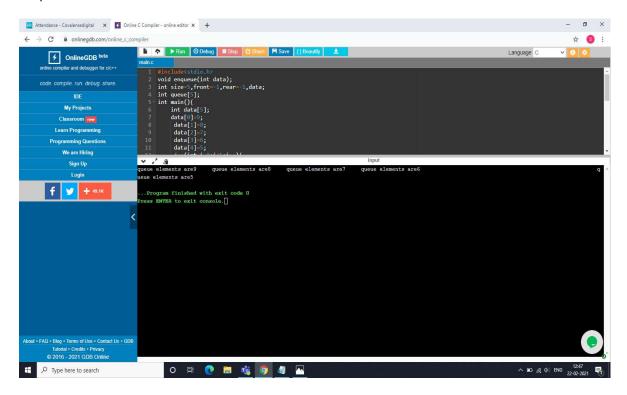
```
3)//queue
#include<stdio.h>
void enqueue(int data);
int size=5,front=-1,rear=-1,data;
int queue[5];
int main(){
  int data[5];
  data[0]=9;
  data[1]=8;
   data[2]=7;
  data[3]=6;
   data[4]=5;
  for(int i=0;i<5;i++){
     enqueue(data[i]);
     printf("queue elements are%d\t",queue[i]);
  return 0;
}
  void enqueue(int data){
    if(rear==size-1){
      printf("queue is full\n");
    }
    else{
      front=0;
      rear=rear+1;
```

```
queue[rear]=data;
}
```

output:-



4)//linked stack push and print function
#include<stdio.h>
#include<stdlib.h>
void push ();

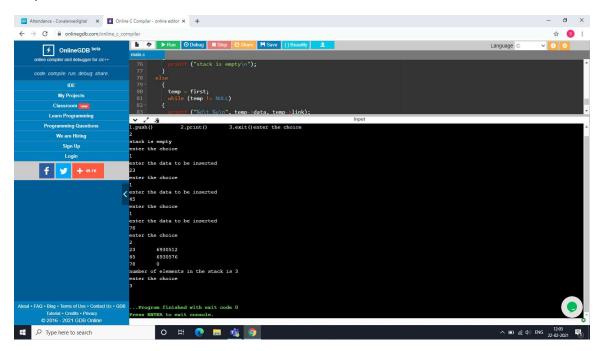
```
void print ();
struct node
 int data;
 struct node *link;
};
typedef struct node ls;
Is *first = NULL, *temp, *last = NULL;
int data, count;
int
main ()
{
 int choice;
 printf ("1.push()\t 2.print()\t 3.exit()");
 while (choice < 4)
  {
   printf ("enter the choice\n");
   scanf ("%d", &choice);
   switch (choice)
        {
        case 1:
         push ();
         break;
        case 2:
         print ();
         break;
        case 3:
         exit (0);
```

```
break;
        default:
         printf ("invalid choice\n");
         break;
        }
  }
 return 0;
}
void
push ()
 if (first == NULL)
  {
   first = (Is *) malloc (sizeof (Is));
   printf ("enter the data to be inserted\n");
   scanf ("%d", &data);
   first->data = data;
   first->link = NULL;
   ++count;
  }
 else
  {
   temp = (Is *) malloc (sizeof (Is));
   temp = first;
   while (temp->link!=NULL)
        {
         temp = temp->link;
```

```
}
   last = (ls*) malloc (sizeof (ls));
   printf ("enter the data to be inserted\n");
   scanf ("%d", &data);
   last->data = data;
   temp->link = last;
   last->link = NULL;
   ++count;
  }
}
void print()
{
 if (first == NULL)
  {
   printf ("stack is empty\n");
  }
 else
  {
   temp = first;
   while (temp != NULL)
        {
         printf ("%d\t %u\n", temp->data, temp->link);
         temp = temp->link;
        }
        printf("number of elements in the stack is %d\n",count);
```

```
}
```

output:-



5)&6)//linked list insertend and insert at position functions

```
#include<stdio.h>
#include<stdlib.h>
void insertend();
void insertpos();
void print ();
struct node
{
  int data;
  struct node *link;
};
```

```
typedef struct node II;
II *first = NULL, *temp, *last = NULL;
int data,pos,size=0,i;
int
main ()
{
 int choice;
 printf ("1.insertend()\t 2.insertpos\t 3.print()\t 4.exit()");
 while (choice < 4)
  {
   printf ("enter the choice\n");
   scanf ("%d", &choice);
   switch (choice)
        {
        case 1:
         insertend ();
         break;
        case 2:
         insertpos ();
         break;
        case 3:
         print ();
         break;
         case 4:exit(0);
         break;
        default:
         printf ("invalid choice\n");
         break;
```

```
}
  }
 return 0;
}
void insertbeg(){
  if(first==NULL){
  first=(II*)malloc(sizeof(II));
    printf("enter the data to be inserted\n");
    scanf("%d",&data);
    first->data=data;
    first->link=NULL;
    ++size;
  }
  else{
    temp=(II*)malloc(sizeof(II));
    printf("enter the data to be inserted\n");
    scanf("%d",&data);
    temp->data=data;
    temp->link=first;
    first=temp;
  }
}
void
insertend ()
{
```

```
if (first == NULL)
   first = (II *) malloc (sizeof (II));
   printf ("enter the data to be inserted\n");
   scanf ("%d", &data);
   first->data = data;
   first->link = NULL;
   ++size;
  }
 else
   temp = (II *) malloc (sizeof (II));
   temp = first;
   while (temp->link!=NULL)
        {
         temp = temp->link;
        }
   last = (II*) malloc (sizeof (II));
   printf ("enter the data to be inserted\n");
   scanf ("%d", &data);
   last->data = data;
   temp->link = last;
   last->link = NULL;
   ++size;
  }
}
```

void insertpos(){

```
Il *temp1;
  printf("enter the position to insert the data\n");
  scanf("%d",&pos);
  if(pos==0){
   insertbeg();
  }
  else{
    if(pos>size){
      printf("out of bound\n");
    }
    else{
    temp=first;
    for(i=1;i<pos;i++){
      temp=temp->link;
    }
    temp1=(II*) malloc(sizeof(II));
    printf("enter the data to be inserted");
    scanf("%d",&data);
    temp1->link=temp->link;
    temp->link=temp1;
    temp1->data=data;
    ++size;
  }
  }
}
```

```
void print()
{
 if (first == NULL)
  {
   printf ("linked list is empty\n");
  }
 else
  {
   temp = first;
   while (temp != NULL)
       {
         printf ("%d\t %u\n", temp->data, temp->link);
         temp = temp->link;
       }
 }
}
output:-
```

