**C Programming and Data Structures**

**Computer Science and Technology-B**

**Week 13**

13. Implement the following Data Structures

a) Stack ADT

**Program:**

#include<stdio.h>

#include<stdlib.h>

int top=-1;

int stack[100];

void push();

void pop();

void display();

void main()

{

int a;

printf("\*\*\*MENU\*\*\*");

printf("\n1.Push\n2.Pop\n3.Display\n4.Exit\n");

while(a!=4)

{

printf("Enter your Choice:");

scanf("%d",&a);

switch(a)

{

case 1:

push();

break;

case 2:

pop();

break;

case 3:

display();

break;

case 4:

exit(0);

default:

printf("Enter a valid choice\n");

break;

}

}

}

void push()

{

int n;

if(top==99)

{

printf("Stack Overflow\n");

}

else

{

printf("Enter a value:");

scanf("%d",&n);

top=top+1;

stack[top]=n;

}

}

void pop()

{

if(top==-1)

{

printf("Stack Underflow\n");

}

else

{

printf("%d is poped out\n",stack[top]);

top=top-1;

}

}

void display()

{

int i;

if(top==-1)

{

printf("Stack is Empty\n");

}

else

{

printf("Stack Elements are:\n");

for(i=top;i>=0;i--)

{

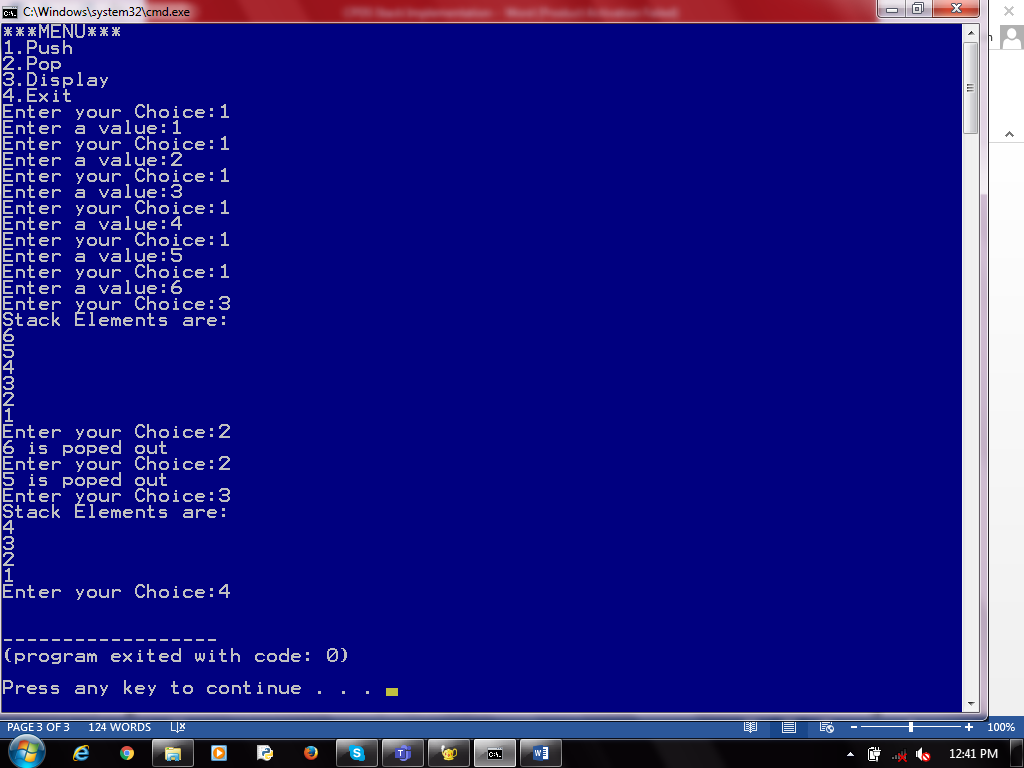
printf("%d\n",stack[i]);

}

}

}

**Output:**



b) Queue ADT

**Program:**

#include<stdio.h>

#include<stdlib.h>

int front=-1;

int rare=-1;

int queue[5]; //Global Variable Declaration

void enqueue();

void dequeue(); //Function prototype

void display();

void main()

{

int a;

printf("\*\*\*MENU\*\*\*");

printf("\n1.Enqueue\n2.Dequeue\n3.Display\n4.Exit\n");

while(a!=4)

{

printf("Enter your Choice:");

scanf("%d",&a);

switch(a)

{

case 1:

enqueue(); //function calling

break;

case 2:

dequeue();

break;

case 3:

display();

break;

case 4:

exit(0);

default:

printf("Enter a valid choice\n");

break;

}

}

}

void enqueue() //function defination

{

int n;

if(rare==5-1)

{

printf("Queue is Full\n");

}

else

{

if(front==-1)

{

front=0;

}

printf("Enter a value:");

scanf("%d",&n);

rare=rare+1;

queue[rare]=n;

}

}

void dequeue()

