adt.h:

struct Queue

{

    int arr[100],f,r,iter,size;

};

void enqueu(struct Queue \*Q, int x);

int dequeue(struct Queue \*Q);

void disp(struct Queue \*Q);

int isEmpty(struct Queue \*Q);

int isFull(struct Queue \*Q);

impl.h:

#include "adt.h"

#include<stdio.h>

void enqueue(struct Queue \*q, int x)

{

    if(isFull(q))

        printf("Queue is full");

    else

    {

        q->iter++;

        q->r=(++q->r)%q->size;

        q->arr[q->r]=x;

    }

}

int dequeue(struct Queue \*q)

{

    if(isEmpty(q))

        printf("Queue is empty");

    else

    {

        q->iter--;

        q->f=(++q->f)%q->size;

        return q->arr[q->f+1];

    }

}

void disp(struct Queue \*q)

{

    for(int i=q->f+1;i!=q->r;i=(++i)%q->size)

    {

        printf("%d ",q->arr[i]);

    }

    printf("%d\n",q->arr[q->r]);

}

int isFull(struct Queue\*q)

{

    if(q->iter==q->size)

        return 1;

    else

        return 0;

}

int isEmpty(struct Queue\*q)

{

    if(q->iter==0)

        return 1;

    else

        return 0;

}

appl.c:

#include "impl.h"

#include<stdio.h>

int main()

{

    struct Queue q;

    int ch,val;

    q.f=-1;

    q.r=-1;

    q.iter=0;

    printf("Enter size:");

    scanf("%d",&q.size);

    do

    {

        printf("\n1)Enqueue 2)Dequeue 3)Display 4)Exit:");

        scanf(" %d",&ch);

        if(ch==1)

        {

            printf("\nEnter element:");

            scanf(" %d",&val);

            enqueue(&q,val);

        }

        else if (ch==2)

        {

            dequeue(&q);

        }

        else if(ch==3)

        {

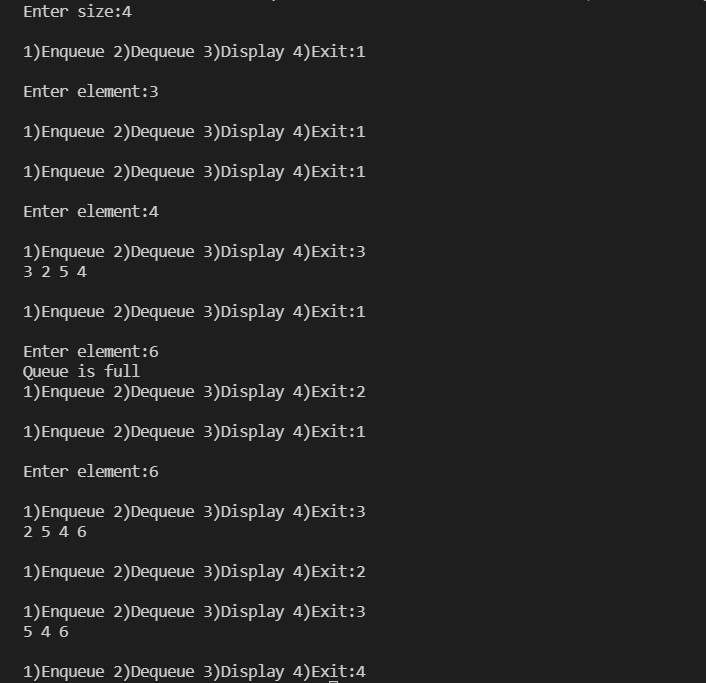
            disp(&q);

        }

    } while (ch!=4);

}

o/p:



Application:

adt.h:

struct data

{

    int jobno,bst;

};

struct Queue

{

    struct data arr[100];

    int f,r,iter,size;

};

void enqueu(struct Queue \*Q, int x);

int dequeue(struct Queue \*Q);

void disp(struct Queue \*Q);

int isEmpty(struct Queue \*Q);

int isFull(struct Queue \*Q);

impl.h:

#include "adt.h"

#include<stdio.h>

void enqueue(struct Queue \*q,struct data d)

{

    if(isFull(q))

        printf("Queue is full");

    else

    {

        q->iter++;

        q->r=(++q->r)%q->size;

        q->arr[q->r]=d;

    }

}

void disp(struct Queue \*q)

{

    for(int i=q->f+1;i!=q->r;i=(++i)%q->size)

    {

        printf("%d %d\n",q->arr[i].jobno,q->arr[i].bst);

    }

    printf("%d %d\n",q->arr[q->r].jobno,q->arr[q->r].bst);

}

int isFull(struct Queue\*q)

{

    if(q->iter==q->size)

        return 1;

    else

        return 0;

}

int isEmpty(struct Queue\*q)

{

    if(q->iter==0)

        return 1;

    else

        return 0;

}

appl.c:

#include "impl.h"

#include<stdio.h>

int main()

{

    struct Queue q1,q2;

    int ch,job,bst;

    q1.f=-1;

    q1.r=-1;

    q1.iter=0;

    q2.f=-1;

    q2.r=-1;

    q2.iter=0;

    printf("Enter size:");

    scanf("%d",&q1.size);

    q2.size=q1.size;

    do

    {

        printf("\n1)Insert 2)Display 3)Exit:");

        scanf(" %d",&ch);

        if(ch==1)

        {

            printf("\nEnter job no. and cpu burst time:");

            scanf(" %d %d",&job,&bst);

            struct data d;

            d.jobno=job;

            d.bst=bst;

            if(isEmpty(&q1))

                enqueue(&q1,d);

            else if(isEmpty(&q2))

                enqueue(&q2,d);

            else

            {

                int avg1=0,avg2=0,sum=0;

                for(int i=q1.f+1;i!=q1.r;i=(++i)%q1.size)

                {

                    sum+=q1.arr[i].bst;

                }

                sum+=q1.arr[q1.r].bst;

                avg1=sum/q1.iter;

                sum=0;

                for(int i=q2.f+1;i!=q2.r;i=(++i)%q2.size)

                {

                    sum+=q2.arr[i].bst;

                }

                sum+=q2.arr[q2.r].bst;

                avg2=sum/q2.iter;

                if(avg1<avg2)

                    enqueue(&q1,d);

                else

                    enqueue(&q2,d);

            }

        }

        else if (ch==2)

        {

            printf("\nQueue 1: ");

            disp(&q1);

            printf("\nQueue 2: ");

            disp(&q2);

        }

    } while (ch!=3);

}

o/p:

