Q. Write a program to implement stacks using linked list:

Program

```
#include <stdio.h>

struct node
{ int data;
  struct node *next;
};
struct node *top = 0;
void push(int val)
{ struct node *newnode;
  newnode = (struct node *)malloc(sizeof(struct node));
  newnode->data = val;
  newnode->next = top;
  top = newnode;
}
void display()
{ struct node *temp;
  temp = top;
  if (top == 0) { printf("stack is empty"); }
  else
  { while (temp != 0)
    { printf("%d\n", temp->data);
      temp = temp->next;
    }
  }
}
void pop()
{ struct node *temp;
  temp = top;
```

```
if(top==0){printf("stack underflow");}
else
{ printf("poped element is %d\n", top->data);
  top=top->next;
  free(temp);
}
}
void main()
{ int choice, num;
  printf("Enter the operation:\n1.push\n2.pop\n
          3.display\n");
  while(1){
          printf("Enter operation\n");
          scanf("%d", &choice);
          if(choice==-1)
          { printf("operation completed.\n");
            break;
          }
  else
  { switch(choice)
    {
    case 1: printf("Enter the number:\n");
            scanf("%d", &num);
            push(num);
            break;
    case 2: pop();
            break;
    case 3: display();
            break;
    default: printf("invalid input");
    }
  }
}
```

Date :
Page No :

Output:

enter the operation:
1. push 2. pop 3. display 4. -1 to stop
enter operation
1
Enter the number: 5
enter operation: 1
Enter the number: 6
enter operation: 1
Enter the number: 7
Enter operation: 3
7
6
5
enter operation 2
poped element is 7.
enter operation 2
poped element is 6.
enter operation 2
poped element is 5
enter operation 2
stack underflow

enter operation -1
operation completed

C:\Users\bmsce\Desktop\22cs300\stackII.exe

```
enter the operation:
1.push
2.pop
3.display
4.-1 to stop
enter operation
1
Enter the number:
5
enter operation
1
Enter the number:
6
enter operation
1
Enter the number:
7
enter operation
2
poped element is 7
enter operation
2
poped element is 6
enter operation
2
poped element is 5
enter operation
2
stack underflow
enter operation
```

b Write a program to implement Queues using linked list.

Program

```
#include <stdio.h>

struct node
{ int data;
  struct node *next;
}
struct node *front=0;
struct node *rear=0;
void enque(int x)
 { struct node *newnode;
   newnode=(struct node *)malloc(sizeof(struct node));
   newnode->data=x;
   newnode->next=0;

   if(front==0 && rear==0)
     { front=rear=newnode; }
   else
     { rear->next=newnode;
       rear=newnode;
     }
}

void display()
{ struct node *temp;
  if(front==0 && rear==0)
    { printf("The Queue is empty\n"); }
  else
    { temp=front;
```

```
        while (temp != 0)
        { printf ("%d", temp->data);
          temp = temp->next;
        }
    }
}
void dequeue()
  { struct node *temp;
    temp = front;
    if (front == 0 && rear == 0)
      { printf ("Queue underflow\n"); }
    else
      { printf ("dequeued element is %d\n",
                front->data);
        front = front->next;
        free (temp);
      }
  }
void main()
{ int choice, num;
  printf("Enter the operation:\n1. enque\n2. deque.\n3. display\n4. -1 to stop\n");
  while(1)
  { printf("Enter operation:\n");
    scanf("%d", &choice);
    if (choice == -1)
    { printf("operation completed\n");
      break;
    }
    else
    { switch (choice)
      {
```

```
        case 1: printf("Enter the numbers:\n");
                scanf("%d", &num);
                enqueue(num);
                break;
        case 2: dequeue();
                break;
        case 3: display();
                break;
        default: printf("invalid input\n");
    }
  }
 }
}
```

