FIRST SEMESTER MCA (2020 SCHEME)

PRACTICAL EXAMINATION JUNE-JULY 2021

20MCA135 Data Structures Lab.

Batch - II

Anu Rajan

~~[illegible]~~

ICE20MCA-2014

Date : 30 June 2021

Time : 9:30 - 12:30

1. Merging of 2 sorted Arrays.

<u>Algorithm</u>.

```
1. Start
2. Declare the size of the array
3. Declare the array.
4. While initializing ~~copy~~ ~~all~~ the first and second array
   copy the elements of the array to merged array
5. Sort the merged array
6. Display the resulting array
7. Stop.
```

<u>Default i/p and</u> o/p :-

```
Enter the size of 1st array - 3
Enter the sorted elements of 1st array - 1, 2, 3
Enter the size of 2nd array - 2
Enter the sorted elements of 2nd array - 4,5
After Merging
1
2
3
4
5
```

Program :

```
# include < stdio.h>
# include < conio.h>
void main ()
{
int arr1[50], arr2[50], arr3[100], m, n, i, j, k =0;
clrscr ();
printf ("\n Enter the size of the 1st Array :");
scanf ("%d", &m);
printf ("\n Enter the sorted elements of 1st array :");
for (i=0; i<m; i++)
{
scanf ("%d", ~~&array~~ &arr1[i]):
}
printf ("\n Enter the size of the 2nd Array :");
scanf ("%d", &n);
printf ("\n Enter the sorted elements of 2nd array :");
for (i=0 ; i<n; i++)
{
scanf ("%d", &arr2[i]);
}
i=0;
j=0;
while (i<m&&j<n)
{
if array1 [i]<array2 [i])
{
arr3[k] = arr1 [i];
i++;
}
else
{
```

```
arr3[k]=arr2[j];
j++;
}
k++;
}
if(i>=m)
{
while(j<n)
{
arr3[k]=arr2[j];
j++;
k++;
}
}
if(j>=n)
{
while(i<m)
{
~~arr3~~ arr3[k]=arr1[i];
i++;
k++;
}
}
printf("\n After merging \n");
for(i=0;i<m+n;i++)
{
printf("\n %d", arr3[i]);
}
getch();
}.
```

Output -

```
Enter the size of the 1st array - 3
Enter the sorted elements of 1st array - 1  2  3
Enter the size of the 2nd array - 3
Enter the sorted elements of 2nd array - 5  6  7
After Merging :
 1
 2
 3
 5
 6
 7.
```

2. Implement circular queue.

Algorithm for insertion

```
Step 1 : If (rear+1) % MAX = front
            Write " Overflow"
            Goto step 4.
Step 2: If front = -1 and rear = -1
           Set front = rear = 0
           else if rear = MAX-1 and front != 0
           set rear = 0
           else
           set rear = (rear+1) % max
Step 3: Set queue [rear] = val
Step 4: Exit.
```

Algorithm for deletion

```
1. If front = -1
   Write "underflow"
   Goto step 4
2. Set val = queue [front]
3. If front = rear
   set front = real = -1
   else
   If front = MAX - 1
   Set front = 0
   else
   Set front = front + 1
4. Exit -
```

Program.

```
#include <stdio.h>
#include <conio.h>
# define MAX 5
int cqueue_arr[MAX];
int front = -1;
int rear = -1;
void insert (int item)
{
if ((front = = 0&&rear = = MAX -1)||(front == rear+1))
{
printf ("Queue overflow \n");
return;
}
if (front == -1)
{
```

```
front = 0;
rear = 0;
}
else
{
if(rear= MAX-1)
rear =0;
else
rear = rear+1 ;
}
cqueue_arr[rear]= item;
}
void deletion ()
{
if (front == -1)
{
printf("Queue Underflow");
return;
}
printf ("Element deleted from queue is: %d\n", ~~cqueue~~_
                                  cqueue_arr[front];      [illegible]

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)
{
point ("Queue is Empty\n");
return;
}
pointf ("Queue elements:");
if(front_pos <=rear_pos)
while(front_pos <=rear_pos)
{
pointf (" %d\t", cqueue_arr [front_pos]);
front_pos++;
}
else
{
while (front_pos <= MAX-1)
{
pointf ("%d", cqueue_arr [front_pos]);
front_pos++;
}
front_pos =0;
while (front_pos <= rear_pos)
{
pointf (" %d", cqueue_arr [front_pos]);
front_pos++;
}
}
```

```
printf("\n");
}
void main ()
{
int choice, item;
clrscr();
do
{
printf("\n__Circular Queue__\n");
printf("1. Insert \n");
printf("2. Delete \n");
printf(" 3. Display \n");
printf(" 4. Quit \n");
printf (" Enter your choice :");
scanf ("%d", &choice);
Switch (choice)
{
case 1:
printf(" Insert the element:");
scanf("%d" &item);
insert (item);
break;
case 2:
deletion ();
break;
case 3:
display();
break;
case 4:
[illegible] exit ();
break:
default :
```

```
printf("Wrong choice ");
 }
 }
while (choice != 4);
getch();
 }
```

Output.

```
1. Insert
2. Delete
3. Display
4. Quit
Enter your choice :1
Insert the element : 10

1. insert
2. Delete
3. Display
4. Quit
Enter your choice : 1
Insert the element : 20

1. insert
2. Delete
3. Display
4. Quit
Enter your choice : 2
Element deleted from queue is : 10

1. insert
2. Delete
3. Display
4. Quit
Enter your choice : 3
Queue elements : 20
```