lab-6

1) Stack implementation using single linked light #inchede Litcleo, h> #include < stdlib.h> struct Node of int data; struct Node \*next; void désplay ( struct Node \*+op) { if (top!=NULL) printy ("stack elements are: 14"); while (top!=NULL)d printj ("11 1+", top-data); top=top-nent; printf("stack is empty \n"); Struct Node \* push (struct Node \*top, int date) + struct Node \* new Node = (struct Node \*) malloc( 872 eg (struct Node); if ( new Node = = NULL) { prointy ( "stack over your ("); return top;

rewNode-data = data; newNode-next = topi top = new Node; return top; struct Node\* popl struct Node \* top, int \* poplata){ if ( top == NULL) { printf("Itack Underflowin"); \*popdata = -1; return NULL'S Struct Node \* temp = top; \* pop Data = temp- > data; top = top -> nent; gallu tree (temp); return top; int main() ( int op, n, popElement; struct Node \* top = NULL; printfl" Enter 1. Push in . 2. Popin 3. -1 to stop (n"); while (1) d printf ("Enter operation: In"): scanf ("1.d", &op); i) ( op ==-1) { print ("Eniculian stopped in"): break;

Date Page
Switch (OP) {
case 1:
prints ("Frer the element to push in");
scanf ("1.2", &n);
top=pwh(top,n);
break;
case 2:
top = pop(top, & popElement);
i) ( popElement!=-1){
pring ("Popped Element: \$7.d
In", popElement);
y
break;
default: print ("Invalid choice is");
3
lipay(top);
3
return 0;
9
010
Enter 1. Push
2. Pop
31 to stop
Enter operation
2
Stack underflow
Stack is empty
Enter operation;
Enter element to push 5
Stack elements arc: 5
Enter operation 1

Enter element to push 4 etack elements are: 4 Enter operation:1 Enter element to push Stack elements are: 6 4. 5 Enter operatation: 2 popped element: 6 stack dements are; 4 5 Enter operation: 1: Encution stopped 2) Quelle implementation using single linked list. #include (Stdio, h) #include < stallib, h> smict Node & int duta; smyet rode \* ment; Froid display ( struct node \* front ) { ( ) ( tront == NULL) { prolog ("Queue is empty In"); reharn: Struct Node & temp = tront; mintfl " Occere dements are: It"); while (temp!= NULL)1 printf("1.0 It", temp-reader); temp = temp - next; print ("In"); 4

Page void enqueue correct voile \*\* tront; smut Node the rear, into duta) struct Node + Hew Node = (struct Node +) mallo ( fre of (struct Node)); if I new Node = = NULL){ ' print (" leeue overflow In"); return; new Node -> date = data; new Node + nent = NULL; ij ( + rear = = NULL) { \* front = + reax = new Node: return; (trear) -7 nent = new Nocle; \*rcur = new Node; int dequeuel struct Node \*\* tront, struct Node Atrear )2 if ( throat == NULL) 1 prosif(" dereue Underflow \n"); return -1; · sweet Node \* temp= \* bront; int degreene Data = temp-idates; \*tront = (+tront) - 7 nent; if (\* WON = NULL)1 \$ rear = NULL tree(temp); return dequeue Dates; 3

int main () { int op, o, dequellement; Struct Node \* host = NULL; struct Node \* rear = NVLL; mint ("Enter 1. Enqueuen 2. Dequeue in 3,-1 to stopm"); while (1) d printf("Enter operationin"); scanf ("1-d", bop); if (op==-1)d printf ("Encution Stoped In"); break: 3 switch (op) & ccese 1: printdue ster the element to enquelle in"); scarf ("10", bn); enquere (& tront & rear, n); break; cufe 2: dequille Flegment = dequille (4 Front, Grear): if (dequeue Element != -1)/ printy (" Dequelued element : Y.d'it, degreerette ment); break; displace (mont); returno:

Enter 1. Enqueue 2. Dequeue 3. -1 to stop Enter operation: 2 ducue underflow anche is empty Enter operation: 1 Enler the clement to engille Queue elements are: 3 Enter operation; 1 Enter element to enqueue 2 Queue elements are: 3 2 Enter speration: 1 Enter element to enqueue ! ducue elements ave: 3 2 1 Enter operation: 2 required clement; 3 ducue elements ure ; 2 1 Enter operation: -1 Enrurion stopped. 3) standard linked list sorting #include < stdiv. h> #include < stellib.h> Struct Noder indades struct Noile \* nent;

void display (struct Node\* head) { struct Node urrent = head; while (current != NULL) { mint ("1.8 It", current slater); current = current = rent; pring("In"); struct Node sostlinkedlist (struct Nade \*head) i) ( head == NULL 11 head - nent == NULL) return head; int swapped; sweet Node \* temp; struct Noile xend = NULL do swapped = 0; smuch Noile \* werest = head: while ( werent = end) if (urrent -rduta) urrent -rent-dutal d int temptata = current + data. current - data = current = nent-7 dati, current - ment- data = temp Data; swapped=1: 3 were = were + rent while (snapped); return head;

int main ( )d Struct Node & mealet = (structNale ) mallor (size y (smuthal); s went Nocle Anol 2 = (Sweet Node +) mallar ( +) eg (smith d) struct Node \* node3 = (truct Nodes) mellor (size of waithres); struct Node \* node4= (struct Node\*) mallocliney (irridre); node 1 > data = 4; noder > duta = 2; rude3 -> dala = 7; node4 > data = 1; node 1 -> rent = rode2; no de 2 - nent = no de 3: node3 -> nent = node4; node4 7 neut = NULL; printf ("original linked list In"); display (nixle1); model = sortlinkedlist (node1); mint ("sorted linked list In"); display (node1); bere (node1). tree (rod 2); tree (roll3); free (node4); returno: Original Linked list: Sorted Linked litt:

