

Nama : Muhammad Alvin Faa'iz

NIM : 103012400229

```
header.h x sources.cpp x main.cpp x
1 #ifndef HEADER_H_INCLUDED
2 #define HEADER_H_INCLUDED
3 #include <iostream>
4
5 using namespace std;
6
7 typedef struct elmqueue *address;
8
9 struct infotype {
10     string nama;
11     int usia;
12     string pekerjaan;
13     bool prioritas;
14     int nomorAntrean;
15 };
16
17 struct Queue{
18     address head;
19     address tail;
20 };
21
22 struct elmqueue{
23     infotype info;
24     address next;
25 };
26
27 void createQueue(Queue &Q);
28 bool isEmpty(Queue Q);
29 address allocate(string nama, int usia, string pekerjaan, int nomorAntrean);
30 address Front(Queue Q);
31 address Back(Queue Q);
32 void enqueue(Queue &Q, address p);
33 void dequeue(Queue &Q, address p);
34 int Size(Queue Q);
35 void printPatient(address p);
36 void serveQueue(Queue &Q);
37 void reassignQueue(Queue &Q);
38 void emergencyHandle(Queue &Q, int nomorAntrian);
39
40
41 #endif // HEADER_H_INCLUDED
42
```

```
header.h X sources.cpp X main.cpp X
1 #include "header.h"
2 #include <iostream>
3
4 using namespace std;
5
6 void createQueue(Queue &Q) {
7     Q.head = nullptr;
8     Q.tail = nullptr;
9 }
10
11 bool isEmpty(Queue Q) {
12     return (Q.head == nullptr) && (Q.tail == nullptr);
13 }
14
15 address allocate(string nama, int usia, string pekerjaan, int nomorAntrean) {
16     address p;
17     infotype x;
18
19     p = new elmqueue;
20     x.nama = nama;
21     x.usia = usia;
22     x.pekerjaan = pekerjaan;
23     x.prioritas = (usia >= 60) || pekerjaan == "tenaga_kesehatan";
24     x.nomorAntrean = nomorAntrean;
25     p->info = x;
26     p->next = nullptr;
27
28     return p;
29 }
30
31 address Front(Queue Q) {
32     return Q.head;
33 }
34
35 address Back(Queue Q) {
36     return Q.tail;
37 }
38
39 void enqueue(Queue &Q, address p) {
40     address temp;
41     bool inputPasienBukanPrioritas;
42     bool semuaPasienPrioritas;
43     bool tidakAdaPasienPrioritas;
44
45     if (!isEmpty(Q)) {
46         inputPasienBukanPrioritas = ! p->info.prioritas;
47         semuaPasienPrioritas = Q.tail->info.prioritas;
48         tidakAdaPasienPrioritas = ! Q.head->info.prioritas;
49     }
50
51     if (isEmpty(Q)) {
52         Q.head = p;
53         Q.tail = p;
54     } else if (inputPasienBukanPrioritas || semuaPasienPrioritas) {
55         Q.tail->next = p;
56         Q.tail = p;
57     } else if (tidakAdaPasienPrioritas) {
58         p->next = Q.head;
59         Q.head = p;
60     } else {
61         temp = Q.head;
62 }
```

```

62     while(temp->next != nullptr && temp->next->info.prioritas){
63         temp = temp->next;
64     }
65
66     p->next = temp->next;
67     temp->next = p;
68 }
69 }
70 }
71
72 void dequeue(Queue &Q, address p){
73 if(!isEmpty(Q)){
74     p = Q.head;
75     Q.head = p->next;
76
77 if(Q.head == nullptr){
78     Q.tail = nullptr;
79 }
80 else{
81     p = nullptr;
82 }
83 }
84
85 int Size(Queue Q){
86     int Count;
87     address p;
88
89     Count = 0;
90     p = Q.head;
91
92     while(p != nullptr){
93         Count = Count + 1;
94         p = p->next;
95     }
96
97     return Count;
98 }
99
100 void printPatient(address p){
101 if (p != nullptr) {
102     cout << "Nama: " << p->info.nama << endl;
103     cout << "Usia: " << p->info.usia << endl;
104     cout << "Pekerjaan: " << p->info.pekerjaan << endl;
105     cout << "Prioritas: " << p->info.prioritas << endl;
106     cout << "Nomor Antrian: " << p->info.nomorAntrean << endl;
107     cout << "Vaksinasi berhasil." << endl;
108 }
109 }
110
111 void serveQueue(Queue &Q){
112     int count = 0;
113     address p;
114
115     while (!isEmpty(Q) && count < 5) {
116         p = Q.head;
117         printPatient(p);
118         cout << "Vaksinasi berhasil." << endl << endl;
119         cout << "-----" << endl;
120         dequeue(Q, p);
121         count++;
122     }

