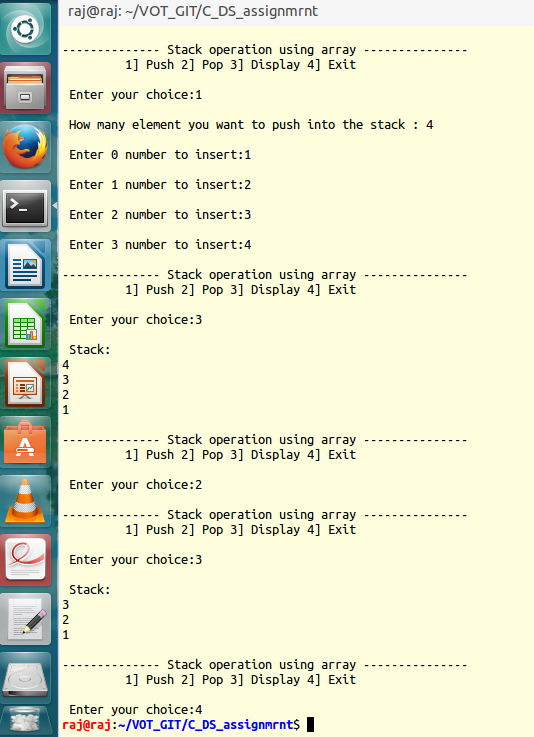
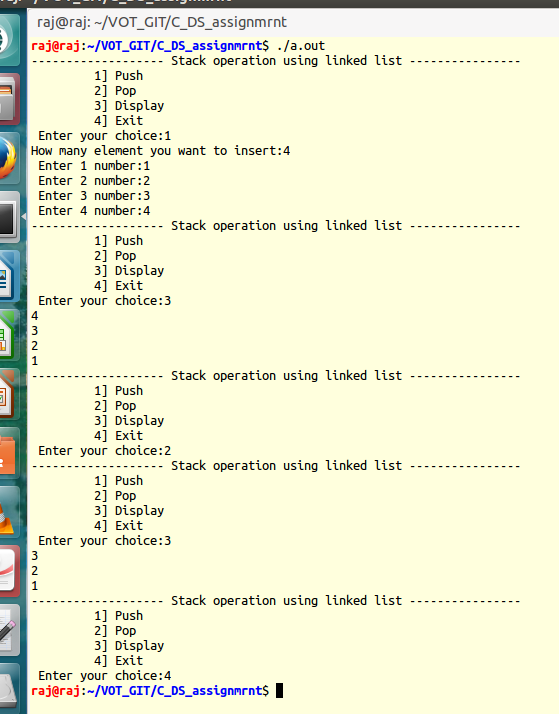
|  |
| --- |
| **Q1./\* function to sort a given singly linked list and to return the sorted list to the caller. Add**  **main program to test the function \*/**  #include<stdio.h>  struct emp  {  int eno;  struct emp \*link;  }\*start;  void create(int data)  {  struct emp \*temp,\*q;  temp=(struct emp\*)malloc(sizeof(struct emp));  temp->eno=data;  temp->link=NULL;  if(start==NULL)  start=temp;  else  {  q=start;  while(q->link != NULL)  {  q=q->link;  }    q->link=temp;  }  }  void display()  {  struct emp \*temp;  temp=start;  if(temp==NULL)  {  printf("\n List is empty \n");  exit(0);  }  else  {  printf("\n The data in the list:\n");  while(temp)  {  printf("|%d|-->",temp->eno);  temp=temp->link;  }  }  }    struct emp \*sort\_asending(struct emp \*start)  {  struct emp \*current=start, \*largest=start, \*largest\_prev=start,\*prev,\*temp;  if(start == NULL || start->link == NULL)  return;    while(current !=NULL )  {  if(current->eno > largest->eno)  {  largest\_prev=prev;  largest=current;  }    prev=current;  current=current->link;  }    if(largest != start)  {  largest\_prev->link=start;  temp=start->link;  start->link=largest->link;  largest->link=temp;  }  }    int main()  {  int data, number,choice,i;  while(1)  {  printf("\n 1. Create \n 2. Display \n 3. Sorting \n 4. Exit \n");  printf("\n Enter your choice:");  scanf("%d",&choice);  switch(choice)  {    case 1: printf("\n How many nodes you want:");  scanf("%d",&number);  for(i=0;i<number;i++)  {  printf(" %d data =",i);  scanf("%d",&data);  create(data);  }  break;  case 2: display();  break;  case 3: start=sort\_asending(start);  break;  case 4: exit(0);  default: printf("\n Enter vvalid choice \n");    }  }  } |



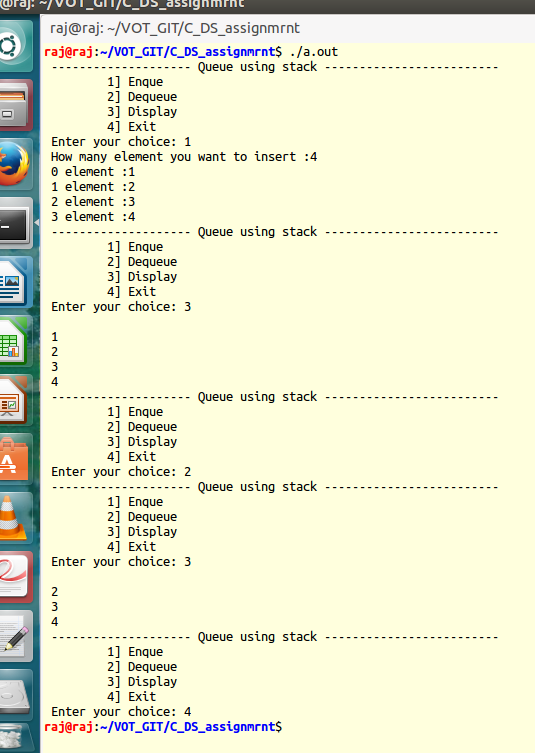
|  |
| --- |
| **Q2\_a. /\* Program to perform stack operations (push, pop, display) using array \*/**  #include<stdio.h>  #include<stdlib.h>  #define MAX\_SIZE 100  void push(int);  void pop();  void isempty();  void display();  int array[MAX\_SIZE],top=-1;  int main()  {  int choice,size,number,count,i;  while(1)  {  printf("\n-------------- Stack operation using array ---------------\n");  printf("\t 1] Push 2] Pop 3] Display 4] Exit \n");  printf("\n Enter your choice:");  scanf("%d",&choice);  switch(choice)  {  case 1: printf("\n How many element you want to push into the stack : ");  scanf("%d",&count);  for(i=0;i<count;i++)  {  printf("\n Enter %d number to insert:",i);  scanf("%d",&number);  push(number);  }  break;  case 2: pop();  break;  case 3: display();  break;  case 4:  exit(0);    default:  printf("\n Please enter the valid choice \n");  }  }  return 0;  }  void push(int x)  {  if(top==MAX\_SIZE - 1)  {  printf("\n Stack overflow \n");    }  array[++top]=x;  }  void pop()  {  if(top==-1)  {  printf("\n Error: No element to pop\n");  }  top--;  }  void display()  {  int i;  printf("\n Stack:\n");  for(i=top;i>=0;i--)  {  printf("%d",array[i]);  printf("\n");  }  } |