4) standard linked with reverted. #include & Stolioin #inducle & Hollib, h> smuct ruled int date: struct Node \* rent: void display (smuct Node \* head) 1: smuct node unrest = hered; while ( current! = NULL) { print (" 1.d It", current - data): ament = airrent - nont; being ( , 10 ,)? much worle \* reverselinked List ( smuch node \* head) struct Node \* prev, \* current, \* nent; Frev = NULL; current=head; while ( current! = NULL){ rent = current -> nent; arrest - nent = prev: mev=current; unvert = aent; return prev;

	Date
	Page
	int main() d
	struct Nale * nodel = (struct Node *) medlor (12eg (struct Nale));
	smut Node * node 2 = (much Node *) mellor (sizeg (smuch Node));
	struct Node * rade 3 = (struct Node*) multic (812eg) (struct Node));
	nodel-7 data : w;
	nodez -> dala = 20;
_	node3 - delta = 30;
_	node 2 -7 neut = node 2 i
_	node2 -7 new = node3;
_	node3 -> nent = NULL;
_	
_	printf("original linked list In");
_	display (rode1);
_	struct Moxle * revtend = reverselinkedlist
	(node1);
	pring ("Reversed linked list 10");
	display (revHead);
_	A Comment Head of the Comment of the
The state of the s	ree(node1);
K	tree (node 2);
MI	y tree (node3);
29114	return o;
	3

original linked list:
10 20 30
Reversed Linked list:

30 20 10

5) standard linked list concatnestion. Hindude ( sectio, h) #include < stdlib.h> smuct Nodel Int data; Struct Node + next; 3 roid display ( smeet rode \* nead ) 2 smuch Node current = head: while ( current!= NULL) { printf(" /d It", current - data); current - ment; prints ("in"); struct Node & .: conclinked lists (struct Node \* Lists, street Noclet lists) { if ( list == NULL) { rehim 1002; smuch Noche \* current = list; while (current + neut!=NULL) current = unrent - rent: current -ment = list 2; return list 1;

	Date Page
1	int meein()1
-	struct Node * list1=(struct
-	
-	wit 1 -> dela=10;
_	eist 1 - nent = NULL;
_	
	inuct Nodet list 2 =
_	List 2 - date = 20
	Wit 2 - nent = WULL;
_	The property of the second
_	mind (" original linked list 1 and 2 10");
	asplay (list1);
	display (list 2);
	me as a second of the second o
	smuct nodex conclinicallist = conclinicallist
	( List1, list2);
	mint (" concutenated Linked list in);
gr	Phlendisplay (conclinicallist);
	2 th
	ree (list1);
	tree Llist 2);
	return o;
	3
	original linked list 1:
	10
	original linked list 2:
	9.0
	concute nated linted hilt:
	10 20
Steel !	

Demonstration of Lect code created and loyed in account. For solving questions we need to consider the parameters given and return the required answer as the platform wents. Then it checks our solutions with some predefined test cases and displays our stand and codes efficiency amound the global ugers,

#### Stack implementation using linked lists:

```
Enter 1. Push
2. Pop
3. -1 to stop
Enter operation:
Stack Underflow
Stack is empty
Enter operation:
Inter the element to push
Stack elements are:
Inter operation:
Inter the element to push
Stack elements are:
Inter operation:
Inter the element to push
Stack elements are:
Inter operation:
Inter the element to push
Stack elements are:
Inter operation:
Inter the selement to push
Stack elements are:
Inter operation:
Inter the element to push
Inter the element to push
Stack elements are:
Inter operation:
Inter opera
```

## Queue implementation using linked lists:

```
■ CAUSers/bmsckDesktop(18M22CS29)\queueUsingLLexe

= The Time I Enqueue

2. Dequeue

3. -1 to stop
Enter operation:

2
Queue Underflow
Queue is empty
Enter operation:

1
Enter the element to enqueue

2
Queue elements are:

5
Queue elements are:

1
Enter operation:

1
Enter the element to enqueue

3
Queue elements are:

5
Queue elements are:

2
3
Enter operation:

1
Enter the element to enqueue

3
Queue elements are:

3
Dequeue elements are:

4
Queue elements are:

5
Queue elements are:

5
Queue elements are:

5
Queue elements are:

7
Pequeue elements are:

5
Queue elements are:

5
Queue elements are:

6
Queue elements are:

7
Pequeue elements are:

8
Pequeue elements are:

9
Pequeue elements are:

1
Execution stopped

Process returned 0 (0x0) execution time: 11.357 s

Press any key to continue.
```

#### linked list sorting:

# Linked lists reversing:

### Linked lists concatenation: