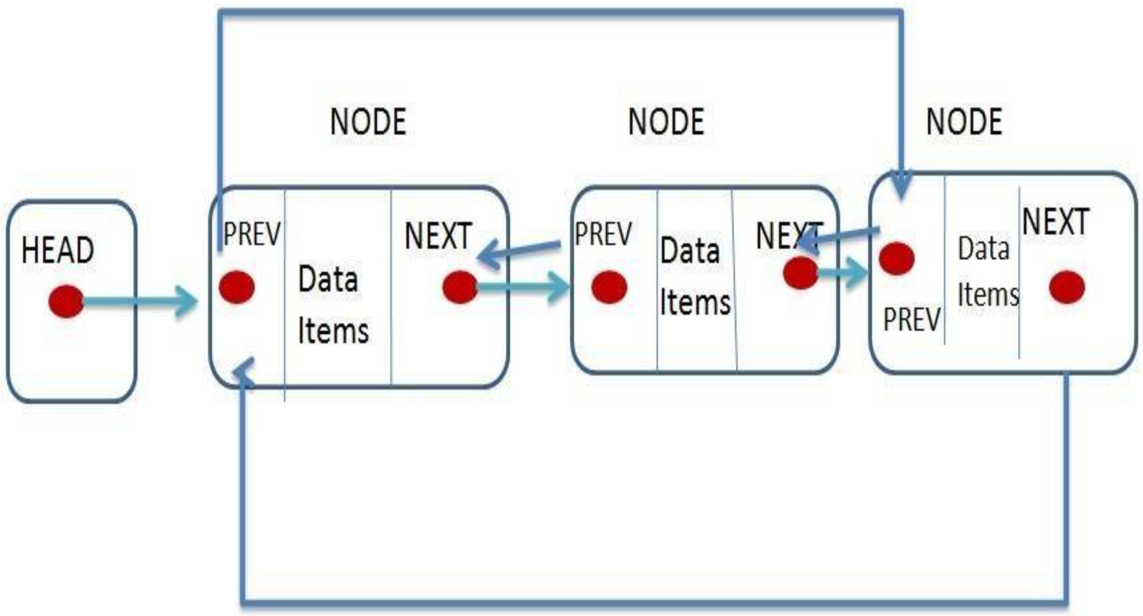
DISPLAY

Displaying all the labor details such as name, designation, total number of working days, worked days, leave taken, cost to company (salary).



2.3 DOUBLE LINKED LIST

Doubly connected list may be a connected organisation that consists of a group of consecutive connected records known as nodes. Each node contains three fields: two linked fields (references to the previous and to the next node in the sequence of nodes and one data field. The beginning and ending nodes' previous and next links, respectively, point to terminator, typically a NULL, to facilitate traversal of the list. The two node links permit traversal of the list in either direction. While adding or removing a node in an exceedingly doubly connected list needs dynamic a lot of links than an equivalent operations on a one by one connected list, the operations ar less complicated and potentially more efficient (for nodes other than the first nodes) because there is no need to keep track of the previous node during traversal or no need to traverse the list to seek out the previous node, in order that its link will be changed.



2.4 Trees:

Trees can be used for finding an any number and it will be used for many operations.
In trees mainly we will be considered as a binary tree , complete binary tree, strict binary tree and binary searching tree.

In binary trees we will have traversal technique. In traversal technique it will be divided into three conversions:

