**B ASHISH RAJU**

**1BM19CS030**

**DATA STRUCTURE**

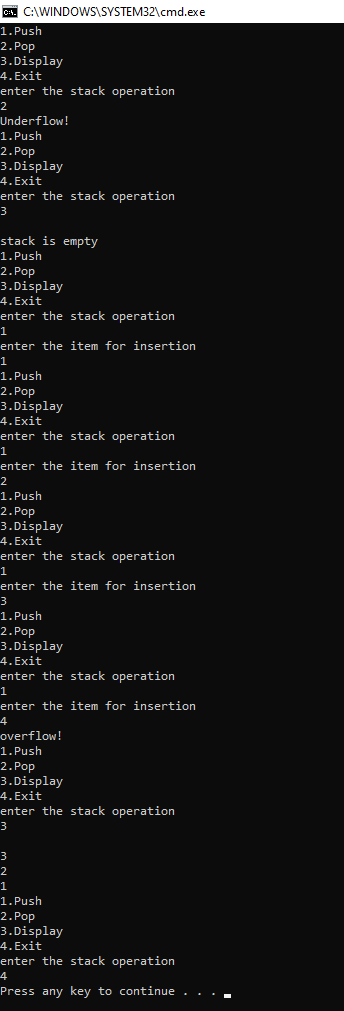
**2nd YEAR**

**LAB 2**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| #include <stdio.h> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| int top=-1; | | | | | | | | | | | | | | | | | | |
| int size=3; | | | | | | | | | | | | | | | | |
| void push(int[],int); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| int pop(int[]); | | | | | | | | | | | | | | | | | | | | | |
| void display(int[]); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| int main() | | | | | | | | | | | | | | | |
| { | |
| int ch,stack[size],item,it; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| do | | |
| { | |
| printf("1.Push\n2.Pop\n3.Display\n4.Exit\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("enter the stack operation\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&ch); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(ch==4) | | | | | | | | | | | | | |
| break; | | | | | |
| switch(ch) | | | | | | | | | | | | | | | | | |
| { | |
| case 1: | | | | | | | | |
| printf("enter the item for insertion\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| push(stack,item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
|  |
| case 2: | | | | | | | | |
| it= pop(stack); | | | | | | | | | | | | | | | | | | | | | | |
| if(it==-1) | | | | | | | | | | | |
| printf("Underflow!\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| else | | | |
| { | |
| printf("deleted element is %d\n\n",it); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| break; | | | | | |
|  |
| case 3: | | | | | | | | |
| printf("\n"); | | | | | | | | | | | | | | | | | | | | |
| display(stack); | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
|  |
| case 4: | | | | | | | | |
| printf("exiting!\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
|  |
| default: | | | | | | | | | |
| printf("wrong choice\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
|  |
| } | |
| }while(ch!=4); | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| return 0; | | | | | | | | | | |
| } | |
| void push(int stack[],int ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| if(top==size-1) | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("overflow!\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| else | | | |
| { | |
| top++; | | | | | | | |
| stack[top]=ele; | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| int pop(int stack[]) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| int ret; | | | | | | |
| if(top==-1) | | | | | | | | | | | | | | | | | | | |
| return -1; | | | | | | | | | | | | |
| else | | | |
| { | |
| ret= stack[top]; | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| top--; | | | | |
| } | |
| return ret; | | | | | | | | | | | | | | |
| } | |
| void display(int stack[]) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| if(top==-1) | | | | | | | | | | | | | | | | | | | |
| printf("stack is empty\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| else | | | |
| { | |
| for(int i=top;i>=0;i--) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("%d\n",stack[i]); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |

}

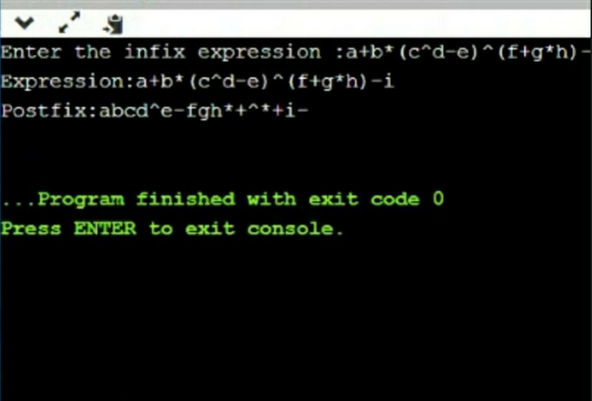
**OUTPUT**

****

**LAB 3**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| #include <stdio.h> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| # define MAX 100 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| char stack[MAX]; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| int top=-1; | | | | | | | | | | | | | | | | | | | | |
|  |
| void push(char ch) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| if (top==MAX-1) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("Stack is full\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| else | | | | | | |
| { | |
| top++; | | | | | | | | | |
| stack[top]=ch; | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | | |
| } | |
| char pop() | | | | | | | | | | | | | | | | | | |
| { | |
| char item; | | | | | | | | | | | | | | | | |
| if (top==-1) | | | | | | | | | | | | | | | | | | | | | | | |
| printf("\n stack is empty !"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| else | | | |
| { | |
| item=stack[top]; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| top--; | | | | | | | |
| return item; | | | | | | | | | | | | | | | | | | | | | |
| } | |
|  |
| } | |
|  |
| int stackempty() | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| if(top==-1) return 1; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| else return 0; | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
|  |
| char stacktop() | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| if( top==-1) | | | | | | | | | | | | | | | | | | | | | | |
| printf("\n stack is empty!"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| else | | | |
| return stack[top]; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| int priority(char ch) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| switch(ch) | | | | | | | | | | | | | | | | | | | |
| { | |
| case '+': | | | | | | | | | | | | | | |
| case '-':return (1); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| case '\*': | | | | | | | | | | | | |
| case '/':return (2); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| case '^': return (3); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| default : return (0); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
|  |
|  |
|  |
| int main(int argc, char \*\*argv) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| char infix[100]; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| int i, item; | | | | | | | | | | | | | | | | | |
| printf("Enter the infix expression :"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%s",infix); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("Expression : %s",infix); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("\n Postfix: "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| i=0; | | | | |
| while (infix[i]!='\0') | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
|  |
|  |
| switch (infix[i]) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| case '(': push(infix[i]); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | | | |
| case ')':while(( item=pop())!='(') | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("%c",item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | | | |
| case '+': | | | | | | | | | | | | | | |
| case '-': | | | | | | | | | | |
| case '\*': | | | | | | | | | | | | |
| case '/': | | | | | | | | | | | | | |
| case '^': | | | | | | | | | | | |
| while(!stackempty() && priority(infix[i])<=priority(stacktop())) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| item=pop(); | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| printf("%c", item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
|  |
| push(infix[i]); | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | | | |
| default : printf("%c", infix[i]); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | | | |
|  |
|  |
| } | |
| i++; | | | | | |
| } | |
|  |
| while(!stackempty()) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| char item; | | | | | | | | | | | | | | | | |
| item=pop(); | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("%c", item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| } | |
| printf("\n"); | | | | | | | | | | | | | | | | | | | | | | | | | |
| return 0; | | | | | | | | | | | | | | | |
|  |

