	Week-5
25 - 1 - 24	Linked list operations: - sort, severse, concaterate, warren
(0)	Linked hist operation
	# 1 1 (1) 1 >
	#include < stellib. h> #include < stellib. h>
	# include Letelles. h
	struct Node
	1
	it data i
	stemt Node * next;
	3;
	typedel steut Node Node;
	Node * exeate Node (int value)
Take !	Established they day down holden
	Node * new Node = (Node *) nallox (size of (Node));
	new Node -> dota = nalue;
	rewNode → rest = NULL;
	return now Node:
	}
	void display (Node * head)
	{
	while (head! = NULL) {
	printf (" "d → ", head → data);
	head = head - next;
	· · ·
	printf ("NULL \n");
	<u></u>
-	Node * sort List (Node * head)
	<u>{</u>
Of the state of th	if (head == NULL 11 head -> next == NULL)
The state of the s	return head;
	int swapped;
	Node *temp;
	Node * erd = NULL;
	do
	1

```
swapped = 0;
           temp = head; while (temp - rest! = ad) {
           steif (temp - data > tong - vert - data)
             int temp Pata = temp - data:
              temp - data = temp - next - data;
             temp - next -> data = temp Data;
            swapped = -1;
         temp - temp - next:
   3 while (swapped) it to the
 Node * severelist (Node * head)
    Node * pear = NULL; shill have
    Node * ensent = head; ((still
    Node * next Node = NULLE 1211
    while ( everent != NULL) . til
        nextivale = wigest - aest:
        quesent - rest = prev
       entert = next Node;
Node * concot Lists (Node * list 1, Node * list 2)
     if (list1 == list 2)
```

Node * temp = list1; while (temp = next! = NULL) turnp = temp - next; temp - next = list2; eituen list 2; Node * list 2 = create Node (3); list = rest = create Node (1); list 2 - next - rest = create Node (4); Node *list 2 = create Node (2); list 2 - rest = create Node (5); printf (" original list 1:1 " "); display (list1); printf ("original list 2: ") in display (list 2); list 1 = sortList (list1); printf (" sorted list: "): 1000 = 1 display (list 1); list 1 = reverse List (list 1) in = douten printfl" severed List 1: "); display (list 1); Node * concaterated = concathists (list I, list 2) printf (" concaterated list: ");
display (concaterated); Output: original list 2 1 3-1, -4 - NULL original list 2: 2-5 -> NULL

4-3-1-2-5 -> NULL

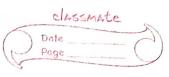
concaterated list:

>	Implement stack using linked list #include (stdio, h)
_	hadren = Jakala
	#inlude < etdio, h>
	Firelude < stdlib. h > (Int + to shall) you
	struct Nede (1300)
	Strut Nede
	1 1 1
	strut Node Frest
	Node * create Node (it value)
	Node * create Node (it value)
	E sucke V berger needed
_	Node * new Node = (Node *) malloc (size of (Node));
	rewNode → data = value:
	new Nordo -> next = NVLL;
	return new Node;
	Bunk Kentadal
	void display (Nacle * head)
	f ("al continuo data") There
	while (head ! = NULL)
	(05 date 3) de
	printf (" /d > " head - data):
	head = head - next;
	? (at tate) which
	printf ("NULLA In"); "
	3 contract of super banas I they
	typedel strut {
	Node *top:
	3 Linked List:
	an trees dista 1 also
	void push (LinkedList x stack, int value)
	OF who were
	Node * new Norde = create Node (value);

```
new Norle - next = stack - top;
         stack - top = new Nede i
 int pop (LinkedList * stack)
     if ( stack - top == NULL)
          psintf ("stack is empty: ");
     int popped Value = stack -> top -> data
     Node * temp = stack
     stack - top = stack -
     free (temp)
     ectuen popped Values
void main ()
      printf (" stack operations: \n'):
      push ( & stack 10);
      puch (4 stark 20);
        h (4 stark 25);
     puch (& stark 30);
     display (stark, top);
printf ("popper value:
     printly (" popped value: ", pop (&
     display (stack top);
Output:
           stack specations.
            30 + 25 + 20 - 10 - NULL
           popped value: 30
            20 - 10 - NULL
```

3)	Implement queue ring Lished list
The state of the s	#. 1. 1
and the second second section of the second	#include < stdio. h>
	#inelude <stdlib b=""></stdlib>
	Stewt Neele
	int data;
	3;
	typedef struct Node Node;
	lypedef steut Node Node; Node * createNode (int value) 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
	Node * new Node = (Node*) mallor (riged (Node));
	new Node -> data = value;
	newNode > next = NULL;
	Machin new Node:
	S LAND TO LAND TO A STATE OF THE STATE OF TH
	void display (Node *head)
	State of the state
	while (head! = NULL)
	}
	pentf ("%d → ", head → data);
	head = head - next; () june him
	prints ("NULL \n");
	3 NM = trasf sure
	typedef stenet &
	Node * front:
-	Node * seas:
	3 LinkedList;
	void erqueue (Liskedlist *queue, jet value)
	{ de transportante
11. 16	Node * new Node = create Node (value);

-	if(queue - front = = NULL):
	queue -> front = new Nocle;
	que -> sear = rew Nodes
	else
	Y
	queur - rent - new Node;
	queue - rear = new Norde;
	3
	int dequeux (Linkedlist *queux)
	<i>f</i>
	if (queue -> front = = NVIL)
	prints ('queue is empty:));
	setuen -1; I will a tour continue
	int dequenedralue = queue > feart > data:
	Nede * temp = queue -> front;
	queue - front = queue - front - next;
_	Jace (temp);
	return dequeredralue;
	3
	Mathematical - b. 1" that
_	void main ()
	17
_	LinkedList queue; (d) sin Ifini
_	queue. front = NVIL;
	queue reage = NULL:
	prints ("In queue operation: In");
	erqueue (Equeue, 40);
-	enqueue (& queue, 50); enqueue (& queue, 60);
-	dienland (anom A. A)
	prints (" dequeued from queue: "d)" dequeue (& queue)
	print aqueue from queue: /d In dequeue aqueue



	prints ('dequeecd from queue : "d \n", dequeue (rqueue)); display (queue front);
	display (queue front):
	3
	output:
	queue operation:
	40 - 50 - 60 -NULL
	dequeued from exqueue: 40
	dequeued from exquere: 50
	60 - NULL
1.2	
1	
	The state of the s