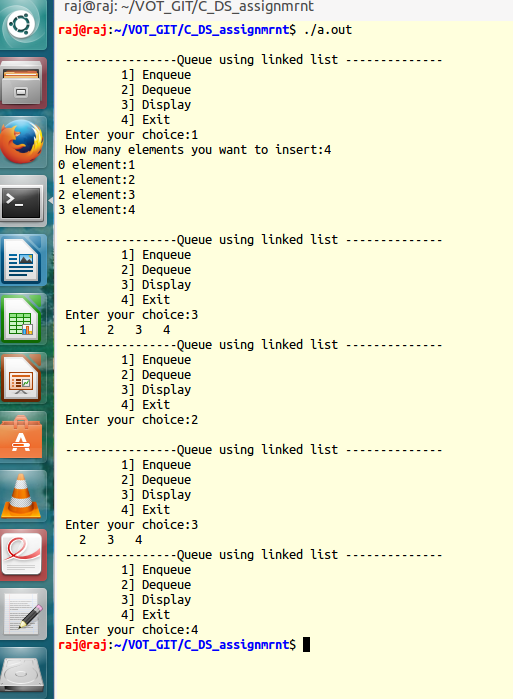
|  |
| --- |
| **Q2\_b. /\* Program to perform stack operations (push, pop, display) using linked list \*/**  #include<stdio.h>  #include<stdlib.h>  #include<unistd.h>  struct node  {  int data;  struct node \*link;  };  struct node \*top=NULL;  void push(int x)  {  struct node \*temp=(struct node \*)malloc(sizeof(struct node\*));  temp->data=x;  temp->link=top;  top=temp;  }  void pop()  {  struct node \*temp;  if(top==NULL)  {  printf(" Stack is empty\n");  return;  }  temp=top;  top=top->link;  free(temp);  }  void display()  {  struct node \*temp=top;    if (temp==NULL)  {  printf(" List is empty \n");  return;  }    else  {  while(temp)  {  printf("%d \n",temp->data);  temp=temp->link;  }    }  }  int main()  {  int choice,number,count,i;  while(1)  {  printf("------------------- Stack operation using linked list ----------------\n");  printf(" \t 1] Push\n\t 2] Pop\n\t 3] Display\n\t 4] Exit\n");  printf(" Enter your choice:");  scanf("%d",&choice);  switch(choice)  {    case 1: printf("How many element you want to insert:");  scanf("%d",&count);  for(i=1;i<=count;i++)  {  printf(" Enter %d number:",i);  scanf("%d",&number);  push(number);  }  break;  case 2: pop();  break;  case 3: display();  break;  case 4: exit(0);  default: printf(" Please enter correct choice \n");  }  }  } |



|  |
| --- |
| **/\*Q3\_a program to perform queue operations (insert, delete, display) using arrays \*/**  #include<stdio.h>  #include<stdlib.h>  #include<unistd.h>  #define MAX 20  int array[MAX],front=0,rear=0;  void enqueue(int x)  {  if(rear==MAX)  {  printf("Overflow \n");  }  else  {  array[rear++]=x;  }  }  void dequeue()  {  if(rear==front)  {  printf(" Underflow \n");  }  else  {  front++;  }  }  void display()  {  int i;  if(front==rear)  {  printf("Queue is empty \n");  }  else  {  printf("\n");  for(i=front;i<rear;i++)  {  printf(" %d \n",array[i]);  }  }    }  int main()  {  int choice, i, count, number;  while(1)  {  printf(" -------------------- Queue using stack -------------------------\n");  printf("\t 1] Enque\n\t 2] Dequeue\n\t 3] Display\n\t 4] Exit \n");  printf(" Enter your choice: ");  scanf("%d",&choice);  switch(choice)  {    case 1: printf(" How many element you want to insert :");  scanf("%d",&count);  for(i=0;i<count;i++)  {  printf(" %d element :",i);  scanf("%d",&number);  enqueue(number);  }  break;  case 2: dequeue();  break;  case 3: display();  break;  case 4:  exit(0);  default:printf("Please enter proper option \n");    }  }  } |