{

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

{

printf("Queue is Empty\n");

}

else

{

printf("%d is removed out\n",queue[front]);

front=front+1;

}

}

void display()

{

int i;

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

{

printf("Queue is Empty\n");

}

else

{

printf("Queue Elements are:\n");

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

{

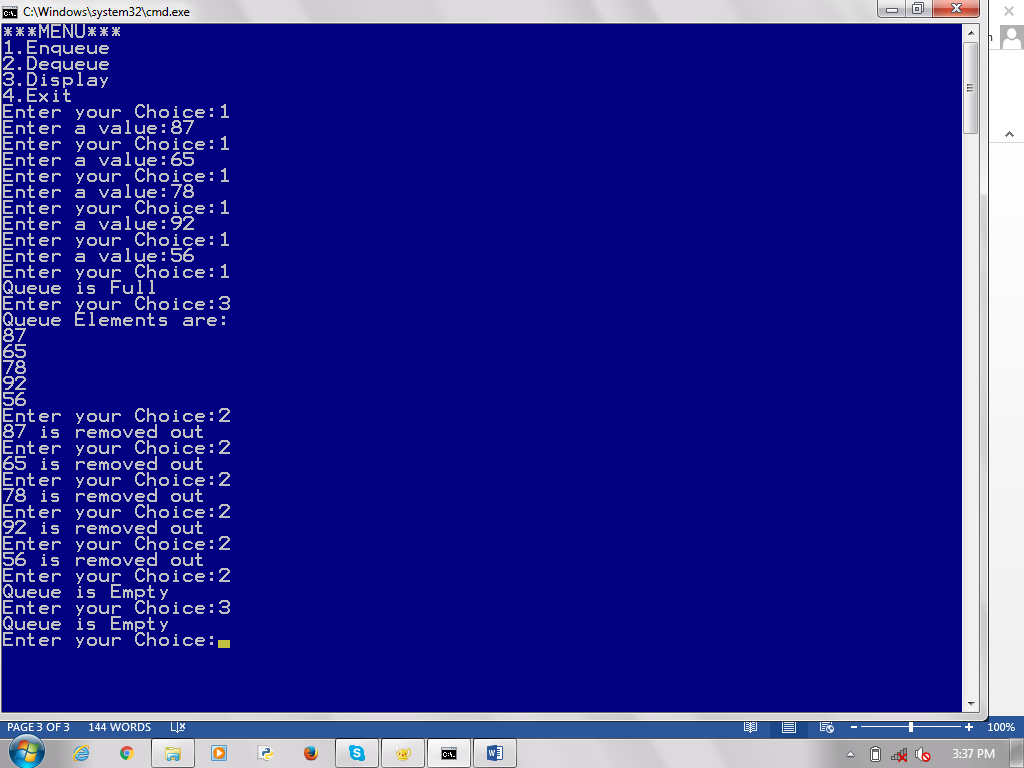
printf("%d\n",queue[i]);

}

}

}

**Output:**



c) Circular queue ADT

**Program:**

#include<stdio.h> //Circular Queue

#include<stdlib.h>

#define MAX 5

int front=-1;

int rear=-1;

int queue[MAX]; //Global Variable Declaration

void enqueue();

void dequeue(); //Function prototype

void display();

void main()

{

int a,n;

printf("\*\*\*MENU\*\*\*");

printf("\n1.Enqueue\n2.Dequeue\n3.Display\n4.Exit\n");

while(a!=4)

{

printf("Enter your Choice:");

scanf("%d",&a);

switch(a)

{

case 1:

printf("Enter a value:");

scanf("%d",&n);

enqueue(n); //function calling

break;

case 2:

dequeue();

break;

case 3:

display();

break;

case 4:

exit(0);

default:

printf("Enter a valid choice\n");

break;

}

}

}

void enqueue(int n) //function defination

{

if((front==0&&rear==MAX-1)||(front==rear+1))

{

printf("Queue is Full\n");

return;

}

if(front==-1)

{

front=0;

rear=0;

}

else

{

if(rear==MAX-1)

rear=0;

else

rear=rear+1;

}

queue[rear]=n;

}

void dequeue()

{

if(front==-1)

{

printf("Queue is Empty\n");

return;

}

printf("%d is removed out\n",queue[front]);

if(front==rear)

{

front=-1;

rear=-1;

}

else

{

if(front==MAX-1)

front=0;

else

front=front+1;

}

}

void display()

{

int front\_pos=front,rear\_pos=rear;;

if(front==-1)

{

printf("Queue is Empty\n");

return;

}

if(front\_pos<=rear\_pos)

while(front\_pos<=rear\_pos)

{

printf("%d",queue[front\_pos]);

front\_pos++;

}

else

{

while(front\_pos<=MAX-1)

{

printf("%d",queue[front\_pos]);

front\_pos++;

}

front\_pos=0;

while(front\_pos<=rear\_pos)

{

printf("%d",queue[front\_pos]);

front\_pos++;

}

}

printf("\n");

}

**Output:**

