Arrray Implementation of A Double Ended Queue

```
//Arrray Implementation of A Double Ended Queue
#include<stdio.h>
int rear=0,front=0,size,dequeue[20];
void push();
void pop();
void inject();
void eject();
void display();
main()
{
  int i,choice;
  printf("\n Enter the Size of the DeQueue : ");
  scanf("%d",&size);
  do
    {
      printf("\nMENU\n1.Push(Front)\n2.Pop(Front)\n3.Inject(Rear)\n4.Eject(
      printf("\nEnter your choice\t");
      scanf("%d",&choice);
      switch(choice)
  {
  case 1:
    push();
    break;
  case 2:
    pop();
    break;
  case 3:
    inject();
    break;
  case 4:
    eject();
    break;
  case 5:
    display();
    break;
  case 6:
    exit(0);
  default:
    printf("\nInvalid entry");
  }
    }while(choice!=6);
}

void push()
{
  int temp;
```

```
  if(front==0)
    temp=1;
  else if(front==1)
    temp=size;
  else
    temp=temp-1;
  if(temp==rear)
    printf("\nThe dequeue is full\n");
  else
    {
      front=temp;
      if(rear==0)
  rear=1;
      printf("\nEnter the element to be inserted: ");
      scanf("%d",&dequeue[front]);
    }
}

void eject()
{
  int item;
  if(front==0)
    printf("\nThe dequeue is empty\n");
  else
    {
      item=dequeue[rear];
      printf("\nThe deleted element is %d\n",item);
      if(rear==front)
  {
    front=0;
    rear=0;
  }
      else if(rear==1)
  rear=size;
      else
  rear=rear-1;
    }
}

void inject()
{
  int i;
  if(((rear%size)+1)==front)
    {
      printf("\nQueue is full\n");
    }
  else
    {
```

```
      for(i=0;i<1;i++)
  {
    rear=((rear%size)+1);
    printf("\nEnter the element to be inserted ");
    scanf("%d",&dequeue[rear]);
  }
    }
  if(front==0)
    {
      front=1;
    }
}
void pop()
{
  int item;
  if(front==0)
    {
      printf("\nDequeue is empty\n");
    }
  else
    {
      item=dequeue[front];
      printf("\nThe deleted element is %d",item);
      if(front==rear)
  {
    front=0;
    rear=0;
  }
      else
  {
    front=(front%size)+1;
  }
    }
}
void display()
{
  int i;
  if(front==0)
    {
      printf("\nDequeue is empty\n");
    }
  else
    {
      printf("\nThe dequeue is\n");
      for(i=front;i!=rear;i=((i%size)+1))
        {
          printf(" %d ",dequeue[i]);
        }
```

```
        printf(" %d ",dequeue[rear]);
    }
}
```

Output

```
 Enter the Size of the DeQueue : 4

MENU
1.Push(Front)
2.Pop(Front)
3.Inject(Rear)
4.Eject(Rear)
5.Display
6.Exit
Enter your choice 1

Enter the element to be inserted: 3

MENU
1.Push(Front)
2.Pop(Front)
3.Inject(Rear)
4.Eject(Rear)
5.Display
6.Exit
Enter your choice 1

Enter the element to be inserted: 2

MENU
1.Push(Front)
2.Pop(Front)
3.Inject(Rear)
4.Eject(Rear)
5.Display
6.Exit
Enter your choice 3

Enter the element to be inserted 5

MENU
1.Push(Front)
2.Pop(Front)
3.Inject(Rear)
4.Eject(Rear)
5.Display
6.Exit
Enter your choice 3
```

```
Enter the element to be inserted 6

MENU
1.Push(Front)
2.Pop(Front)
3.Inject(Rear)
4.Eject(Rear)
5.Display
6.Exit
Enter your choice 5

The dequeue is
 2  3  5  6
MENU
1.Push(Front)
2.Pop(Front)
3.Inject(Rear)
4.Eject(Rear)
5.Display
6.Exit
Enter your choice 2

The deleted element is 2
MENU
1.Push(Front)
2.Pop(Front)
3.Inject(Rear)
4.Eject(Rear)
5.Display
6.Exit
Enter your choice 4

The deleted element is 6

MENU
1.Push(Front)
2.Pop(Front)
3.Inject(Rear)
4.Eject(Rear)
5.Display
6.Exit
Enter your choice 5

The dequeue is
 3  5
MENU
1.Push(Front)
```

```
2.Pop(Front)
3.Inject(Rear)
4.Eject(Rear)
5.Display
6.Exit
Enter your choice 6
42813@user:/mnt/42813/dslab/latex/programs/1$
```

