Q.2 a. Write the preorder traversal algorithm of binary tree. Explain with the help of an example.

Answer:
If a tree, "T" is traversed in preorder fashion then the root node of "T" is visited first and then the left sub-tree of "T" is traversed and finally the right subtree of "T" is traversed.I T is NULL then traversal of "T" involves doing nothing.

```
void preorder(bnode *T)
{
        (if T!=NULL)
        {
                printf("%c", T->data);
                preorder(T->left);
                preorder(T->right);
                }
                        return;
                        }
```

Q.2 b. Explain the algorithm for searching a target key in a binary search tree.

Answer:
Searching for a key in binary search tree can be easily defined recursively. The key is compared with the key at the root. If they are equal then 'search' terminates successfully. If they are equal then 'search 'terminates successfully. If the given key is less than the key at the root, then search is initiated only at the left sub-tree of the current root.

Otherwise, the right subtree of the current root is to be searched. The 'search' process terminates unsuccessfully when search is done on an empty tree. Before describing the recursive function for search, it is required to define the data type for a node in a binary tree. It is assumed that the information field in each node of the tree is an integer.

Q.3 a. Write an algorithm in C language syntax to delete the ith node in a singly linked list.

Answer:

In a linked list in which the node pointed to by "x" is to be deleted ,then node that is affected by this deletion is the node which precedes the node pointed by by "x".So to perform delete function ,the pointer to the node preceding the node to be deleted is required to be found out.
Let "y" be the pointer to the node preceding the node pointed to by "x".After deletion , the "next" of the node pointed to by the "y" is to be set to the next node of the node pointed to by "x".
The " C" function to delete the ith node in a list "p"

```
void deletelist(intnode **p,int i)
        {
```

```
                    intnode *y, *x;
                    int j;
                    y =*p;
                    if(i==0)
        {
                *p= (*p)->next;
                tree(y);
                    return;
}               x=y->next;
        for(j=1;j<i && (x!=NULL); j++)
        {
                y=x;
```

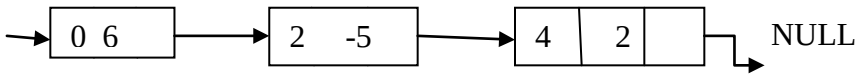
Q.3 b. how linked list can be used for representing polynomials? Explain using a suitable example.

Answer:

Polynomials can be thought as an ordered list of non-zero elements. Each non-zero term is a 2-tuple containing two pieces of information, the exponent part and the coefficient part.

If the polynomial 2x^4 - 5x^2 + 6 is represented as:
((0,6) , (2, -5),(4, 2))

The first tuple indicates the term 6.x^0 , i.e. 6; the second tuple indicates -5. x^2 and the last tuple denotes 2x^4 .The tuples are arranged in increasing order of their exponent part, that means the first tuple contains the non-zero term with the least power of "x" and the last tuple contains the non-zero term with the highest power of x . The linked representation of this polynomial is as:-



Each term in the polynomial can be defined as the following struct:-

```
typedef struct polynode
{
   int coefficient;
   int power;
   struct polynode *nextnode; }  polynode;
```

Q.4 a. Represent stacks using a single one dimensional array. Write and explain function for "push" and "pop" operation on stack.

Answer:

The most primitive representation of a stack can be done by using a one-dimensional array of data records. In addition to this, another variable "top" is required to keep track of the index of the last inserted element.

So, a stack can be defined as follows:-

```
typedef struct  stack {
        datatype anarray[maxsize];
        int top; }stack;
stack astack;
```

Function to push a data element in a stack:-

```
Int push ( datatype a,stack *x)
{
        If(x->top == maxsize)
                return 0;
else
        {
        x->top +=1;
        x->anarray[x->top]=a;
        return 1;
```

Q.4 b. Write a C program that shows implementation of priority queue.

Answer: Page no. 256 text book

Q.5 a. Write an algorithm that searches an element using binary search method.

Answer:

For a binay search, it is mandatory to have the sorted list in ascending order.

Algorithm:-

```
int binarysearch(int k,int a[],int n)
{
        int lower=0,mid;
        int i= -1;
        int upper;
        upper= n-1;
        while(upper>=lower)
{
        Mid=(upper+lower)/2;
        If ( k== a[mid])
{
        i=mid;
        break;
}
else
{
        if (k>a[mid])
        lower=mid+1;
else
        upper=mid-1;
}}
return(i)    }
```

Q.5 b. Describe merge sort using an example.

Answer:

Merge sort is a typical "divide and conquer" sorting algorithm. It divides the list in to two equal halves, sort them and finally merges these sorted lists. As the merge operation may be done in linear time, the worst case complexity of merge sort is O(n log n).The major shortcoming of this approach is that a single implementation requires an O(n) additional space during the merge operation.