```

```
123     if (count == 0)
124         cout << "Tidak ada warga dalam antrean." << endl;
125     }
126
127     void reassignQueue(Queue &Q) {
128         address p = Q.head;
129         while (p != nullptr) {
130             if (!p->info.prioritas) {
131                 p->info.prioritas = true;
132             }
133             p = p->next;
134         }
135         cout << "data prioritas pasien telah di update." << endl;
136     }
137
138     void emergencyHandle(Queue &Q, int nomorAntrean) {
139         if (isEmpty(Q)) return;
140
141         address prev = nullptr;
142         address curr = Q.head;
143
144
145         while (curr != nullptr && curr->info.nomorAntrean != nomorAntrean) {
146             prev = curr;
147             curr = curr->next;
148         }
149
150
151         if (curr == nullptr) {
152             cout << "Nomor antrean " << nomorAntrean << " tidak ditemukan." << endl;
153             return;
154         }
155
156         curr->info.prioritas = true;
157
158         if (curr == Q.head) return;
159
160         if (curr == Q.tail) {
161             Q.tail = prev;
162         }
163         if (prev != nullptr) {
164             prev->next = curr->next;
165         }
166
167         curr->next = Q.head;
168         Q.head = curr;
169
170         cout << "Pasien dengan nomor antrean " << endl;
171     }
172 }
```

```
header.h x sources.cpp x main.cpp x
1 #include <iostream>
2
3 #include "header.h"
4 using namespace std;
5
6 int main() {
7     Queue Q;
8     createQueue(Q);
9     address p;
10    infotype x;
11    int nomor, n, i, jml;
12    string nama, pekerjaan;
13    int usia, nomorAntrean;
14
15    cout << "masukkan banyak pasien yang ingin di input: ";
16    cin >> n;
17
18    for (int i = 1; i <= n; i++) {
19        string nama, pekerjaan;
20        int usia, nomorAntrean;
21
22        cout << "\npasien ke-" << i << endl;
23
24        cout << "nama: ";
25        cin >> nama;
26
27        cout << "usia: ";
28        cin >> usia;
29
30        cout << "pekerjaan: ";
31        cin >> pekerjaan;
32
33        cout << "nomor antrean: ";
34        cin >> nomorAntrean;
35
36        p = allocate(nama, usia, pekerjaan, nomorAntrean);
37        enqueue(Q, p);
38    }
39    cout << endl;
40    cout << "\nbanyak pasien dalam antrean: " << Size(Q) << endl;
41    cout << endl;
42
43    serveQueue(Q);
44    cout << endl;
45    cout << endl;
46
47    cout << "\nmasukkan nomor antrean yang ingin di prioritaskan: ";
48    cin >> nomor;
49    emergencyHandle(Q, nomor);
50    serveQueue(Q);
51
52    cout << endl;
53    cout << endl;
54
55    reassignQueue(Q);
56    serveQueue(Q);
57
58
59    return 0;
60}
61
```

```
masukkan banyak pasien yang ingin di input: 6

pasien ke-1
nama: andi
usia: 25
pekerjaan: guru
nomor antrean: 1

pasien ke-2
nama: bela
usia: 42
pekerjaan: tenaga_kesehatan
nomor antrean: 2

pasien ke-3
nama: citra
usia: 33
pekerjaan: mahasiswa
nomor antrean: 3

pasien ke-4
nama: doni
usia: 45
pekerjaan: petani
nomor antrean: 4

pasien ke-5
nama: eka
usia: 70
pekerjaan: pensiunan
nomor antrean: 2

pasien ke-6
nama: alvin
usia: 21
pekerjaan: mahasiswa
nomor antrean: 5

banyak pasien dalam antrean: 6
```

```
Nama: bela
Usia: 42
Pekerjaan: tenaga_kesehatan
Prioritas: 1
Nomor Antrian: 2
Vaksinasi berhasil.
Vaksinasi berhasil.
```

---

```
-----  
Nama: eka
Usia: 70
Pekerjaan: pensiunan
Prioritas: 1
Nomor Antrian: 2
Vaksinasi berhasil.
Vaksinasi berhasil.
```

---

```
-----  
Nama: andi
Usia: 25
Pekerjaan: guru
Prioritas: 0
Nomor Antrian: 1
Vaksinasi berhasil.
Vaksinasi berhasil.
```

---

```
-----  
Nama: citra
Usia: 33
Pekerjaan: mahasiswa
Prioritas: 0
Nomor Antrian: 3
Vaksinasi berhasil.
Vaksinasi berhasil.
```

---

```
-----  
Nama: doni
Usia: 45
Pekerjaan: petani
Prioritas: 0
Nomor Antrian: 4
Vaksinasi berhasil.
Vaksinasi berhasil.
```

---

```
-----  
masukkan nomor antrean yang ingin di prioritaskan: 4
Nomor antrean 4 tidak ditemukan.
Nama: alvin
Usia: 21
Pekerjaan: mahasiswa
Prioritas: 0
Nomor Antrian: 5
Vaksinasi berhasil.
Vaksinasi berhasil.
```

---

```
-----  
data prioritas pasien telah di update.
```