They are: 1. In order
2.pre order
3.post order

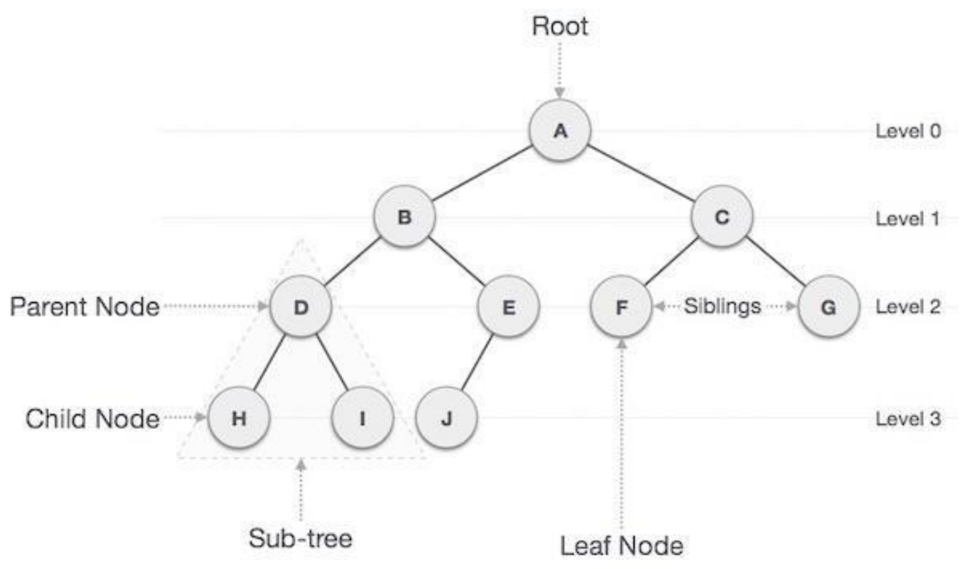
We will be these types of trees and and these orders we will do it a simple programs in manually or linearly.

Trees can be used in some times for linking one process to another process because which process comes first and comes last that will be decided in trees also because in trees there is a parent node and left child is always lesser than a parent node and right child is always greater than a parent node.

When deleting a node in binary search tree the property of a binary search tree (left structure should have lesser than the root element. Right subtree greater than root element should be preserved.

In this operation of tree we have searching an element in a binary search tree is an important topic that should be learnt from the trees lesson. So a binary search tree is a binary tree with the following properties:

- Each and every element is unique.
- For each and every element of parent node the left subtree has lesser value compare to the parent key right subtree has greater value compare to the parent key.
- Compare the key value to be inserted with the root element. If the key value is less it should be insertedto the left side of the root node.
- If the key value is greater it should be inserted to the right side of the root node.
- This applies to all the subtrees in the binary search tree.
- The root node of the key remains same as the root node for binary tree.
- The left subtrees of the binary tree consists the nodes in key.
- The remaining trees forms the right subtree of the binary search tree.
- Every tree should be converted to a binary tree using left child right sibling notation.
- Trees can be used in so many ways for searching any element and any other programs that can be easily convey through trees



INTRODUCTION OF GRAPHS :

Graphs is nothing but a analysing the issue and it is a calculating of any survey or issue and we will be know the clear cut of the damage of an issue or any survey. A graph which contains g(v,e) where v is vertices and e is the edges graph is a combination of vertices and edges.
Graph can also so many types and one of the important object is multigraph and a multigraph is a graph which has some edges which has same purpose. The no of distinct unordered pair where the vertices is same and equal.
If an edge between one vertex to other vertex the value of the graph is 1. If there is no edge between vertex the value is zero. Consider the adjacent node to the vertex v, travel till the depth of the adjacent node. If the node as reached to the leaf node or is adjacent to the already visited keep back tracking and apply the same steps.

CHAPTER -3:

DESIGN

3.1 DESIGN GOALS :

-> file is created to store customer details.

-> Linked list is dynamic data structure.

-> Linked list is used to access each word from the file, which is easier and consumes less time.

-> Linked list can grow and shrink during run time.

-> Efficient memory utilization that is no need to pre allocate memory.

-> We can insert any node at any place easily and similarly we can remove it easily.

-> Insertion and deletion Operations are easier.