}

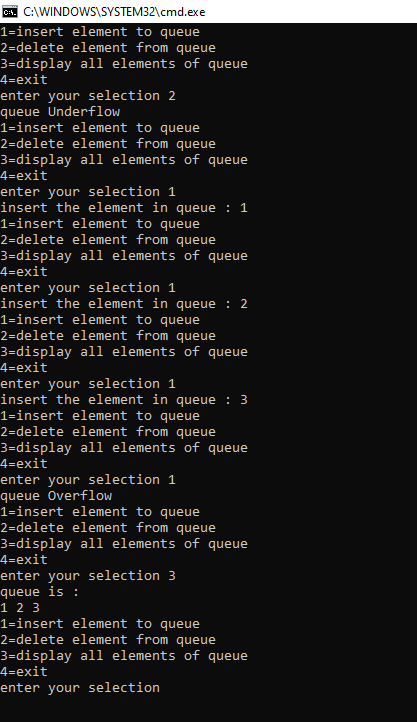
**OUTPUT**

**LAB 4**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| #include <stdio.h> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| #define MAX 3 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| void insert(); | | | | | | | | | | | | | | | | | | | | |
| void delete(); | | | | | | | | | | | | | | | | | | | | | | | |
| void display(); | | | | | | | | | | | | | | | | | | | | | | | | | | |
| int queue\_array[MAX]; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| int rear = - 1; | | | | | | | | | | | | | | | | | | | | | |
| int front = - 1; | | | | | | | | | | | | | | | | | | | | | | | | | |
| main() | | | | | | |
| { | |
| int choice; | | | | | | | | | | | | | | | | |
| while (1) | | | | | | | | | | | | | |
| { | |
| printf("1=insert element to queue \n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("2=delete element from queue \n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("3=display all elements of queue \n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("4=exit \n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("enter your selection "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d", &choice); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| switch (choice) | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| case 1: | | | | | | | |
| insert(); | | | | | | | | | | | |
| break; | | | | | |
| case 2: | | | | | | | |
| delete(); | | | | | | | | | | | | |
| break; | | | | | |
| case 3: | | | | | | | |
| display(); | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 4: | | | | | | | |
| exit(1); | | | | | | | | |
| default: | | | | | | | | | | |
| printf("wrong choice \n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | | |
| } | | |
|  |
| void insert() | | | | | | | | | | | | | | | | | | |
| { | |
| int add\_item; | | | | | | | | | | | | | | | | | | | | | | |
| if (rear == MAX - 1) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("queue Overflow \n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| else | | | |
| { | |
| if (front == - 1) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| /\*If queue is initially empty \*/ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| front = 0; | | | | | | | | | | | | | | |
| printf("insert the element in queue : "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d", &add\_item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rear = rear + 1; | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| queue\_array[rear] = add\_item; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } /\* End of insert() \*/ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| void delete() | | | | | | | | | | | | | | | | | | | |
| { | |
| if (front == - 1 || front > rear) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("queue Underflow \n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| return ; | | | | | | | | | |
| } | |
| else | | | |
| { | |
| printf("element deleted from queue is : %d\n", queue\_array[front]); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| front = front + 1; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } /\* End of delete() \*/ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| void display() | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| int i; | | | | |
| if (front == - 1) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("queue is empty \n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| else | | | |
| { | |
| printf("queue is : \n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| for (i = front; i <= rear; i++) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("%d ", queue\_array[i]); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("\n"); | | | | | | | | | | | | | | | | | |
| } | |

