**Programs**

1. Write a program to implement stack using array.

Ans:

#include<stdio.h>

#include<conio.h>

#define max 5

int top=-1,stack[max];

void push();

void pop();

void display();

void main()

{

int ch;

clrscr();

while(1)

{

printf("\n------------------------Menu------------------------------");

printf("\n1.Push \t 2.Pop \t 3.Display\t4.Exit");

printf("\n----------------------------------------------------------");

printf("\nEnter your choice:");

scanf("%d",&ch);

switch(ch)

{

case 1:push();

break;

case 2:pop();

break;

case 3:display();

break;

case 4:exit(0);

default:

printf("Enter valid choice....");

}

}

}

void push()

{

int val;

if(top==max-1)

printf("Stack is full...");

else

{

printf("Enter element to push:");

scanf("%d",&val);

top=top+1;

stack[top]=val;

printf("Element inserted successfully...");

}

}

void pop()

{

if(top==-1)

printf("Stack is empty...");

else

{

top=top-1;

printf("Element deleted successfully...");

}

}

void display()

{

int i;

if(top==-1)

printf("Stack is empty...");

else

{

printf("Elements in stack are:\n");

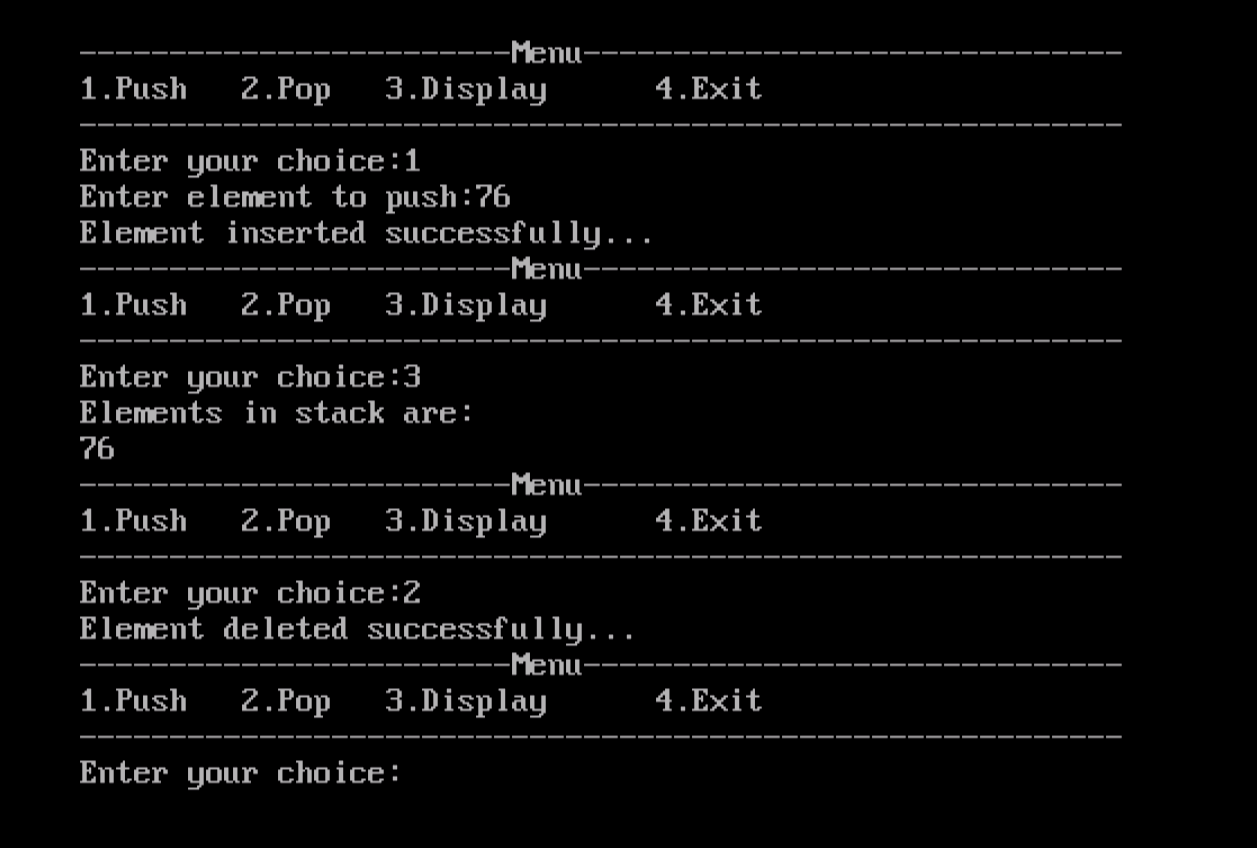
for(i=0;i<=top;i++)

printf("%d\t",stack[i]);