This algorithm has a natural non-recursive version which works in a bottom up manner.

Another advantage of merge sort is that it can be conveniently implemented on linked representation of lists.

This method is stable but data is insensitive. Number of comparison in merge sort is nlogn.

Q.6 a. What are different ways of representing a graph? Explain using suitable example.

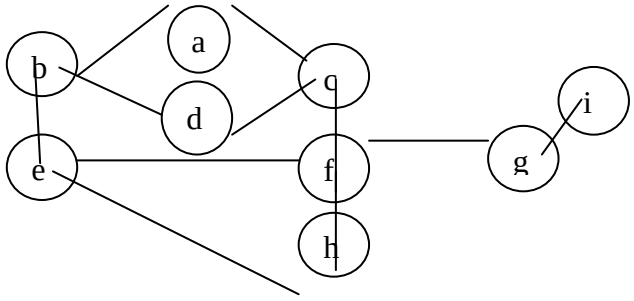
Answer:

The adjacency matrix or the connectivity matrix "A" representing an undirected graph "G" is defined as follows:

The rows and columns of "A" correspond to the vertices of "G". If the element "a(i,j)" of "A" is "1" then there is an edge from the vertex v(i) to the vertex v(j) in "G".

If a(i,j) is 0 then the graph "G" does not contain an edge from v(i) to v(j).

The adajency matrix of the graph of following graph is shown:-



Undirected graph has set of vertices: V={ a,b,c,d,e,f,g,h,i};

The set of edges are: E = {

(a,b),(c,a),(b,c),(d,b),(b,e),(c,d),(c,f),(e,f),(e,h),(f,h),(g,f),(i,g) }

Adjacency matrix for above graph:-

	a	b	c	d	e	f	g	h	I
a	0	1	1	0	0	0	0	0	0
b	1	0	1	1	1	0	0	0	0
c	1	1	0	1	0	1	0	0	0
d	0	1	1	0	0	0	0	0	0
e	0	1	0	0	0	1	0	1	0
f	0	0	1	0	1	0	1	1	0
g	0	0	0	0	0	1	0	0	1
h	0	0	0	0	1	1	0	0	0
i	0	0	0	0	0	0	1	0	0

Q.6 b. What is minimum cost spanning tree? Using a suitable example, derive Depth first spanning tree.

Answer:

DFS algorithm is from a vertex 'i' would mean marking the vertex 'i' as visited and initiating DFS from a vertex 'j' where 'j' is an unvisited adjacent vertex of the vertex 'i'.

Algorithm DFS:-

```
void dfsearch( int start)
{
        int v;
        adjvert 8adj;
        visited[start] = 1;
        printf("%d",start);
        adj=g->adjlist[start];
        while(adj != NULL)
{
        v= adj->vertex;
        if(!visited[v])
                dfsearch(v);
        adj= adj->next;
} }
```

Q.7 a. Write a 'C' function to insert a node at the end of a doubly linked list.

Answer:

The "C" function for inserting a node pointed to by "t" in a doubly linked list (*p) after a node pointed to by "x" is given below:

```
void doubleinsert(intdnode **p,intdnode *x, intdnode *t)
        {
        t->right = x->right;
        t->left= x;
        x->right=t;
                if( x->right != NULL)
                x->right ->left = t;
}
```

Q.7 b. Write an algorithm to merge two circular linked lists.

Answer:

The first operation is concatenation of two NULL terminated linked lists given by "p" and "q".After concatenation ,"p" will point to the concatenated list.

```
intnode *concatenate ( intnode *p, intnode *q)
{
        intnode *r;
        if(p== NULL)
        p=q;
        else
{
        r=p;
while (r->next !=NULL)
        r= r->next;
        r->next=q;
}       return p;    }
```

Q.8 a. Using a suitable example program, explain stack overheads in recursion.

Answer: Page no. 128 of text book

Q.8 b. Explain with differences the following three functions used for dynamic memory allocation:
Mallow, relloc, aclloc

Answer: Page no. 123 of text book

Q.9 a. Differentiate between structures and unions using suitable example.

Answer:

Actual parameter: - It is a parameter , which is used in function call to send the value from calling environment.

Formal parameter;- It is a parameter, which is used in function header , to receive the value from actual parameter.

Example:-

```
void calc(int t)   /* t is a formal parameter */
{       printf("%d\n", 5*t);  }
void main()
{   int a= 45;
        calc(a);  /*  a is actual parameter  */    }
```

Q.9 b. Write a C program that Implements a direct access file.

Answer: Page no. 156 of text book

Text Book

C& Data structures, P.S. Desponded & O.G. kerbed, Dramatech press, 2007