}

**OUTPUT**

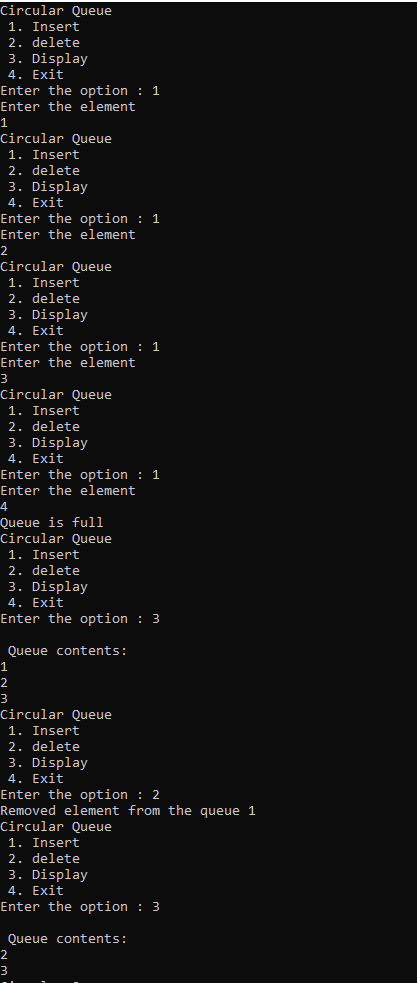
****

**LAB 5**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| #include <stdio.h> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| #include <stdlib.h> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| #define MAX 5 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| int front=-1; | | | | | | | | | | | | | | | | | |
| int rear=-1; | | | | | | | | | | | | | |
|  |
| int queue[MAX]; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| void Enque(int); | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| int Deque(); | | | | | | | | | | | | | | | | |
| void display(); | | | | | | | | | | | | | | | | | | | | | | | | |
| int main(int argc, char \*\*argv) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| int option; | | | | | | | | | | | | |
| int item; | | | | | | | | | |
| do{ | | |
| printf("Circular Queue"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("\n 1. Insert"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("\n 2. delete"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("\n 3. Display"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("\n 4. Exit\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("Enter the option : "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&option); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| switch(option) | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| case 1: printf("Enter the element\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Enque(item); | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | |
| case 2: item=Deque(); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(item==-999) | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("Queue is empty \n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| else | | | |
| printf("Removed element from the queue %d \n",item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | |
| case 3: display(); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | |
| case 4: exit(0); | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } while (option!=4); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| return 0; | | | | | | | | | | |
| } | |
|  |
| void Enque(int ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| if(((front == 0 && rear == MAX - 1))|| (front == rear + 1) ) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("Queue is full\n");return; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| } | |
| else | | | |
| { | |
| rear=(rear+1)%MAX; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| queue[rear]=ele; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(front ==-1) | | | | | | | | | | | | | | | | | | | | |
| front=0; | | | | | | | | |
|  |
|  |
| } | |
| } | |
| int Deque() | | | | | | | | | | | | | | |
| { | |
| int item; | | | | | | | | | |
| if((front == -1)&&(rear == -1)) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
|  |
| return(-999); | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | | |
| { | |
| item=queue[front]; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| if(front==rear) | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| front=-1; | | | | | | | | | | | |
| rear=-1; | | | | | | | |
| } | |
| else | | | |
| { | |
| front=(front+1)%MAX; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| return item; | | | | | | | | | | | | | | | |
| } | |
|  |
| } | |
|  |
| void display() | | | | | | | | | | | | | | | | | | | | | |
| { | |
| int i; | | | | |
| if(((front==-1)&& (rear==-1))|| (front==rear)) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
|  |
| printf("Queue is empty\n");return; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| } | |
| else | | | |
| { | |
| printf("\n Queue contents:\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| for(i=front;i<=rear;i++) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("%d \n", queue[i]); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |

}

**OUTPUT**

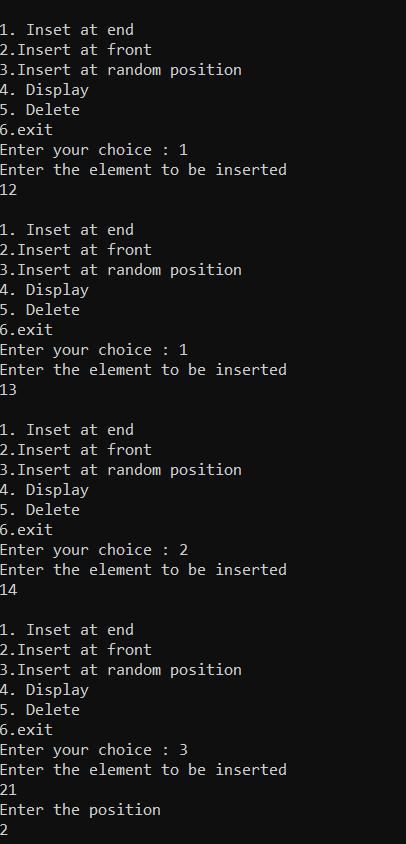
****

**LAB 7**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| #include<stdio.h> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| struct node | | | | | | | | | | | | | | | | | | |
| { | |
| int data; | | | | | | | |
| struct node \*next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| }; | | |
|  |
| struct node \*head=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| int length=0; | | | | | | | | | | | | | | | | | | | | | |
|  |
| void inend(int ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*newnode,\*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->data=ele; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(head==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| head=newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| length=1; | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=head; | | | | | | | | | | | | | | | | | | | |
| while(temp->next!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| temp->next=newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| length++; | | | | | | | | | | | | |
| } | |
|  |
| } | |
|  |
| void infro(int ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp->data=ele; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp->next=head; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| head=temp; | | | | | | | | | | | | | | | | | | | |
| length++; | | | | | | | | | | | | |
| } | |
|  |
| void inran(int ele,int pos) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| if(pos==1) | | | | | | | | | | | | | | | | |
| infro(ele); | | | | | | | | | | | | | | | |
| else if(pos>=length) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| inend(ele); | | | | | | | | | | | | | | | | | |
| else | | | | |
| { | |
| struct node \*inst; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| inst=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| struct node \*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=head; | | | | | | | | | | | | | | | | | | | |
| for(int i=1;i<pos-1;i++) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| inst->data=ele; | | | | | | | | | | | | | | | | | | | | | | | | | |
| inst->next=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp->next=inst; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| length++; | | | | | | | | | | | | |
|  |
| } | |
|  |
| } | |
|  |
| void del(int ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp,\*del; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del=NULL; | | | | | | | | | | | | | | | | | | | | |
| if(head->data==ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| del=head; | | | | | | | | | | | | | |
| head=head->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| temp=head; | | | | | | | | | | | | | | | | | | | |
| while(temp->next!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| if(temp->next->data==ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
|  |
| del=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp->next=del->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| length--; | | | | | | | | | | |
| break; | | | | | |
| } | |
| else | | | | |
| { | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
|  |
| } | |
| } | |
| if(del==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("\nElement not found.\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
|  |
| void display() | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=head; | | | | | | | | | | | | | | | | | | | |
| if(temp==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("\n List is empty \n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| printf("\nThe contents of the list are :\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| while(temp!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("%d\n",temp->data); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
|  |
| } | |
|  |
| int main() | | | | | | | | | | | | | | |
| { | |
| int choice,ele,pos; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| char ch; | | | | | | |
| do | | | |
| { | |
| printf("\n1. Inset at end \n2.Insert at front \n3.Insert in between \n4. Display \n5. Delete \n6.Exit"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("\nEnter your choice : "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&choice); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| switch(choice) | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| case 1: printf("Enter the element to be inserted\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&ele); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| inend(ele); | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 2: printf("Enter the element to be inserted\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&ele); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| infro(ele); | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 3: printf("Enter the element to be inserted\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&ele); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("Enter the position \n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&pos); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| inran(ele,pos); | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 4: display(); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 5: printf("Enter the element to be deleted\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&ele); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del(ele); | | | | | | | | |
| break; | | | | | |
| } | |
| }while(choice!=6); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| return 0; | | | | | | | | | |