}

}

Output:



1. Write a program to implement stack using linked allocation.

Ans:

#include<stdio.h>

#include<conio.h>

void push();

void pop();

void display();

struct node

{

int val;

struct node \*next;

};

struct node \*head;

void main()

{

int ch=0;

clrscr();

while(ch!=4)

{

printf("\n--------------------------Menu--------------------------------");

printf("\n1.Push \t 2.Pop \t 3.Display \t 4.Exit");

printf("\n--------------------------------------------------------------");

printf("\nEnter your choice:");

scanf("%d",&ch);

switch(ch)

{

case 1:push();

break;

case 2:pop();

break;

case 3:display();

break;

case 4:exit(0);

break;

default:printf("Enter valid choice...");

}

}

}

void push()

{

int val;

struct node \*ptr=(struct node \*)malloc(sizeof(struct node));

if(ptr==NULL)

{

printf("Unable to push the element..");

}

else

{

printf("Enter the value:");

scanf("%d",&val);

if(head==NULL)

{

ptr->val=val;

ptr->next=NULL;

head=ptr;

}

else

{

ptr->val=val;

ptr->next=head;

head=ptr;

}

printf("Item pushed successfully..");

}

}

void pop()

{

int item;

struct node \*ptr;

if(head==NULL)

{

printf("Underflow..");

}

else

{

item=head->val;

ptr=head;

head=head->next;

free(ptr);

printf("Item poped..");

}

}

void display()

{

int i;

struct node \*ptr;

ptr=head;

if(ptr==NULL)

{

printf("Stack is empty..");

}

else

{

printf("Elements in stack are:\n");

while(ptr!=NULL)

{

printf("%d\t",ptr->val);

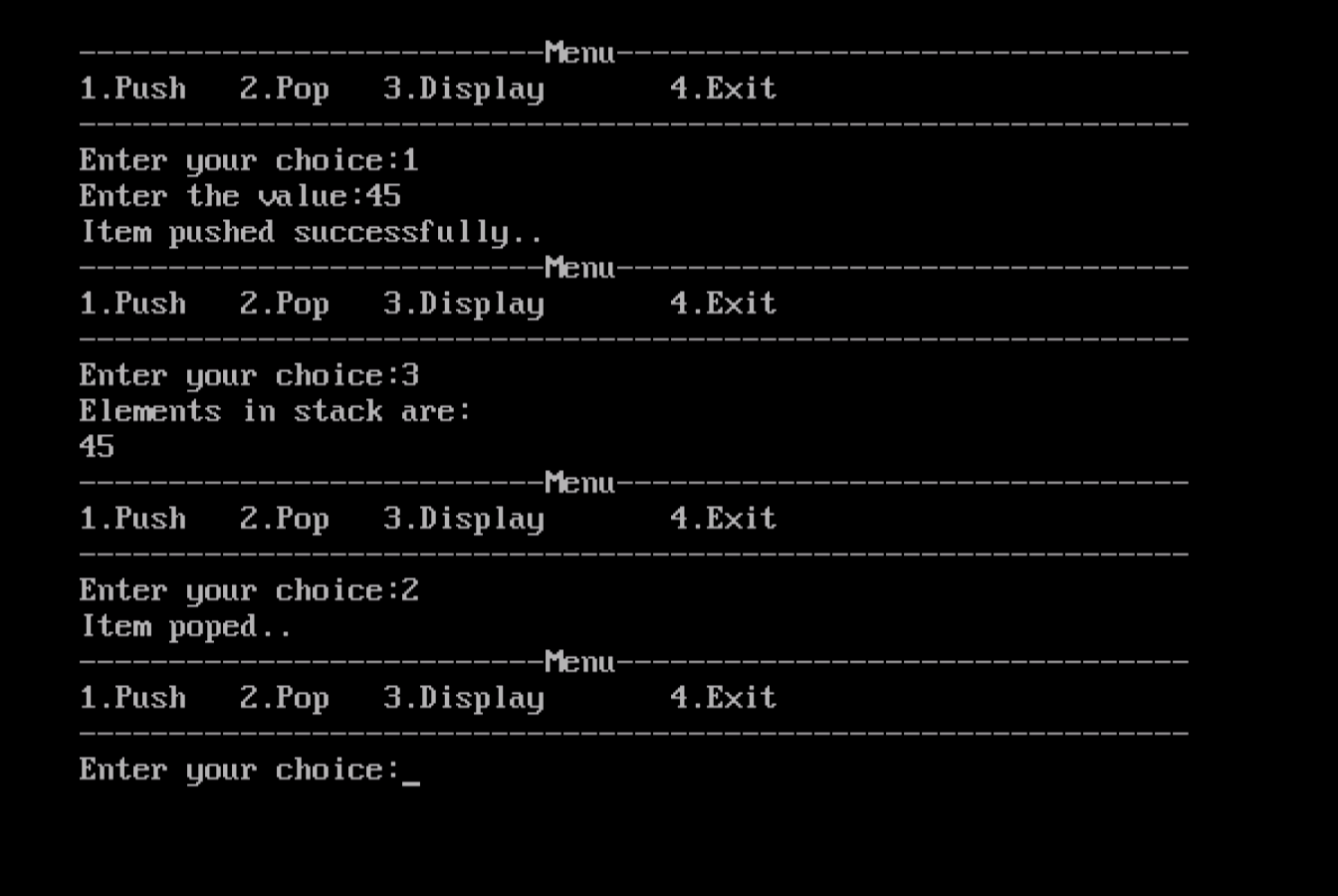
ptr=ptr->next;