|  |
| --- |
| **/\* Q3\_b Program to perform queue operations (insert, delete, display) using linked list \*/**  #include<stdio.h>  #include<stdlib.h>  #include<unistd.h>  struct node  {  int data;  struct node \*next;  };  struct node \*front=NULL;  struct node \*rear=NULL;  void enqueue(int x)  {  struct node \*temp=(struct node\*)malloc(sizeof(struct node\*));  temp->data=x;  temp->next=NULL;  if(front==NULL && rear==NULL)  {  front=rear=temp;  }  else  {  rear->next=temp;  rear=temp;  }  }  void dequeue()  {  struct node \*temp=front;  if(front==NULL)  printf(" Queue is empty\n");  if(front==rear)  {  front=rear=NULL;  }  else  {  front=front->next;  }  free(temp);  }  void display()  {  struct node \*temp=front;  if(temp==NULL)  printf(" Queue is empty \n");  else  {  while(temp)  {  printf(" %3d",temp->data);  temp=temp->next;  }  }  }  int main()  {  int choice,count,number,i;  while(1)  {  printf("\n ----------------Queue using linked list -------------- \n");  printf("\t 1] Enqueue\n\t 2] Dequeue\n\t 3] Display\n\t 4] Exit");  printf("\n Enter your choice:");  scanf("%d",&choice);  switch(choice)  {  case 1: printf(" How many elements you want to insert:");  scanf("%d",&count);  for(i=0;i<count;i++)  {  printf("%d element:",i);  scanf("%d",&number);  enqueue(number);  }  break;  case 2: dequeue();  break;  case 3: display();  break;  case 4:  exit(0);  default:printf(" Please enter correct choice !!!!\n");  }  }  } |



|  |
| --- |
| **Q4. /\* Write a function to delete middle node for given singly linked list and to send the deleted node list (through call by reference) to the caller. Add main program to test the function. \*/**  #include<stdio.h>  #include<stdlib.h>  struct emp  {  int eno;  struct emp \*link;  }\*start;  void create(int data)  {  struct emp \*temp,\*q;  temp=(struct emp\*)malloc(sizeof(struct emp));  temp->eno=data;  if(start==NULL)  start=temp;  else  {  q=start;    while(q->link != NULL)  {  q=q->link;  }    q->link=temp;  }  }  void display()  {  struct emp \*temp;  temp=start;  if(temp == NULL)  {  printf("List is empty \n");  return ;  }  else  {  printf("The data in list: \n");  while(temp)  {  printf("|%d|-->",temp->eno);  temp=temp->link;  }  }  }  int count()  {  struct emp \*q=start;  int cnt=0;  while(q!