}

**OUTPUT**

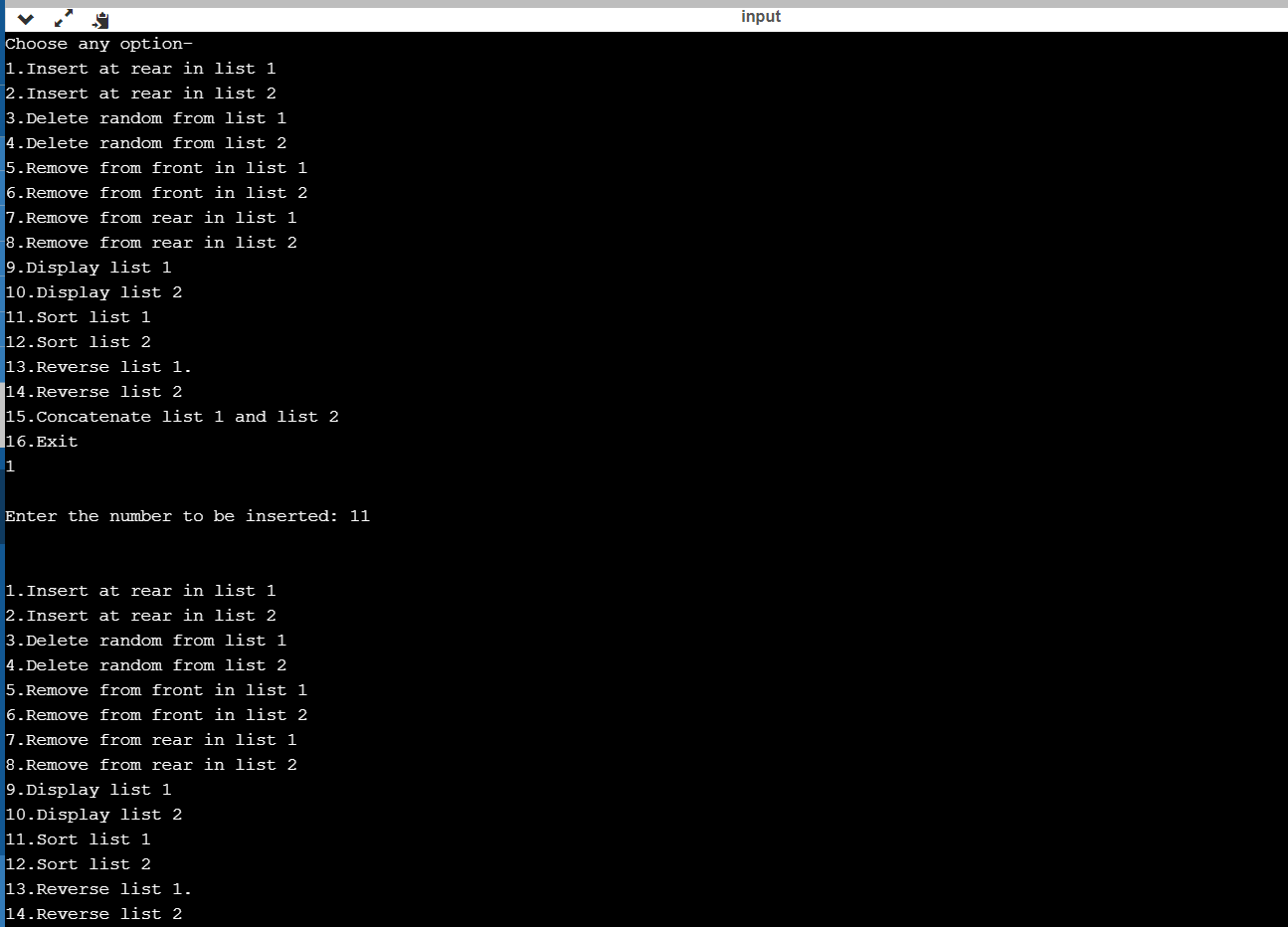
****

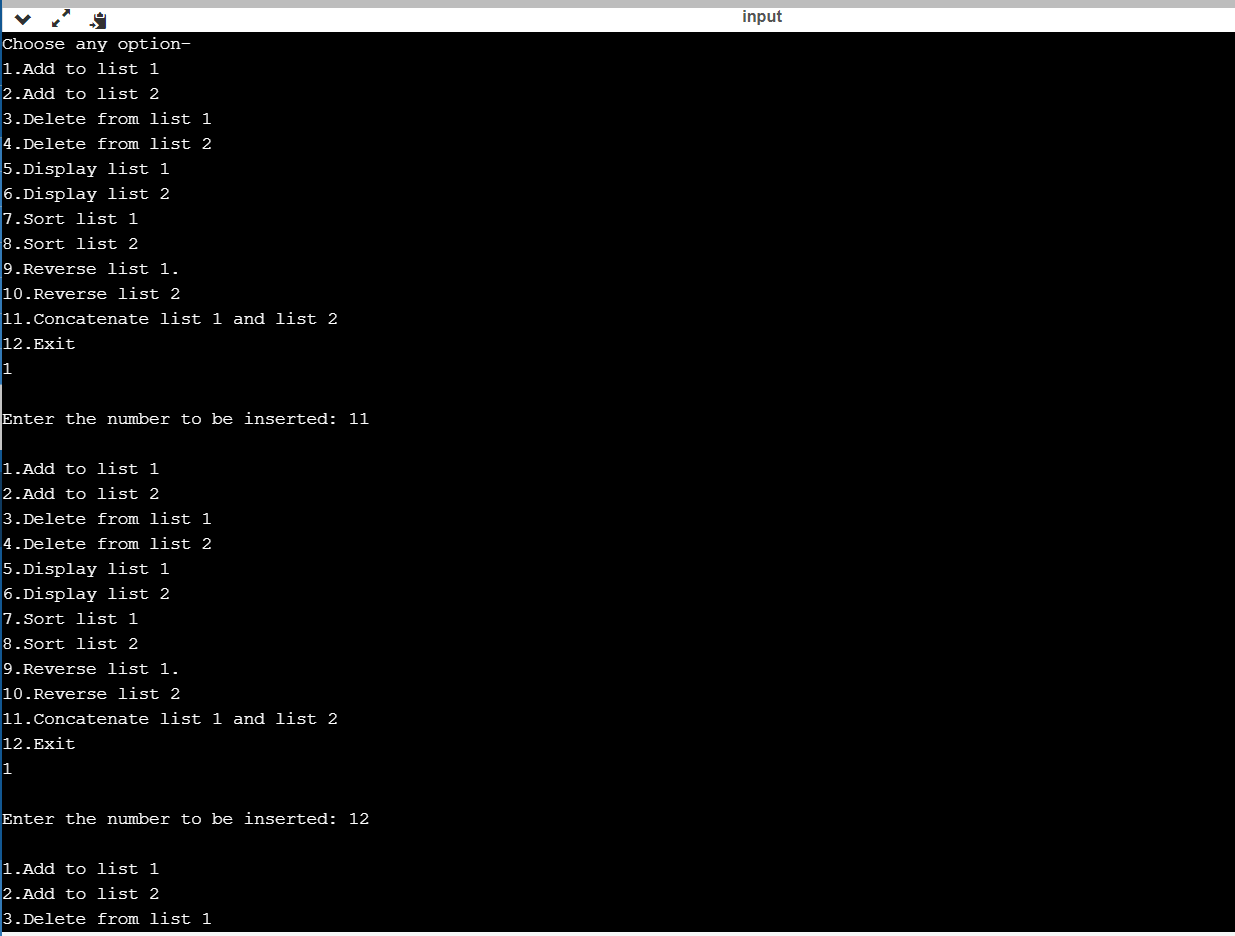
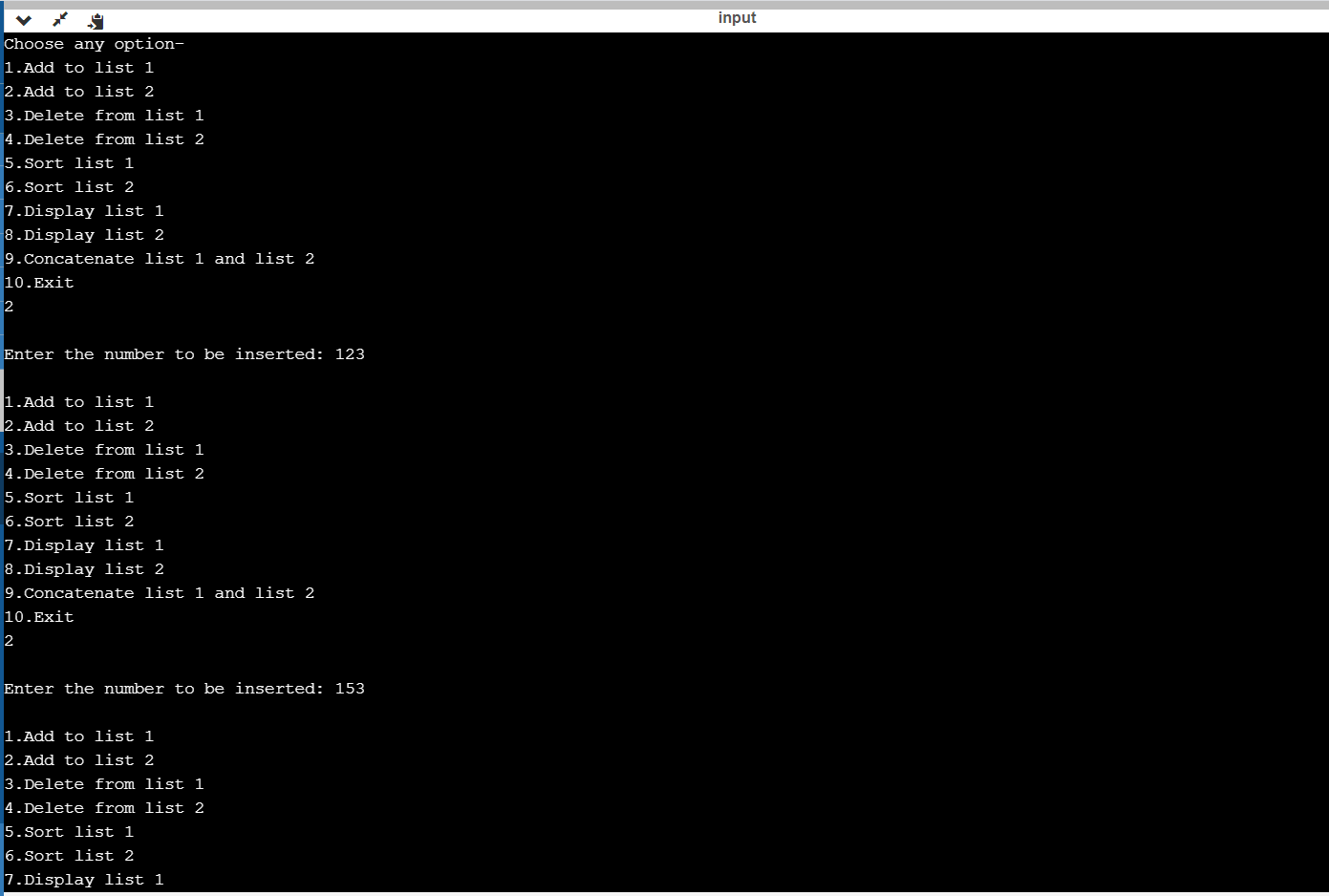
**LAB 8**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| #include<stdio.h> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| #include<stdlib.h> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| struct node | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| int data; | | | | | | | | | | | | |
| struct node \*next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| }; | | |
| struct node \*head1=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| struct node \*head2=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| int length1=0; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| int length2=0; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| void ins\_rear1(int ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*newnode, \*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->data=ele; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(head1==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| head1=newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| length1=1; | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=head1; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| while(temp->next!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| temp->next=newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| length1++; | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void ins\_rear2(int ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*newnode, \*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->data=ele; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(head2==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| head2=newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| length2=1; | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=head2; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| while(temp->next!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| temp->next=newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| length2++; | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void del\_front1() | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(head1==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("List is empty!!\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| temp=head1; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| head1=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("%d is removed front front!!\n",temp->data); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| free(temp); | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void del\_front2() | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(head2==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("List is empty!!\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| temp=head2; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| head2=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("%d is removed front front!!\n",temp->data); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| free(temp); | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void del\_rear1() | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp, \*run; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=(struct node \*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| run=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=head1->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(run==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("List is empty!!\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| while(temp->next!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| run=temp; | | | | | | | | | | | | | | | | | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| printf("%d is deleted from rear\n",temp->data); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| run->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| free(temp); | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void del\_rear2() | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp, \*run; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=(struct node \*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| run=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=head2->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| run=head2; | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(run==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("List is empty!!\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| while(temp->next!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| run=temp; | | | | | | | | | | | | | | | | | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| printf("%d is deleted from rear\n",temp->data); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| run->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| free(temp); | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void del1(int ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp, \*del; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(head1->data==ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| del=head1; | | | | | | | | | | | | | | | | | | | | | | | | |
