Exam 1

6 iii.)

f(n)=(n+2)^4 (2n+1)^2 + (n^3 – 2n + 1) log(n^5+5)

O(n^4).O(n^2) + O(n^3) . O(logn)

O(n^4.n^2) + O(n^3 log(n))

O(n^6) + O(n^3logn)

max = O(n^6)

A= {3,5,1,2}

B= {1,7,3}\_\_\_\_\_\_

AUB = {3,5,1,2,7}

A🡪[1|]->[2|]->[3|]->[5|/]

B🡪[1|]->[2|]->[7|/]

ORDEREDLL \*AUB;

p=A; q=B;

while(!p.empty\_o() && !q.empty\_o())

{

int x1 = p->info;

int x2 = q->info;

if(x1==x2)

{

AUB.push\_o(x1);

q=q->next;

//p and q are pointing to 2nd value, 2

}//endif

else if(x1<x2)

{

AUB.push\_o(x1);

p=p->next

}//endelseif

else//x1>x2

{

AUB.push\_o(x2);

q=q->next;

}

/\*TRACING

A🡪[1|]->[2|]->[3|]->[5|/]

B🡪[1|]->[2|]->[7|/]

p is pointing to info 1 and that first location

q is pointing to info 1 and that first location

go to loop. . .

then q is 7. . .?

\_\_\_\_\_\_\_\_\_\_\_\_

x1=x

AUB->[1|]->[2|]->[3|]->[5|/]

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

if(q.empty\_o())

{

//there are more items left in set B

while(!q.empty\_o())

{

q.push.o(x2)

q=q->next;

x2=q->info;

}//endwhile

}

if(!p.empty\_o())

{

AUB.push\_o(x1);

p=p->next;

x1=p->info;

}

}//endwhile

CIRCULAR LINKED LIST

list🡪[3|]->[9|]->[2|/]

start over

list🡪[3|]->[9|]->[2|]

1. write statements to convert a regular linked list to a circular linked list

-we need a pointe rto point to the last node and = it to the beginning of the list

node \*p;

p=list;

//to stop loop we stop when ->next == NULL

while(p->next!=NULL)

{

p=p->next;

}//endwhile

//when we exit this loop p is pointing to last node which is [2|/]

p->next=list; //goes back to the beginning

1. list🡪[3|]->[9]->[2|]

display the content of all nodes

p=list;

//do while ensures the loop once

do{

cout<<p->info;

p=p->next;

}while(p!=list;)

//output 3 9 2

1. insert 10 at the rear of circular list

list🡪[3|]->[9|]->[2|]

//at the end we want to insert q->[10|];

//we’ll want to connect list->[2|] to q->[10|] and connect it back to list

p=list;

while(p->next!=list)

{[=p->next;} //pointing to last node

node \*q = new(node);

q->info = 10;

1. p->next = q;
2. q->next=list;

DOUBLE LINKED LIST

list🡪[/|10|]->[|5|]->[|15|/] //with this we can’t go back

a10 a5 a15

//we’ll need something to access previos nodes

struct node

{

int info;

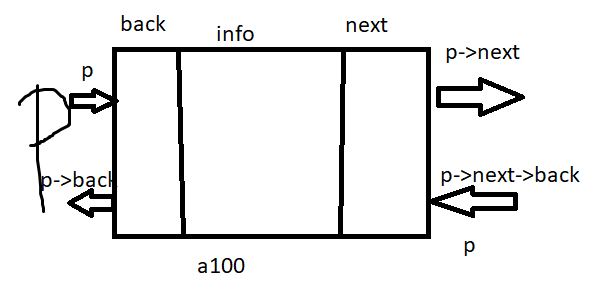
node \*next;

node \*back;

}//end strucy

node \*p;

p=new(node);



list🡪[/|10|]->[|5|]->[|15|/]

a10 a5 a15

i.)visit all nodes from left to right

node \*r;

r= list; //r is a pointer to a node and is pointing same as list

while(r!=NULL)

{

cout<<r->info<<”\_“;

r=r->next;

}//endwile

//output: 10\_5\_15

ii.) visit nodes from right to left

//we’ll need a pointer to point to the last node

//make r point to the last node

r=list;

while(r->next!=NULL)

{

r=r->next;

}//endwhile

while(r!=NULL) //bec [/|10|]

{

cout<<r->info<<”\_”;

r=r->back;

}//endwhile

//output: 15\_5\_10

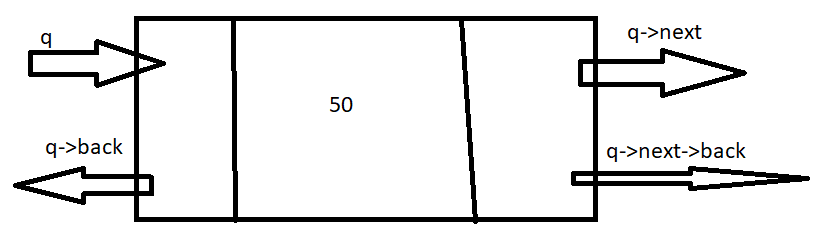
1. insert 50 before the node with info=15;

list->[/|10|]->[|5|]🡨-----🡪[|15|/]

q->[|50|]

q=new node;

q->info=50;



//have another pointer r and p

//we want r to follow p

r=p=list;

while(p->info!=15)//if it is 15 we exit

{

r=p;

p=p->next;

}//endwhile

//we want to change [|5|a15] to a[|5|a50] instead because the address of the q is a50

1. r->next=q; //now we have [|5|a50] and we want address of back to be 5 so [a5|50|]
2. q->back=r;

//we now want to connect q to p address [a5|50|a15]

1. q->next p; //we want p->back to be a50 so [a50|15|/]
2. p->back=q;

IV) delete the node with info = 5;

list->[/|10|]<->[|5|]<->[|15|/]

make r to point to node with info 5

and let p to follow r

p=r=list;

while(r->info!=5)

{

p=r;

r=r->next;

}

1. p->next = r->next; //r info is 5, r->next is info 15
2. r->next->back=p;//go to address a15 and connect it’s back to p->next bec p is pointing to a10
3. delete(r); //removes [a10|5|a15]

CIRCULAR DOUBLY LINKED LIST!!

list->[|10|a5]<->[a10|5|a15]<->[a5|15|]

ANOTHER APPLICATION OF LINKED LIST

Hashing

//if u have group of

[714]->[ |/]->[ |/]->. . .[ |/]

[909]->[ |/]->[ |/]

[949]->[ |/]->[ |/]

[562]->[ |/]->[ |/]

[310]->[ |/]->[ |/]

//what is the fastest way to organize these area codes

HASHING IS A METHOD TO MAKE SEARCHING PROCESS FASTER

Data

EX: hash table of size 5

[\_]--H[0]->[~~10~~|]->[10|]->[10|/]

[\_]--H[1]->[~~11~~|]->[11|/]

[\_]--H[2]->[7}]

[\_]--H[3]->[~~13~~|]

[\_]--H[4]->[~~19~~|]

10

11

13

19 i=Data%5

20

5

7

21

18

9