Batch-II FIRST SEMESTER MCA (2020 SCHEME) Pratical Examination
June 2021.

20MCA132 – DataStructure LAB.

Date: 30-06-2021

Time: 9:30am – 12:30pm

Submitted by

Anjana. K. P

ICE20MCA2011

1. Merging of two arrays.

Algorithm:

Step 1: Start

Step 2: Declare 3 integer arrays and variables m, n, i, j, k=0

Step 3: ~~Enter~~ Read the number of elements in first array as m.

Step 4: Read the elements in the first array.

Step 5: Read the number of elements in second array as n.

Step 6: Read the elements in the second array.

Step 7: initialise i and j as 0.

Step 8: ~~while i<m and j<n then~~, Repeat step 9 to step 12. until i<m && j<n.

Step 9: check if a[i] < b[j] then,

Step 10: assign the element a[i] to C[k], and increment i by 1.

Step 11: else assign the element b[j] to C[k].

Step 12: increment j by 1 and k by 1.

Step 13: check i >= m then,

Step 14: Repeat step 15 to step 16 until j<n.

Step 15 : Assign the element in arr b[j] to c[k]

Step 16 : increment j and k by 1.

Step 17 : check j>=n. then,

Step 18 : Repeat step 19 to step20 until i<m.

Step 19 : Assign the ~~value~~ element in a[i] to c[k]

Step 20 : increment j and k by 1.

Step 21 : print the merged array.

step 22 : stop.

Default input and output

~~output~~ ~~input~~:

Input: the element in first array 1 2 3

element in IInd array 5 6 7 9

Output: Merged array: 1 2 3 5 6 7 9

Program code:

```
#include <stdio.h>
#include <conio.h>
  Void main ()
{ int arr1[50], arr2[50], arr3[100], m, n, i, j, k=0;
  Printf("Enter the number of elements in 1st array:\n");
  Scanf("%d", &m);
  printf("Enter the elements:\n");
  for(i=0; i<m; i++)
  scanf("%d", arr1[i]);
printf("Enter the no of elements in IInd array:\n");
  Scanf("%d", &n);
```

```
printf("enter the elements :\n");
for(i =0; i<n; (++)
   scanf("%d", &arr2[i]);
 i=0; j=0;
while(i<m && j<n)
{ if(arr1[i] < arr2[j])
   { arr3[k] = arr1[i];
     i++;
   }
else   arr3[k] =arr2[j];
       j++;
}
k++;
}
if ((>=n)
{while(j <n)
{ arr3[k] =arr2[j];
   j++; k++;
}
}
if(j >=n)
{ while (i<n)
  { arr3[k] = arr1[i];
     i++; j++;
}
printf("%d\n" Merged array:\n");
 for(i=0; i <n+m;i++)
 {printf("%d\n", arr3[i]);
 }
getch();
```

3 Display
4 Exit
Choice : 1
element : 23
Choice : 1
element : 25
Choice : 2
Deleted element is : 23
Choice : 3
Queue elements :
25.

2. Implement circular queue.

Algorithm:

Step 1 : Start.
Step 2 : Define MAX as 5.
Step 3 : Declare an integer array of size MAX and two var
as front and rear initialize them with -1 as g
~~Step 3 : Declare the function insert.~~
Step 3 : Declare two ~~variables~~ integer variables choic
item.
Step 4 : Repeat step 4 to step 11. until choice !=
Step 5 : print the menu and read choice as
Step 6 : ~~case switch~~. Repeat step 7 to step 11 until
if choice is 1
Step 7 : read the value as item.
Step 8 : call the insert function.
if choice is 2:
Step 9 : call deletion function.

Step 10 : if choice is 3 call display function.

Step 11 : if choice is 4 exit.

Step 12 : Stop.

Algorithm of insert function.

Step 1 : Start.

Step 2 : check front = 0 and rear = max-1 then,

Step 3 : print queue overflow and return.

Step 4 : check if front = -1 then,

Step 5 : set front and rear as 0.

Step 6 : else, check if rear = max-1 then,

Step 7 : set rear as 0

Step 8 : else, rear increment rear by 1.

Step 9 : assign item to the array.

Step 10 : Stop.

Algorithm of Deletion function.

Step 1 : Start

Step 2 : check if front = -1 then,

Step 3 : print queue underflow and return.

Step 4 : print the element deleted from the queue.

Step 5 : check if (front == rear) then,

Step 6 : set front and rear as -1.

Step 7 : else, check if front = MAX-1 then,

Step 8 : Assign front as 0.

Step 9 : else, increment front by 1

Step 10 : Stop.

Algorithm for display function

Step 1 : Start
Step 2 : initialise the integer variables as, front-pos = front and rear-pos = rear.
Step 3 : check if front = -1 : then,
Step 4 : print queue is empty and return.
Step 5 : print the queue elements,
Step 6 : check if front-pos <= rear-pos
Step 7 : Repeat steps to step 8 until front-pos <= rear-pos.
Step 8 : print the value of front-pos in array and increment front pos by 1.
Step 9 : else, Repeat step 10 to to step 10 until front-pos <= rear-1
Step 10 : print the value of front-pos in array and increment front-pos by 1
Step 11 : set front-pos as 0.
Step 12 : Repeat step 13 to step 15 until front pos <= rear-pos.
Step 13 : print value of front pos in array
Step 14 : Increment front-pos by 1.
Step 15 : Stop.

Default input and output:

choice : 1
element : 23
choice : 1
element : 25
choice : 2
deleted element is 23
choice : 3 Queue element are 25.

Program code:

```
#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\n");
    return;
 }
printf("element deleted from Queue is :%d\n", cqueue_arr[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;
    }
printf("Queue elements\n");
if(front_pos <= rear_pos)
while(front_pos <= rear_pos)
{ printf("%d", cqueue_arr[front_pos]);
    front_pos++;
}
else
{
    while(front_pos <= MAX-1)
    {printf("%d ", cqueue_arr[front_pos]);
```

```
    front_pos++;
  }
}
void main()
{ int choice, item;
do{ printf("1. Insertion \n, 2. Deletion \n  3. Display \n  4. Exit\n");
  printf("Enter your choice : \n");
    scanf("%d", &choice);
 Switch(choice)
{
  case 1: printf("input element for insertion:");
          scanf("%d", item); break; insert(item);
            break;
  Case 2 : deletion(); break;
    case 3 : display(); break;
    case 4 : break;
  }
 }
while(choice != 4);
    getch();
}.
```