| head1=head1->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| temp=head1; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| while(temp!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| if(temp->next->data==ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| del=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp->next=del->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| length1--; | | | | | | | | | | | | | | | |
| break; | | | | | |
| } | |
| else | | | | |
| { | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| } | |
| if(del==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("\nElement not found!!\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void del2(int ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp, \*del; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(head2->data==ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| del=head2; | | | | | | | | | | | | | | | | | | | | | | | | |
| head1=head2->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| temp=head2; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| while(temp->next!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| if(temp->next->data==ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| del=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp->next=del->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| length2--; | | | | | | | | | | | | | | | |
| break; | | | | | |
| } | |
| else | | | | |
| { | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| } | |
| if(del==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("\nElement not found!!\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void sort1(struct node \*h) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| int i,j,a; | | | | | | | | | |
|  |
| struct node \*temp1; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| struct node \*temp2; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| for(temp1=h;temp1!=NULL;temp1=temp1->next) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| for(temp2=temp1->next;temp2!=NULL;temp2=temp2->next) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | | |
| if(temp2->data < temp1->data) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| a = temp1->data; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp1->data = temp2->data; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp2->data = a; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| } | |
| } | |
| void sort2(struct node \*h) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| int i,j,a; | | | | | | | | | |
|  |
| struct node \*temp1; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| struct node \*temp2; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| for(temp1=h;temp1!=NULL;temp1=temp1->next) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| for(temp2=temp1->next;temp2!=NULL;temp2=temp2->next) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | | |
| if(temp2->data < temp1->data) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| a = temp1->data; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp1->data = temp2->data; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp2->data = a; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| } | |
| } | |
| void rev1(struct node \*h) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| int i,j,a; | | | | | | | | | |
|  |
| struct node \*temp1; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| struct node \*temp2; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| for(temp1=h;temp1!=NULL;temp1=temp1->next) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| for(temp2=temp1->next;temp2!=NULL;temp2=temp2->next) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| a = temp1->data; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp1->data = temp2->data; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp2->data = a; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| } | |
| void rev2(struct node \*h) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| int i,j,a; | | | | | | | | | |
|  |
| struct node \*temp1; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| struct node \*temp2; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| for(temp1=h;temp1!=NULL;temp1=temp1->next) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| for(temp2=temp1->next;temp2!=NULL;temp2=temp2->next) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| a = temp1->data; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp1->data = temp2->data; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp2->data = a; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| } | |