Polynomial Addition using Structures

```
// polynomial addition
#include<stdio.h>

struct polynomial
{
  int exp,coeff;
}poly1[10],poly2[10],poly3[10];


main()
{
  int i,temp1=0,temp2=0,temp3=0,count1,count2,count3;


  //READ POLYNOMIALS
  printf("\nEnter the no of terms in Poly1: ");
  scanf("%d",&count1);

  printf("\nEnter the elements\n");
  for(i=count1;i>=0;i--)
    {
      printf("Enter the coefficient for degree%d: ",i);
      scanf("%d",&poly1[i].coeff);
      poly1[i].exp=i;
    }

  printf("\n\nEnter the no of terms in Poly2: ");
  scanf("%d",&count2);

  printf("\nEnter the elements\n");
  for(i=count2;i>=0;i--)
    {
      printf("Enter the coefficient for degree%d: ",i);
      scanf("%d",&poly2[i].coeff);
      poly2[i].exp=i;
    }


  //BEGIN ADDITION
  while(temp1<=count1 && temp2<=count2)
    {
      if(poly1[temp1].exp>poly2[temp2].exp)
  {
    poly3[temp3].exp=poly1[temp1].exp;
    poly3[temp3].coeff=poly1[temp1].coeff;
    temp3++;
```

```
    temp1++;
  }
      else if(poly1[temp1].exp<poly2[temp2].exp)
  {
    poly3[temp3].exp=poly2[temp2].exp;
    poly3[temp3].coeff=poly2[temp2].coeff;
    temp3++;
    temp2++;
  }
      else
  {
    poly3[temp3].exp=poly1[temp1].exp;
    poly3[temp3].coeff=poly1[temp1].coeff+poly2[temp2].coeff;
    temp3++;
    temp2++;
    temp1++;
  }
    }

  while(temp1<=count1)
    {
      poly3[temp3].exp=poly1[temp1].exp;
      poly3[temp3].coeff=poly1[temp1].coeff;
      temp3++;
      temp1++;
    }

  while(temp2<=count2)
    {
      poly3[temp3].exp=poly2[temp2].exp;
      poly3[temp3].coeff=poly2[temp2].coeff;
      temp3++;
      temp2++;
    }

  printf("\nThe Entered Polynomials:\n");
  printf("\nPolynomial1\n");
  for(i=count1;i>=0;i--)
    {
      printf("%dx^%d",poly1[i].coeff,poly1[i].exp);
      if((i+1)!=1)
  printf(" + ");
      else
  printf(" =0\n");
    }

  printf("\nPolynomial2\n");
  for(i=count2;i>=0;i--)
```

```
    {
      printf("%dx^%d",poly2[i].coeff,poly2[i].exp);
      if((i+1)!=1)
  printf(" + ");
      else
  printf(" =0\n");
    }

  printf("\nPolynomial After addition is\n");
  for(i=temp3-1;i>=0;i--)
    {
      printf("%dx^%d",poly3[i].coeff,poly3[i].exp);
      if((i+1)!=1)
  printf(" + ");
      else
  printf(" =0\n");
    }
}
```

Output

```
 Enter the no of terms in Poly1: 3

Enter the elements
Enter the coefficient for degree3: 5
Enter the coefficient for degree2: 6
Enter the coefficient for degree1: 4
Enter the coefficient for degree0: 2


Enter the no of terms in Poly2: 4

Enter the elements
Enter the coefficient for degree4: 2
Enter the coefficient for degree3: 4
Enter the coefficient for degree2: 5
Enter the coefficient for degree1: 2
Enter the coefficient for degree0: 1

The Entered Polynomials:

Polynomial1
5x^3 + 6x^2 + 4x^1 + 2x^0 =0

Polynomial2
2x^4 + 4x^3 + 5x^2 + 2x^1 + 1x^0 =0

Polynomial After addition is
2x^4 + 9x^3 + 11x^2 + 6x^1 + 3x^0 =0
```

