#include <iostream>  
#include <string>  
  
using namespace std;  
  
struct Task {  
 string description;  
 int priority;  
};  
int partition(Task tasks[], int low, int high) {  
 int pivot = tasks[high].priority;  
 int i = low - 1;  
  
 for (int j = low; j < high; j++) {  
 if (tasks[j].priority < pivot) {  
 i++;  
 swap(tasks[i], tasks[j]);  
 }  
 }  
 swap(tasks[i + 1], tasks[high]);  
 return i + 1;  
}  
void quickSort(Task tasks[], int low, int high) {  
 if (low < high) {  
 int pi = partition(tasks, low, high);  
 quickSort(tasks, low, pi - 1);  
 quickSort(tasks, pi + 1, high);  
 }  
}  
  
void displayTasks(Task tasks[], int size) {  
 if (size == 0) {  
 cout << "No tasks available." << endl;  
 return;  
 }  
 cout << "\nTasks sorted by priority:" << endl;  
 for (int i = 0; i < size; i++) {  
 cout << i + 1 << ". [" << tasks[i].priority << "] " << tasks[i].description << endl;  
 }  
}  
  
int getValidPriority() {  
 int num;  
 while (true) {  
 cout << "Enter task priority (1-100, lower means higher priority): ";  
 cin >> num;  
 if (cin.fail() || num < 1 || num > 100) {  
 cout << "Invalid input! Please enter a number between 1 and 100.\n";  
 cin.clear();  
 while (cin.get() != '\n');  
 } else {  
 return num;  
 }  
 }  
}  
int main() {  
 Task tasks[100]; // Array to store tasks (max 100)  
 int size = 0;  
 int choice;  
  
 do {  
 cout << "\nTask Manager - Menu";  
 cout << "\n1. Add Task";  
 cout << "\n2. Sort Tasks by Priority (Quick Sort)";  
 cout << "\n3. Display Tasks";  
 cout << "\n4. Exit";  
 cout << "\nEnter your choice: ";  
 cin >> choice;  
  
 if (cin.fail() || choice < 1 || choice > 4) {  
 cout << "Invalid choice! Please enter a number between 1 and 4.\n";  
 cin.clear();  
 while (cin.get() != '\n');  
 continue;  
 }  
  
 switch (choice) {  
 case 1:  
 if (size >= 100) {  
 cout << "Task list is full!" << endl;  
 } else {  
 cin.ignore();  
 cout << "Enter task description: ";  
 getline(cin, tasks[size].description);  
 tasks[size].priority = getValidPriority();  
 size++;  
 }  
 break;  
  
 case 2:  
 if (size == 0) {  
 cout << "No tasks to sort!" << endl;  
 } else {  
 quickSort(tasks, 0, size - 1);  
 cout << "Tasks sorted successfully using Quick Sort!" << endl;  
 }  
 break;  
  
 case 3:  
 displayTasks(tasks, size);  
 break;  
  
 case 4:  
 cout << "Exiting Task Manager..." << endl;  
 break;  
 }  
 } while (choice != 4);  
  
 return 0;  
}