| void conc() | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp, \*run; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| run=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| run=head1; | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(length1==0 && length2==0) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("\nBoth lists empty\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| temp=head1; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| while(temp->next!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| temp->next=head2; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| printf("\nThe elements in the concatenated list is-\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| while(run!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("%d\n",run->data); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| run=run->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void display1() | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=head1; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(temp==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("\nList is empty!!\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| printf("\nThe contents in list 1 are-\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| while(temp!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("%d\n",temp->data); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| } | |
| void display2() | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=head2; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(temp==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("\nList is empty!!\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| printf("\nThe contents in list 2 are-\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| while(temp!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("%d\n",temp->data); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| } | |
| int main() | | | | | | | | | | | | | | | | |
| { | |
| int ch, item; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("Choose any option-"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| do{ | | | |
| printf("\n1.Insert at rear in list 1\n2.Insert at rear in list 2\n3.Delete random from list 1\n4.Delete random from list 2\n5.Remove from front in list 1\n6.Remove from front in list 2\n7.Remove from rear in list 1\n8.Remove from rear in list 2\n9.Display list 1\n10.Display list 2\n11.Sort list 1\n12.Sort list 2\n13.Reverse list 1.\n14.Reverse list 2\n15.Concatenate list 1 and list 2\n16.Exit\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&ch); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| switch(ch) | | | | | | | | | | | | | | | | | | |
| { | |
| case 1: | | | | | | |
| printf("\nEnter the number to be inserted: "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ins\_rear1(item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 2: | | | | | | |
| printf("\nEnter the number to be inserted: "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ins\_rear2(item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 3: | | | | | | |
| printf("\nEnter the number to be deleted: "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del1(item); | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 4: | | | | | | |
| printf("\nEnter the number to be deleted: "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| del2(item); | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 5: | | | | | | |
| del\_front1(); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 6: | | | | | | |
| del\_front2(); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 7: | | | | | | |
| del\_rear1(); | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 8: | | | | | | |
| del\_rear2(); | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | | |
| case 9: | | | | | | |
| display1(); | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 10: | | | | | | | | | | | | | |
| display2(); | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 11: | | | | | | | | | | | |
| sort1(head1); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 12: | | | | | | | | | | | | | |
| sort2(head2); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 13: | | | | | | | | | | | | | |
| rev1(head1); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 14: | | | | | | | | | | | | | |
| rev2(head2); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | |
| case 15: | | | | | | | | | | | | | |
| conc(); | | | | | | | | |
| break; | | | | | |
| case 16: | | | | | | | | | | | | | |
| break; | | | | | |
| default: | | | | | | | | | | |
| printf("\nEnter a valid choice!!\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| }while(ch!=16); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| return 0; | | | | | | | | | | | | | | |