29/1/24

//output:

enter the operation

1. enque 2. dequeue 3. display 4. -1 to stop

enter operation 1

enter the number 5

enter operation 1

enter the number 6

enter operation 1

enter the number 7

enter operation 3

5

6

7

enter operation 2

dequeued element is 5

enter operation 2

dequeued elemt is 6

1a. write

linked

Demonstrate

Lecture

29/1/24

C:\Users\bmsce\Desktop\22cs300\queuell.exe

```
enter the operation:
1.enqueue
2.dequeue
3.display
4.-1 to stop
enter operation
1
Enter the number:
5
enter operation
1
Enter the number:
6
enter operation
1
Enter the number:
7
enter operation
3
5
6
7
enter operation
2
deueued element is 5
enter operation
2
deueued element is 6
enter operation
2
deueued element is 7
enter operation
2
queue underflow
```

1a. Write a program to sort and reverse a single linked list.

Demonstration
Lecture
N
22/11/24

1. Write a program to sort, reverse and concatenation of single linked list.

```
#include <stdio.h>

struct node *head=0, *newnode, *temp;
struct node {
        int data;
        struct node *next;
};
struct node *head=0, *newnode, *temp;

void create()
{ int i, n;
  printf("Enter the no: of elements: \n");
  scanf("%d", &n);

  for(i=0; i<n; i++)
  {
    newnode = (struct node *) malloc(sizeof(struct node));
    printf("Enter the %d element: \n", i+1);
    scanf("%d", &newnode->data);
    newnode->next = 0;
```