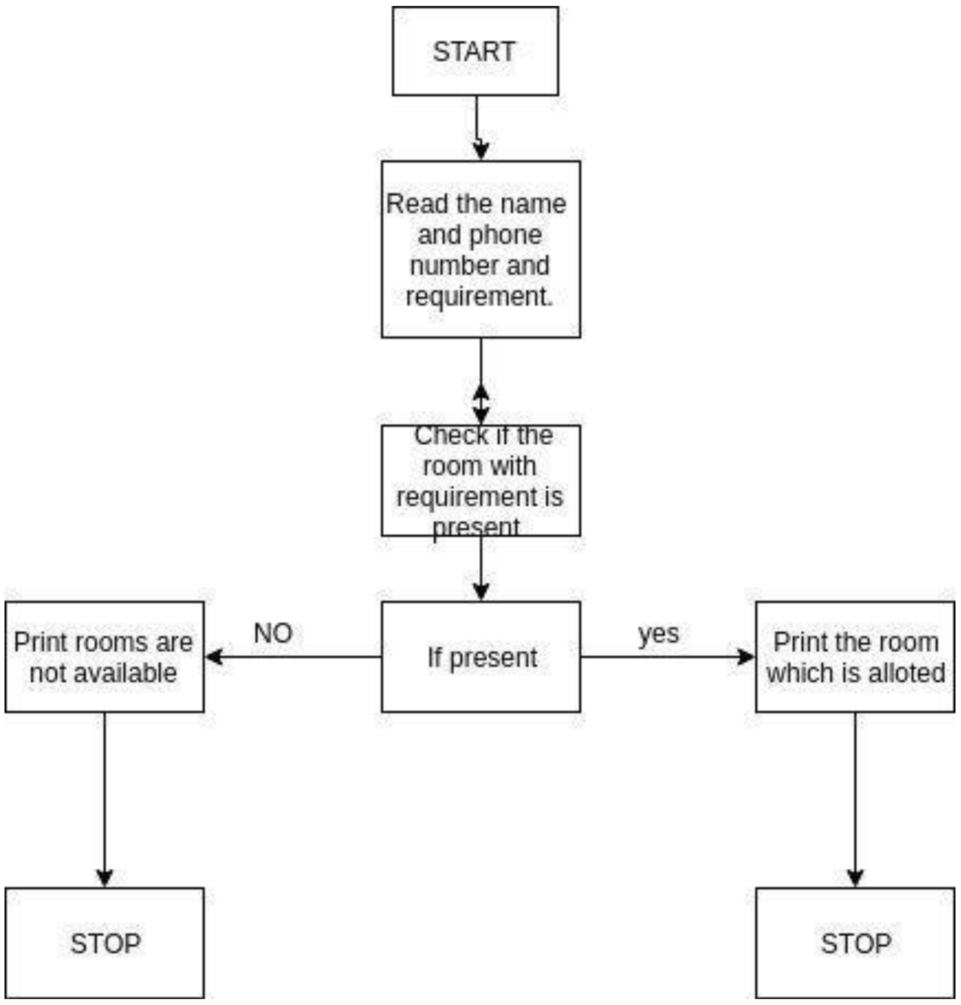
-> We don't have to shift nodes like array insertion. In insertion operation in linked list, we have to just update next link of the node.

-> Faster access time, can be expanded in constant time without memory overhead.

-> Insertion and deletion operations in Linked list is very flexible.

3.2 ALGORITHM :

1. enter the login details
2. If login details do not match exit
3. Enter the option op
4. switch(op){
5. case 1: Add - Read the details of customer and add those details along with the vacant room number to node of the list.
6. case 2: Delete -
    A. Read the room number to be deleted.
    B. Search the list for this room number and delete that node from the list.
    C. Add the room number to the list of vacant rooms
7. case 3: Display the customer record
    A. Read the room number of the customer
    B. search the list for this room number
    C. If the room number is not found
    D. Print that the room is not allocated for any customer and read other option
    E. Print all the details of the customer


START
Read the name and phone number and requirement.
Check if the room with requirement is present
If present
NO
yes
Print rooms are not available
Print the room which is alloted
STOP
STOP

CHAPTER – 4:

IMPLEMENTATION

4.1 MODULE 1: CREATING FOOD MENU

Creating a food menu for a customer to display the items.Here i'm displaying two items creating food menu:-
list of food items present in the hotel will be created. creating a memory allocation for 1 st food item. copying paneer pizza into temporary variable using strcpy. allocating price for it i.e 100/-.making temp of next as null
Menu = temp, creating a memory allocation for the second food item and copying baby corn pizza into temporary variable by using strcpy function. setting price for it i.e 120/- temp1 of price is equals to 120 menu of next is equals to temp1.