}

**OUTPUT**

****

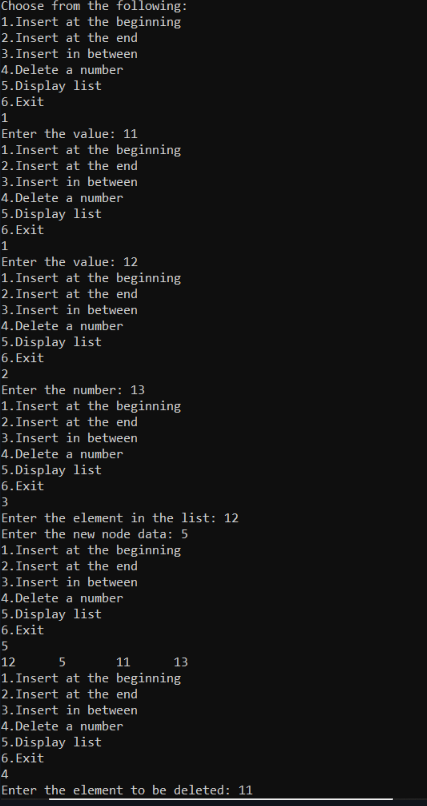
****

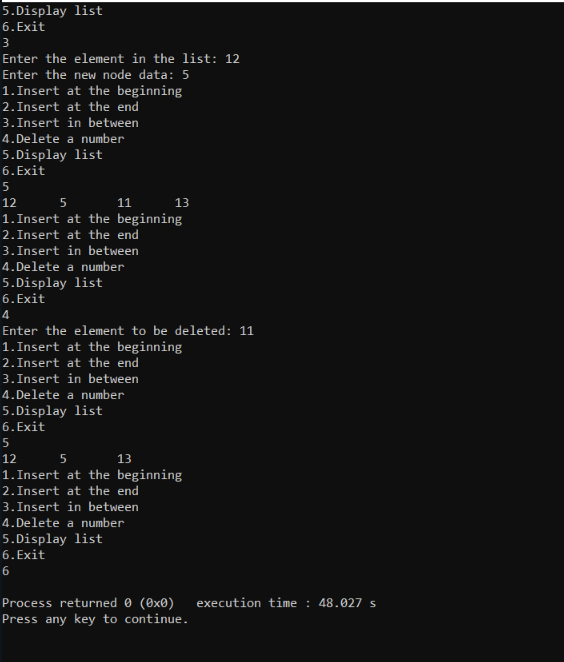
**LAB 9**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| #include<stdio.h> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| #include<stdlib.h> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| struct node | | | | | | | | | | | | | | | | | | | | | |
| { | |
| int data; | | | | | | | | | | | | |
| struct node \*next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| struct node \*prev; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| }; | | |
|  |
| struct node \*head; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| void ins\_left() | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("Enter the value: "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&newnode->data); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->prev=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(head==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| head=newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| newnode->next=head; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| head->prev=newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| head=newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void ins\_end() | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*newnode,\*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("Enter the number: "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&newnode->data); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->prev=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(head==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| head=newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else | | | | |
| { | |
| temp=head; | | | | | | | | | | | | | | | | | | | | | | |
| while(temp->next!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp->next=newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->prev=temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |
| } | |
|  |
| } | |
| void ins\_ran() | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| int ele; | | | | | | | | | |
| struct node \*newnode,\*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("Enter the element in the list: "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&ele); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode=(struct node\*)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("Enter the new node data: "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&newnode->data); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->prev=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(head==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("List is Empty/n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| return; | | | | | | | | |
| } | |
| temp=head; | | | | | | | | | | | | | | | | | | | | | | |
| while(temp->data!=ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(temp==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("Element is not in the list\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| return; | | | | | | | | |
| } | |
| } | |
| newnode->next=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp->next=newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->prev=temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| newnode->next->prev=newnode; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| void del() | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| int ele; | | | | | | | | | |
| if(head==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("List is empty!!\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| return; | | | | | | | | |
| } | |
| printf("Enter the element to be deleted: "); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&ele); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=head; | | | | | | | | | | | | | | | | | | | | | | |
| while(temp->data!=ele) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(temp==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("Element not in list\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | |
| } | |
| } | |
| if(temp==head) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| head=head->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| else if(temp->next==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| temp=temp->prev; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp->next=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
|  |
| else | | | | |
| { | |
| temp->prev->next=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp->next->prev=temp->prev; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void display() | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| struct node \*temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=head; | | | | | | | | | | | | | | | | | | | | | | |
| while(temp!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("%d\t",temp->data); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=temp->next; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| printf("\n"); | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| int main() | | | | | | | | | | | | | | | | | | |
| { | |
| int ch; | | | | | | | |
| printf("Choose from the following: \n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| do{ | | | |
| printf("1.Insert at the beginning\n2.Insert at the end\n3.Insert in between\n4.Delete a number\n5.Display list\n6.Exit\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&ch); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| switch(ch) | | | | | | | | | | | | | | | | | | | | |
| { | |
| case 1: | | | | | | | | | | |
| ins\_left(); | | | | | | | | | | | | | | | | | |
| break; | | | | | | |
| case 2: | | | | | | | | | | |
| ins\_end(); | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | |
| case 3: | | | | | | | | | | |
| ins\_ran(); | | | | | | | | | | | | | | | |
| break; | | | | | | |
| case 4: | | | | | | | | | | |
| del(); | | | | | |
| break; | | | | | | |
| case 5: | | | | | | | | | | |
| display(); | | | | | | | | | | | | | | |
| break; | | | | | | |
| case 6: | | | | | | | | | | |
| break; | | | | | | |
| default: | | | | | | | | | | | |
| printf("Invalid choice!!/n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| }while(ch!=6); | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| return 0; | | | | | | | | | | | | | |

