**NAME** : Tejesh Santosh Yewale

**ROLL NO. :** A-61

**PRACTICAL NO. E1**

**CODE:**

#include <iostream>

#include <queue>

using namespace std;

#define SIZE 10

// Define the patient structure

struct patient {

int patid;

int priority;

};

// Function to create a new patient record

patient\* record(int id, int prior) {

patient\* tmp = new patient;

tmp->patid = id;

tmp->priority = prior;

return tmp;

}

// Main driver code

int main() {

priority\_queue<int> pq; // Priority queue of priorities only

patient que[SIZE]; // Array to store patients

int ch;

int i = 0; // Patient count

int id;

int prior;

while (1) {

cout << "\nMenu\n1. Add patient\n2. Display\n3. Exit\n";

cin >> ch;

switch (ch) {

case 1: {

if (i >= SIZE) {

cout << "Queue full. Cannot add more patients.\n";

break;

}

cout << "\nEnter priority:\n3. H - Serious\n2. M - Not-Serious\n1. L - General\n";

cin >> prior;

if (prior < 1 || prior > 3) {

cout << "\nInvalid priority. Try again.\n";

break;

}

patient\* t = record(i + 1, prior);

que[i].patid = t->patid;

que[i].priority = t->priority;

pq.push(prior);

i++;

cout << "Patient added successfully.\n";

delete t; // clean up dynamic memory

break;

}

case 2: {

if (pq.empty()) {

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

break;

}

cout << "Priority Queue (Highest to Lowest): ";

// Display priorities without modifying actual queue

priority\_queue<int> temp = pq;

while (!temp.empty()) {

cout << temp.top() << ' ';

temp.pop();

}

cout << endl;

break;

}

case 3: {

exit(0);

}

default:

cout << "Invalid choice. Try again.\n";

}

}

return 0;

}

**OUTPUT:**

