Practical 11

//createtion of job queue by operating system jobs are proceesed in the order they enter the system in this case os dose not use priority write cpp program for simulating job queue write fun to add job and delete job from queue

#include <iostream>

#define MAX 10

using namespace std;

struct queue

{ int data[MAX];

int front,rear;

};

class Queue

{ struct queue q;

public:

Queue(){q.front=q.rear=-1;}

int isempty();

int isfull();

void enqueue(int);

int delqueue();

void display();

};

int Queue::isempty()

{

return(q.front==q.rear)?1:0;

}

int Queue::isfull()

{ return(q.rear==MAX-1)?1:0;}

void Queue::enqueue(int x)

{q.data[++q.rear]=x;}

int Queue::delqueue()

{return q.data[++q.front];}

void Queue::display()

{ int i;

cout<<"\n";

for(i=q.front+1;i<=q.rear;i++)

cout<<q.data[i]<<" ";

}

int main()

{ Queue obj;

int ch,x;

do{ cout<<"\n 1.Insert Job\n 2.Delete Job\n 3.Display\n 4.Exit\n Enter your choice : ";

cin>>ch;

switch(ch)

{ case 1: if (!obj.isfull())

{ cout<<"\n Enter data : \n";

cin>>x;

obj.enqueue(x);

cout<<endl;

}

else

cout<< "Queue is overflow!!!\n\n";

break;

case 2: if(!obj.isempty())

cout<<"\n Deleted Element = "<<obj.delqueue()<<endl;

else

{ cout<<"\n Queue is underflow!!!\n\n"; }

cout<<"\nRemaining Jobs : \n";

obj.display();

break;

case 3: if (!obj.isempty())

{ cout<<"\n Queue contains : \n";

obj.display();

}

else

cout<<"\n Queue is empty!!!\n\n";

break;

case 4: cout<<"\n Exiting Program.....";

}

}while(ch!=4);

return 0;

}