Multiqueue Implimentation using Matrix

```
/*Multiqueue implimentation using matrix*/

#include<stdio.h>
void insert();
void delete();
void display();
int r[10],f[10],a[10][10],n,pr;

main()
{
  printf("\n Enter the Priority Range: ");
  scanf("%d",&n);
  int choice,i;
  //Init-Priority
  for(i=1;i<=n;i++)
    {
      r[i]=0;
      f[i]=0;
    }
  do
    {
      printf("\n[1]Insert\n[2]Delete\n[3]Display\n[4]Exit\nEnter
         your choice: ");
      scanf("%d",&choice);
      switch(choice)
  {
  case 1:
    insert();
    break;
  case 2:
    delete();
    break;
  case 3:
    display();
    break;
  case 4:
    exit(0);
  default:
    printf("\nInvalid choice");
  }
    }while(choice!=4);
}



void insert()
```

```
{
  int e;
  printf("\nEnter an element: ");
 scanf("%d",&e);
  printf("\nEnter the priority of %d -->",e);
  scanf("%d",&pr);
  if(pr>n)
    printf("\nPriority is out of range ");
  else
    {
      if(r[pr]>=n)
  printf("\nThe queue is full");
      else
  {
          r[pr]=r[pr]+1;
    a[pr][r[pr]]=e;
    if(f[pr]==0)
      f[pr]=1;
  }
    }
}



void delete()
{
  int item;
  pr=n;
  while(pr>=1)
    {
      if(f[pr]==0)
  {
    printf("\nThe queue of priority %d is empty\n",pr);
    pr=pr-1;
  }
      else
  {
    item=a[pr][f[pr]];
    a[pr][f[pr]]=0;
          printf("\nthe deleted element is %d with priority
            %d\n",item,pr);
    if(f[pr]==r[pr])
      {
        f[pr]=0;
        r[pr]=0;
      }
    else
```

```
        f[pr]=f[pr]+1;
      break;
    }
      }
}



void display()
{
  int i,j;
  for(i=1;i<=n;i++)
    {
      printf(" Priority %d : ",i);
      for(j=1;j<=3;j++)
  printf(" %d ",a[i][j]);
      printf("\n");
    }
}
```

Output

```
Enter the Priority Range: 3

[1]Insert
[2]Delete
[3]Display
[4]Exit
Enter your choice: 1

Enter an element: 2

Enter the priority of 2 -->1

[1]Insert
[2]Delete
[3]Display
[4]Exit
Enter your choice: 1

Enter an element: 3

Enter the priority of 3 -->1

[1]Insert
[2]Delete
[3]Display
[4]Exit
Enter your choice: 1
```

```
Enter an element: 5

Enter the priority of 5 -->3

[1] Insert
[2] Delete
[3] Display
[4] Exit
Enter your choice: 3
 Priority 1 :  2  3  0
 Priority 2 :  0  0  0
 Priority 3 :  5  0  0

[1] Insert
[2] Delete
[3] Display
[4] Exit
Enter your choice: 2

the deleted element is 5 with priority 3

[1] Insert
[2] Delete
[3] Display
[4] Exit
Enter your choice: 2

The queue of priority 3 is empty

The queue of priority 2 is empty

the deleted element is 2 with priority 1

[1] Insert
[2] Delete
[3] Display
[4] Exit
Enter your choice: 2

The queue of priority 3 is empty

The queue of priority 2 is empty

the deleted element is 3 with priority 1

[1] Insert
[2] Delete
```

```
[3] Display
[4] Exit
Enter your choice: 3
 Priority 1 :  0  0  0
 Priority 2 :  0  0  0
 Priority 3 :  0  0  0

[1] Insert
[2] Delete
[3] Display
[4] Exit
Enter your choice: 4
```