}

}

}

Output:



1. Write a program to implement queue using array.

Ans:

#include<stdio.h>

#include<conio.h>

#define max 5

void enqueue();

void dequeue();

void display();

int front=-1,rear=-1;

int queue[max];

void main()

{

int ch;

clrscr();

while(ch!=4)

{

printf("\n----------------------Menu-----------------------------");

printf("\n1.Enqueue \t2.Dequeue \t3.Display \t4.Exit");

printf("\n-------------------------------------------------------");

printf("\nEnter your choice:");

scanf("%d",&ch);

switch(ch)

{

case 1:

enqueue();

break;

case 2:

dequeue();

break;

case 3:

display();

break;

case 4:

exit(0);

break;

default:

printf("Enter valid choice...");

}

}

}

void enqueue()

{

int item;

if(rear==max-1)

printf("Overflow...");

else

{

printf("Enter element:");

scanf("%d",&item);

if(front==-1 && rear==-1)

{

front=0;

rear=0;

}

else

rear=rear+1;

queue[rear]=item;

printf("Value inserted successfully...");

}

}

void dequeue()

{

int item;

if(front==-1 || front>rear)

printf("Underflow...");

else

{

item=queue[front];

if(front==rear)

{

front=-1;

rear=-1;

}

else

{

front=front+1;

}

printf("Value deleted successfully...");

}

}

void display()

{

int i;

if(rear==-1)

printf("Underflow...");

else

{

printf("Elements of queue are:\n");

