AIM:-Queue Implementation using Array

Algorithm:-

Insert

Step 1: If maxsize-1=rear then this means the queue is full.

Step 2: But if rear<maxsize means that we can store an element in an array.

Step 3: So increment the rear value by 1 and then insert an element at the rear index.

Delete

Step 4: If front=-1 or front>rear then no element is available to delete.

Step 5: else delete front index element.

Step 6: if rear=front then set-1 to both front and rear.

Step 7: else we increment front.

DiStep 8: first check whether the queue is not empty.

Step 9: if empty we display that the queue is empty we simply return from the function and not execute further inside the function.

Step 10: else print all elements from front to rear.

Program:-

#include <stdio.h>

#define MAX 50

void insert();

void delete();

void display();

int queue\_array[MAX];

int rear = - 1;

int front = - 1;

main()

{

int choice;

while (1)

{

printf("1.Insert element to queue \n");

printf("2.Delete element from queue \n");

printf("3.Display all elements of queue \n");

printf("4.Quit \n");

printf("Enter your choice : ");

scanf("%d", &choice);

switch (choice)

{

case 1:

insert();

break;

case 2:

delete();

break;

case 3:

display();

break;

case 4:

\* Confidential \*

exit(1);

default:

printf("Wrong choice \n");

} /\* End of switch \*/

} /\* End of while \*/

} /\* End of main() \*/

void insert()

{

int add\_item;

if (rear == MAX - 1)

printf("Queue Overflow \n");

else

{

if (front == - 1)

/\*If queue is initially empty \*/

front = 0;

printf("Inset the element in queue : ");

scanf("%d", &add\_item);

rear = rear + 1;

queue\_array[rear] = add\_item;

}

} /\* End of insert() \*/

void delete()

{

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

{

printf("Queue Underflow \n");

return ;

}

else

{

printf("Element deleted from queue is : %d\n", queue\_array[front]);

front = front + 1;

}

} /\* End of delete() \*/

void display()

{

int i;

if (front == - 1)

printf("Queue is empty \n");

else

{

printf("Queue is : \n");

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

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

printf("\n");

}

}