=NULL)  {  q=q->link;  cnt++;  }  printf("Total number of elements are: %d \n",cnt);  return cnt;  }  void delete\_middle\_node(int total)  {  int pos,i,data;  struct emp \*temp=start,\*q;  pos=total/2;    for(i=0;i<pos-1;i++)  {  temp=temp->link;  }  data=temp->eno;    q=temp;    while(q)  {  if(q->eno==data)  {  temp=q->link;  q->link=temp->link;  free(temp);  }  q=q->link;  }  }  int main()  {  int i,number,data,total,choice;  while(1)  {  printf("\n 1. Create \n 2. Display \n 3. Delete Middle node \n 4. Exit \n");  printf(" Enter your choice:");  scanf("%d",&choice);  switch(choice)  {  case 1: printf(" How many node you want:");  scanf("%d",&number);  for(i=0;i<number;i++)  {  printf(" %d data :",i+1);  scanf("%d",&data);  create(data);  }  break;  case 2: display();  break;  case 3:  total=count();  delete\_middle\_node(total);  break;  case 4:exit(0);  default:printf(" Please enter correct choice !!!!\n");  }  }  } |



|  |
| --- |
| **Q5. /\* function to sort a given singly linked list and to return the sorted list to the caller. Add**  **main program to test the function \*/**  #include<stdio.h>  struct emp  {  int eno;  struct emp \*link;  }\*start;  void create(int data)  {  struct emp \*temp,\*q;  temp=(struct emp\*)malloc(sizeof(struct emp));  temp->eno=data;  temp->link=NULL;  if(start==NULL)  start=temp;  else  {  q=start;  while(q->link != NULL)  {  q=q->link;  }    q->link=temp;  }  }  void display()  {  struct emp \*temp;  temp=start;  if(temp==NULL)  {  printf(" List is empty \n");  exit(0);  }  else  {  printf(" The data in the list:\n");  while(temp)  {  printf("|%d|-->",temp->eno);  temp=temp->link;  }  }  }  add\_at\_begining(int data)  {  struct emp \*temp;  temp=(struct emp \*)malloc(sizeof(struct emp));  temp->eno=data;  temp->link=start;  start=temp;  }    delete\_node(int data)  {  struct emp \*q,\*temp;  if(start->eno==data)  {  temp=start;  start=start->link;  free(temp);  return;  }  q=start;  while(q->link->link!=NULL)  {  if(q->link->eno==data)  {  temp=q->link;  q->link=temp->link;  free(temp);  return;  }  q=q->link;  }  if(q->link->eno==data)  {  temp=q->link;  free(temp);  q->link=NULL;  return;  }  printf(" Element %d not found\n",data);  }  int main()  {  int data, number,choice,i;  while(1)  {  printf(" \n1. Create \n 2. Display \n 3. add node at begining \n 4. Delete a specified node \n 5. exit \n");  printf(" Enter your choice:");  scanf("%d",&choice);  switch(choice)  {  case 1: printf(" How many nodes you want:");  scanf("%d",&number);  for(i=0;i<number;i++)  {  printf(" %d data =",i);  scanf("%d",&data);  create(data);  }  break;  case 2: display();  break;  case 3: printf(" Enter a data to add node at begining:");  scanf("%d",&data);  add\_at\_begining(data);  break;  case 4: printf(" Enter a data to delete a specified node:");  scanf("%d",&data);  delete\_node(data);  break;  case 5: exit(0);  default: printf(" Enter vvalid choice \n");  }  }  } |