}

**OUTPUT**

****

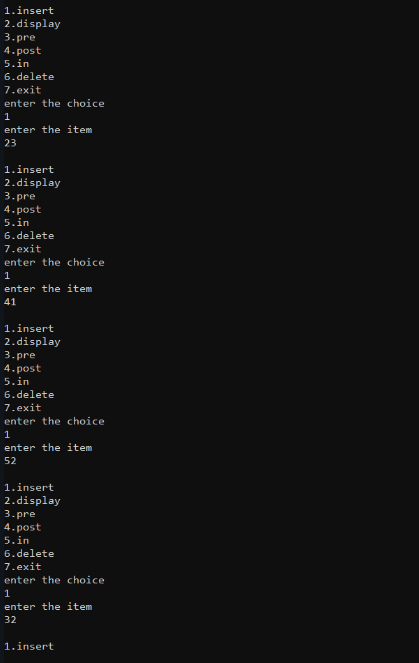
****

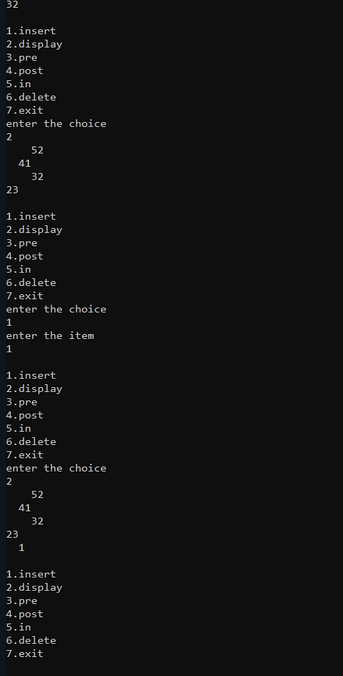
**LAB 10**

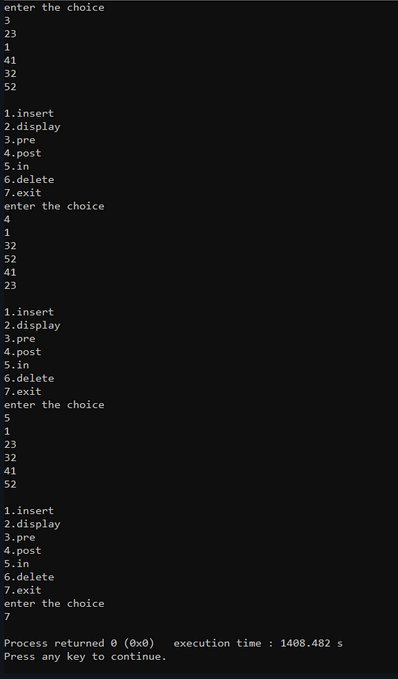
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| #include<stdio.h> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| #include<stdlib.h> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| #include<conio.h> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| struct node | | | | | | | | | | | | | | | |
| { | |
| int info; | | | | | | | | | |
| struct node \*rlink; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| struct node \*llink; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| }; | | |
| typedef struct node \*NODE; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| NODE getnode() | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| NODE x; | | | | | | | | | | | | |
| x=(NODE)malloc(sizeof(struct node)); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(x==NULL) | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("mem full\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| exit(0); | | | | | | | |
| } | |
| return x; | | | | | | | | | | |
| } | |
| void freenode(NODE x) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| free(x); | | | | | | | | |
| } | |
| NODE insert(NODE root,int item) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| NODE temp,cur,prev; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp=getnode(); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp->rlink=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp->llink=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| temp->info=item; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(root==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| return temp; | | | | | | | | | | | | | | | | | | | |
| prev=NULL; | | | | | | | | | | | | | | | | | | | | |
| cur=root; | | | | | | | | | | | |
| while(cur!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| prev=cur; | | | | | | | | | | | | | |
| cur=(item<cur->info)?cur->llink:cur->rlink; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| if(item<prev->info) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| prev->llink=temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| else | | | |
| prev->rlink=temp; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| return root; | | | | | | | | | | | | | | | | |
| } | |
| void display(NODE root,int i) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| int j; | | | | |
| if(root!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| display(root->rlink,i+1); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| for(j=0;j<i;j++) | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf(" "); | | | | | | | | | | | | | | |
| printf("%d\n",root->info); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| display(root->llink,i+1); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| NODE delete(NODE root,int item) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| NODE cur,parent,q,suc; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| if(root==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("empty\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| return root; | | | | | | | | | | | | | | | | |
| } | |
| parent=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| cur=root; | | | | | | | | | | | |
| while(cur!=NULL&&item!=cur->info) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| parent=cur; | | | | | | | | | | | | | | | | | |
| cur=(item<cur->info)?cur->llink:cur->rlink; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| if(cur==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("not found\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| return root; | | | | | | | | | | | | | | | | |
| } | |
| if(cur->llink==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| q=cur->rlink; | | | | | | | | | | | | | | | | | | | | | | |
| else if(cur->rlink==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| q=cur->llink; | | | | | | | | | | | | | | | | | | | | | |
| else | | | |
| { | |
| suc=cur->rlink; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| while(suc->llink!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| suc=suc->llink; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| suc->llink=cur->llink; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| q=cur->rlink; | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| if(parent==NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| return q; | | | | | | | | | | |
| if(cur==parent->llink) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| parent->llink=q; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| else | | | |
| parent->rlink=q; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| freenode(cur); | | | | | | | | | | | | | | | | | | | | | | | | |
| return root; | | | | | | | | | | | | | | | | |
| } | |
|  |
| void preorder(NODE root) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| if(root!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| printf("%d\n",root->info); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| preorder(root->llink); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| preorder(root->rlink); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void postorder(NODE root) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| if(root!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
|  |
| postorder(root->llink); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| postorder(root->rlink); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("%d\n",root->info); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void inorder(NODE root) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| if(root!=NULL) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
|  |
| inorder(root->llink); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("%d\n",root->info); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| inorder(root->rlink); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| } | |
| } | |
| void main() | | | | | | | | | | | | | | | | | | |
| { | |
| int item,choice; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| NODE root=NULL; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| for(;;) | | | | | |
| { | |
| printf("\n1.insert\n2.display\n3.pre\n4.post\n5.in\n6.delete\n7.exit\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| printf("enter the choice\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&choice); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| switch(choice) | | | | | | | | | | | | | | | | | | | | | | | | | |
| { | |
| case 1:printf("enter the item\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| root=insert(root,item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | |
| case 2:display(root,0); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | |
| case 3:preorder(root); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | |
| case 4:postorder(root); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | |
| case 5:inorder(root); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | |
| case 6:printf("enter the item\n"); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| scanf("%d",&item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| root=delete(root,item); | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | |
| default:exit(0); | | | | | | | | | | | | | | | | | | | | | | | | | | |
| break; | | | | | | |
| } | |
| } | |

}

**OUTPUT**

****

****

****