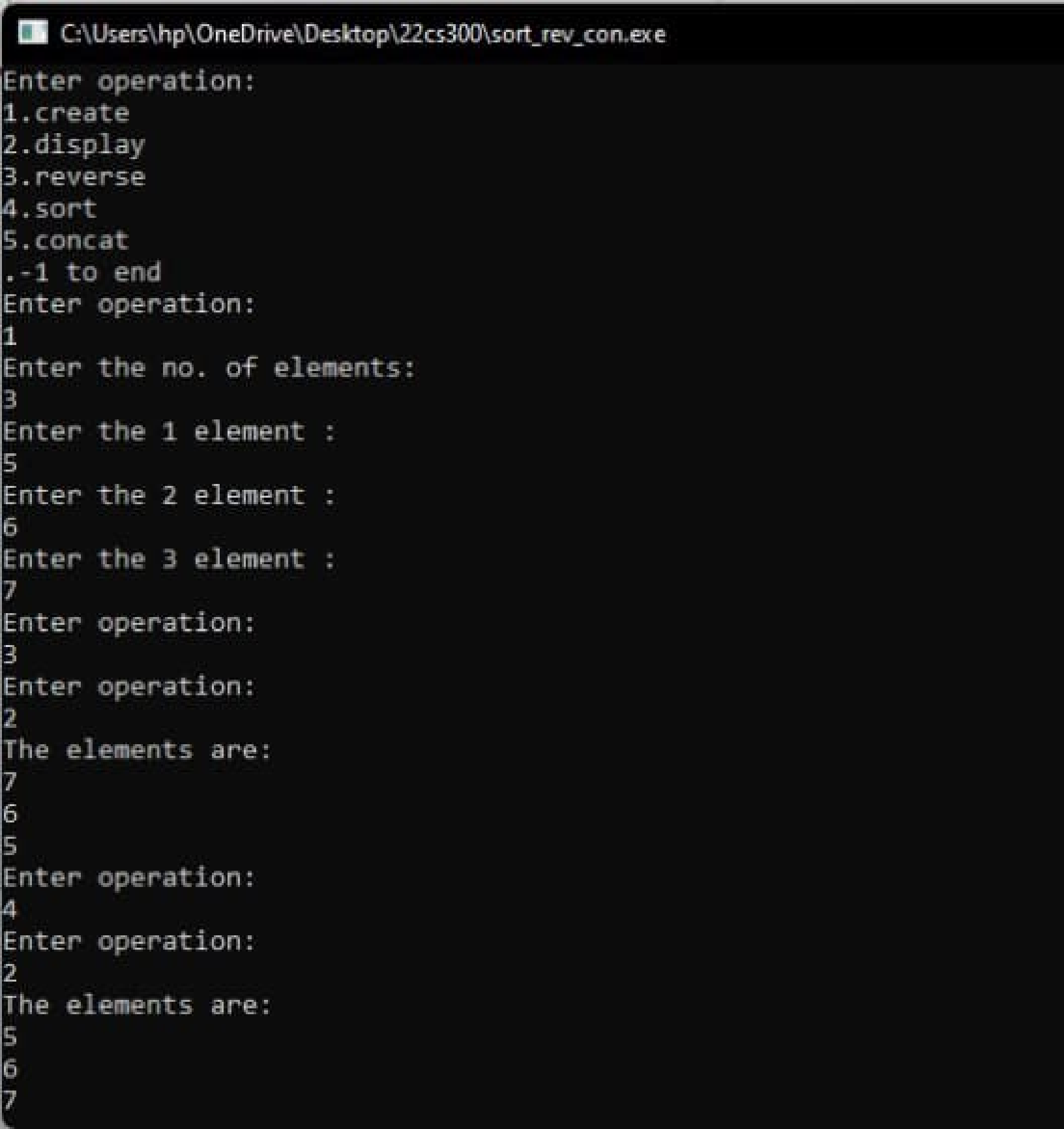
```
    if(head==0)
     { temp=head=newnode; }
     else
       { temp->next=newnode;
         temp=newnode;
       }
   }
}
void display()
{ temp=head;
  printf("The elements are:\n");
  while(temp!=0)
  { printf("%d\n",temp->data);
    temp=temp->next;
  }
}
void reverse()
{ struct node *prenode=0, *currnode=head, *nextnode;
  while(nextnode!=0)
  { nextnode=nextnode->next;
    currnode->next=prenode;
    prenode=currnode;
    currnode=nextnode;
  }
  head=prenode;
}

void sort()
{ int swapped=0
  struct node *end=0;
  if(head==0 || head->next==0){
    printf("already sorted");
  }
```

```
do{ temp=head;
    while(temp->next != end)
    {
    if(temp->data > temp->next->data)
    {
      int t = temp->data;
      temp->data = temp->next->data;
      temp->next->data = t;
      swapped = 1;
    }
     temp = temp->next;
   }
    end = temp;
  } while(swapped);
}

void main()
{ int choice;
  printf("Enter operation: \n1.create \n2.display
        \n3.reverse \n4.sort \n5.concat \n6.-1to exit");
  while(1)
  { printf("Enter operation \n");
    scanf("%d", &choice);

    if(choice == -1)
    { printf("operation completed! \n");
      break;
    }
    else
    {switch(choice)
     { case 1: create();
               break;
```

```
        case 2: display();
                break;
        case 3: reverse();
                break;
        case 4: sort();
                break;
        case 5: printf("Enter first list\n");
                create();
                printf("enter second list\n");
                create();
                break;
        default: printf("invalid input\n");
    }
  }
 }
}
```

//output:

```
Enter operation:
1. create
2. display
3. reverse
4. sort
5. concat
6. -1 to end
Enter the operation: 1
Enter the no. of elements: 3
Enter the 1 element: 5
Enter the 2 element 6
Enter the 3 elmt: 7
Enter operation: 3
Enter operation: 2
The elements are: 7
                  6
                  5
```


C:\Users\hp\OneDrive\Desktop\22cs300\sort_rev_con.exe
Enter operation:
1.create
2.display
3.reverse
4.sort
5.concat
.-1 to end
Enter operation:
1
Enter the no. of elements:
3
Enter the 1 element :
5
Enter the 2 element :
6
Enter the 3 element :
7
Enter operation:
3
Enter operation:
2
The elements are:
7
6
5
Enter operation:
4
Enter operation:
2
The elements are:
5
6
7