Solution

Q1:

#include<iostream> using namespace std; struct Order {

string dishName; int orderId; Order\* next;

};

class Restaurant {

private:

Order\* front;

Order\* rear;

public:

Restaurant() {

front=NULL; rear=NULL;

}

void enqueueOrder(string dishName, int orderId) { Order\* order=new Order();

order->dishName=dishName; order->orderId=orderId; order->next=NULL; if(rear==NULL) {

front=order; rear=order;

} else {

rear->next=order;

rear=order;

}

}

void dequeueOrder() {

if(front!=NULL) {

Order\* temp=front;

cout << "Order " << temp->orderId<< " for " <<temp-

>dishName<<" processed."<<endl;

front=front->next; delete temp;

if(front == NULL) {

rear=NULL;

}

} else {

cout<<"NO more order to process"<<endl;

}

}

void display() {

if (front == NULL) {

cout << "No orders in the queue."<<endl; return;

<< ")" << endl;

}

cout << "Orders in the queue:"<<endl; Order\* temp = front;

while (temp != NULL) {

cout << temp->dishName << " (Order ID: " << temp->orderId

temp = temp->next;

}

}

};

int main() {

Restaurant restaurant; restaurant.enqueueOrder("Pizza", 1);

restaurant.enqueueOrder("Burger", 2);

restaurant.enqueueOrder("Pasta", 3);

restaurant.display();

restaurant.dequeueOrder(); restaurant.dequeueOrder(); restaurant.dequeueOrder(); restaurant.dequeueOrder();

return 0;

}

Q2:

#include <iostream>

#include <queue>

#include <vector>

#include <sstream>

Using namespace std;

Int main() {

String input = “Data Structure and Algorithms”;

Vector<queue<char>> allQueues;

String word;

For (int I = 0; I < input.length(); i++) {

If (input[i] != ‘ ‘) {

Word += input[i]; // Add it to the current word

} else {

Queue<char> q; //

For (int j = 0; j < word.length(); j++) {

q.push(word[j]);

}

allQueues.push\_back(q);

word = “”;

}

}

Queue<char> q;

For (int j = 0; j < word.length(); j++) {

q.push(word[j]);

}

allQueues.push\_back(q);

Cout << “All Queues combined: “;

For (int I = 0; I < allQueues.size(); i++) {

Queue<char> q = allQueues[i];

While (!q.empty()) {

Cout << q.front() << “ “;

q.pop();

}

If (I != allQueues.size() – 1) {

Cout << “→ “;

}

}

Cout << endl;

Return 0;

}