4.2 MODULE 2: DISPLAY CUSTOMER DETAILS:-

struct customer_details *customer = customer_list variable customer list is created
“if(customer list = null)”
checking if customer list is equal to null . printing here “no customers”. Using while loop checking whether customer is not equal to null or not .printing room,name,phone number,bill.
customer->room_no,customer>name,customer->phone_no,customer->bill). customer = customer->next.

4.3 MODULE 2: CREATE ROOM LIST

list of rooms along with its details (AC or NON AC, vacant or occupied, price) will be created.By using ‘for’ loop the number of rooms can be created . Assuming first 10 as AC and remaining 10 as NON-AC. So 20 rooms were created.A node (memory allocation)is created to a struct room in a temporary variable and a room no is taken as “i” in for loop so by this the room can be allocated and traverse to the next node in the list. Inserting “ i ” into temp of room number as “temp->room_no = i”.Indicating vacant room as “ 1 “ in the temporary variable “temp->vacant = 1 ``Making the next node pointer of temporary variable as null temp->next = NULL. Putting condition using “if” condition whether required room number is less than 10 declaring it as AC. Allotting a price for it and i.e 1000(AC).

if(i<=10)

->Declaring AC as 1.

->its price = 1000. (then)
allocating NON-AC as 0 its price = 500;

4.4 MODULE 3: ADD CUSTOMER

The details of the customer will be added in this linked list for example In a character the requirements ,name and phone number were initialized.Here by using printf statement name and phone number need to be printed.
->requirements (ac or non-ac)

initializing here for requirements ,name phone number.struct room *tem = room_list; while(tem!=NULL)
if((!strcmp(req,"AC"))&&tem->vacant==1&&tem->AC == 1)

if((!strcmp(req,"NONAC"))&&tem->vacant==1&&tem->AC == 0) if(tem!=NULL)
allocating a memory for customer details.Copying name into temporary customer name. Copying phone number in temp phone number.Making temp customer as null. tempcustomer->room_no = tem->room_no. printing the room is allotted. tempcustomer->bill=tem->price.if customer list is equals to null then customer list is equal to temp customer and temp customer of next is equal to customer list. customerlist=temp_customer tempcustomer->next=customer_list;

4.5 MODULE 4: REMOVE THE CUSTOMER

If customer wants to leave the hotel the entire details will be deleted(freed)
if(customer list != NULL) struct customer_details *vacate_customer = customer_list.if the

vacate_customer->room_no == room .Found will be printed.Traversing to customer list node.Freeing the customer who has founded..
while(vacate_customer->next!=NULL) if(vacate_customer->next->room_no == room) struct customer_details *tmp = vacate_customer->next.incrementing vacate customer. vacate_customer=vacate_customer->next
freeing the customer

struct room *vacate_room = room_list By using while loop we need to check customer room is not equal to null print room is vacant

4.6 MODULE 5: RECEIVE THE FOOD

customer can receive food directly to the room which is alloted.

initializing dish and room no.customer is able to order food from the food menu the food bill is also added to the total bill of the customer and is collected, when he vacates the room struct food *Dish = menu printing food menu while(Dish != NULL) printing dish which is required and incrementing to the next dish. Dish = Dish->next. Here I'm Printing room no and dish to which room it should delivered
While (Dish != NULL) If(comparing item and dish) price = Dish->price;
printing food price and here
Creating a variable of customer in a structure. while(order customer!=NULL)
if(order customer->room_no == room) printing order customer bill.

CHAPTER :5

RESULTS

Case 1:
Entering customer details.
Entering name and phone number.

Case 2:
Allotting room for the customer.
Entering the type of room and room number.
Case 3:
Food order
By ordering food the amount of the food bill is added to the total bill.

Case 4:

Display
Displaying the details and total amount of bill.

CHAPTER- 6:

CONCLUSION

By this project when people go to new places they can plan easily by booking the rooms and they can reduce their work for searching rooms to stay . And the cost of the rooms are affordable. They can order their food .

* It has been developed in visuality basics keeping in mind the tasks of the system.

* For designing the code we have used simple data flow diagrams.

*Overall the given project teaches us the important essential required skills .

I thank my reviewer Mr. Gagan sir for helping me to complete this project. I have learned many things about data structures especially linked list through this project

CHAPTER-7 :

REFERENCES

Class notes
goggle