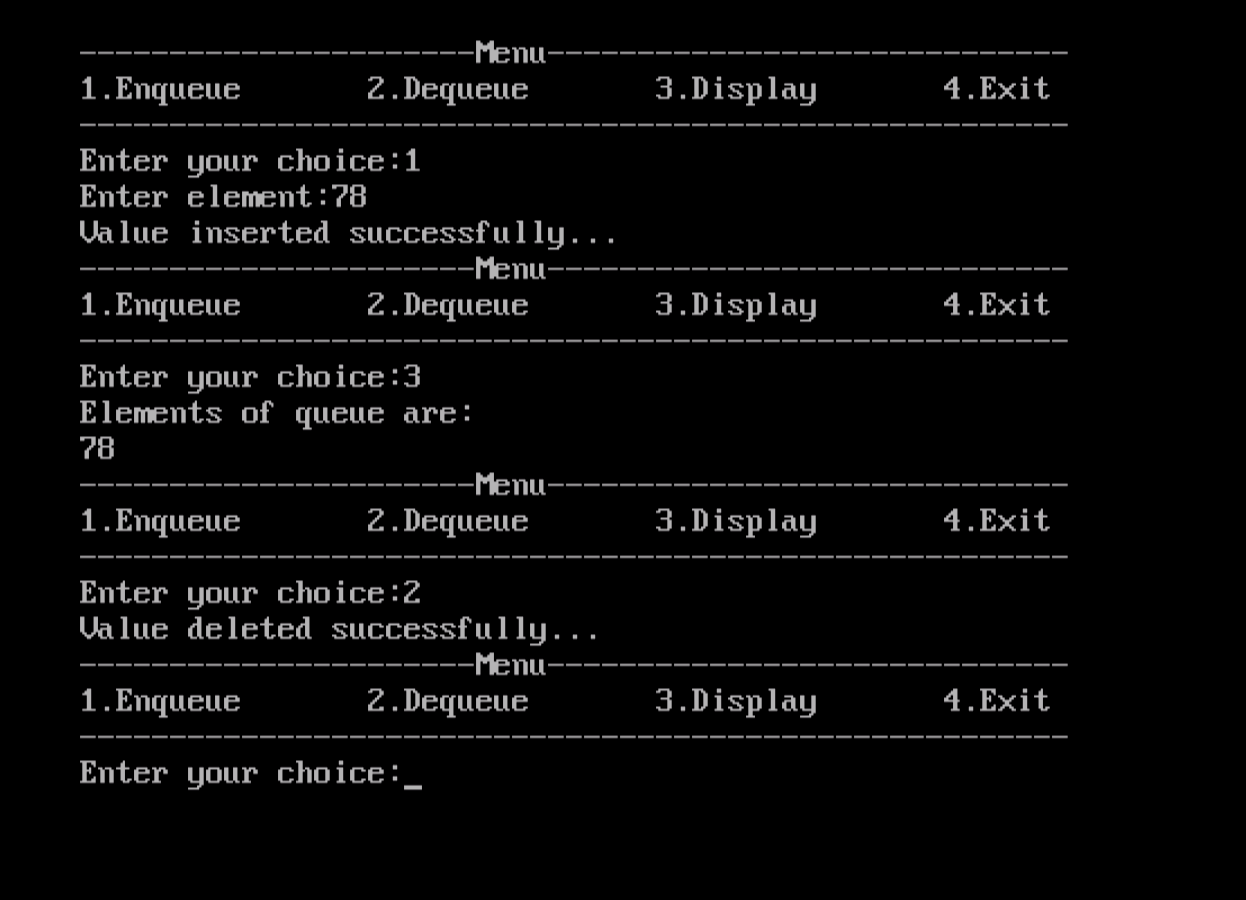
for(i=front;i<=rear;i++)

printf("%d\t",queue[i]);

}

}

Output:



1. Write a program to implement queue using linked allocation.

Ans:

#include<stdio.h>

#include<conio.h>

struct node

{

int data;

struct node \*next;

};

struct node \*front,\*rear;

void enqueue();

void dequeue();

void display();

void main()

{

int ch;

clrscr();

while(ch!=4)

{

printf("\n-------------------------Menu--------------------------------");

printf("\n1.Enqueue \t 2.Dequeue \t 3.Display \t 4.Exit");

printf("\n-------------------------------------------------------------");

printf("\nEnter your choice:");

scanf("%d",&ch);

switch(ch)

{

case 1:enqueue();

break;

case 2:dequeue();

break;

case 3:display();

break;

case 4:exit(0);

break;

default:printf("Invalid choice...");

}

}

}

void enqueue()

{

struct node \*ptr;

int item;

ptr=(struct node \*)malloc(sizeof(struct node));

if(ptr==NULL)

{

printf("Overflow...");

}

else

{

printf("Enter element:");

scanf("%d",&item);

ptr->data=item;

if(front==NULL)

{

front=ptr;

rear=ptr;

front->next=NULL;

rear->next=NULL;

printf("Element insert successfully..");

}

else

{

rear->next=ptr;

rear=ptr;

rear->next=NULL;

printf("Element insert successfully..");

}

}

}

void dequeue()

{

struct node \*ptr;

if(front==NULL)

{

printf("Underflow...");

}

else

{

ptr=front;

front=front->next;

free(ptr);

printf("Element deleted successfully...");

}

}

void display()

{

struct node \*ptr;

ptr=front;

if(front==NULL)

{

printf("Queue is empty..");

}

else

{

printf("Elements in queue are:\n");

while(ptr!=NULL)

{

printf("%d\t",ptr->data);

ptr=ptr->next;

}

}

}

Output:

