6.2.program that Implements Circular Queue using Array.

#include <conio.h> #include <iostream> using namespace std; class queue

{

private:

int data; queue\* next;

public:

void add(int); void display(void); int remove(void);

};

queue\* front; queue\* rear;

void queue::display(void)

{

queue\* q = front;

if ((front == NULL) && (rear == NULL))

{

cout << "queue is empty";

}

else

{

do

{

cout << q->data << " "; q = q->next;

} while (q != front);

}

}

void queue::add(int num)

{

if ((front == NULL) && (rear == NULL))

{

front = rear = new queue; front->data = num;

front->next = front;

}

else

{

rear->next = new queue; rear->next->data = num; rear->next->next = front; rear = rear->next;

}

}

int queue::remove(void)

{

int num; queue\* temp;

if ((front == NULL) && (rear == NULL))

{

return -1;

}

else

{

if (front == rear)

{

temp = front;

num = front->data; delete (temp); front = rear = NULL; return num;

}

else

{

temp = front;

num = front->data; front = front->next; delete (temp); return num;

}

}

}

int main(void)

{

int num, option; queue q;

char ch = 'y'; front = NULL; rear = NULL;

while (ch == 'y')

{

cout << "\n 1.ADD"; cout << "\n 2.REMOVE"; cout << "\n 3.DISPLAY";

cout << "\n Enter an Option: "; cin >> option;

switch (option)

{

case 1:

cout << "Enter a Number to ADD: "; cin >> num;

q.add(num);

cout << "\n Do you want to Continue: "; break;

case 2:

num = q.remove(); cout << num;

cout << "\n Do you want to Continue: "; break;

case 3:

q.display();

cout << "\n Do you want to Continue: "; break;

}

ch = getch();

}

}