Emergency Alert Triggering System

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

BY

Student Name	**Regd. No**
V. Phani Sirisha	22501A05J2
P. Harshitha	22501A05E5
Y. Anusha	22501A05J7
V. Sumanth	22501A05I8

Under the Guidance of

Dr. K Jyothsna Devi, Ph.D.

Assistant Professor

PRASAD V POTLURI SIDDHARTHA INSTITUTE OF TECHNOLOGY

(Permanently affiliated to JNTU :: Kakinada, Approved by AICTE)

(An NBA & NAAC accredited and ISO 9001:2008 certified institution)

Kanuru, Vijayawada - 520007

(2023-2024)

PRASAD V POTLURI

SIDDHARTHA INSTITUTE OF TECHNOLOGY

(Permanently affiliated to JNTU :: Kakinada, Approved by AICTE)

(An NBA & NAAC accredited and ISO 9001:2008 certified institution)

Kanuru, Vijayawada – 520007

CERTIFICATE

This is to certify that the project report title **“Emergency Alert Triggering System“** is the bonafied work of **V. Phani Sirisha – 22501A05J2, P. Harshitha – 22501A05E5, Y. Anusha – 22501A05J7, V. Sumanth – 22501A05I8** in partial fulfilment of completing the Academic project in Advanced Data Structures during the academic year **2023-2024**.

Signature of the Incharge

Dr. K Jyothsna Devi

Asst. Professor, CSE Dept.

INDEX

Abstract:

Emergency Alert Triggering System - Everyday we see many Emergencies triggering all over the world. They are : Fire accidents, Road accidents, Health emergencies, Pregnancy or Labor pain emergencies, Brutal attacks etc. But what if all occur at once or many emergencies trigger at a time, will we be able to prioritise which is the foremost emergency and give it a red alert?

This project mainly focuses on how an organisation or an individual could easily **prioritise the emergencies** that they face or overcome in their daily life and take necessary precautions accordingly. Already we have emergency management systems to help individuals trigger an alert when they feel that they are unsafe like **DISHA in Andhra Pradesh, Emergency monitoring systems team in Telangana etc.** But this system is different from all the existing ones, it gives a view to the user to prioritise which problem is so terrible.

The emergencies that we have taken here are: **Fire accident, Road accident, Heart stroke, Pregnancy alert, Child abuse.** We had assigned a particular number to the respective emergencies so that the users can easily give the number rather than giving an emergency name. Here we included a **Threshold value** which is set by us according to the severity of the particular emergency which prevails in the society daily. The user gives the severity value too, If this value exceeds the threshold value then only the emergency triggering will be considered else it will be just taken and kept in the list. This helps to optimise the correct emergencies to be considered.

It even includes the option to **insert the emergencies, delete the emergencies and even search for the emergencies** which are taken into consideration. Along with that precautions are also provided for the emergencies at last which are triggered. To the disadvantages like the duplicate values we have taken another parameter like distance to check for least distance and give it the most priority. It even provides emergency numbers like Fire station number, Ambulance number etc. There are even the disadvantages to be solved - the user cannot know the severity rate, so questions should be asked by the system and accordingly the severity calculation is done and emergency threshold value by the user is noted.In case if the user is unable to answer the questions then definitely it is taken into consideration with the priority choosen. The value that is fixed by the system as threshold should also be changed timely according to the statistics of the emergencies in the society. These breakpoints can be implemented by **Machine Learning algorithms.**

Therefore, the user can give the emergencies he faces, modify them if he wants and get a list of prioritised emergencies as output. This helps a lot for not only **Government Departments but also Schools, Hospitals, Colleges, and also Individuals etc.**

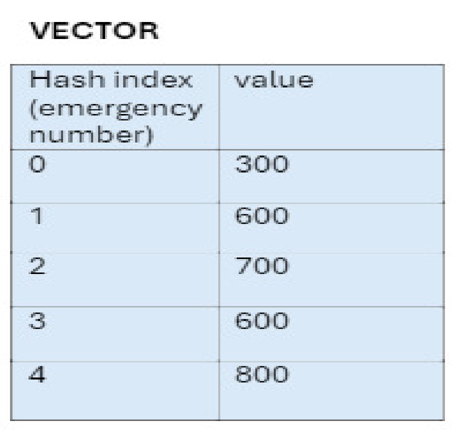
Introduction:

Life is a gift that should be cherished. It is a journey that is full of ups and downs, but it is a journey that is worth taking. Life is a time to learn and grow, to experience new things, and to make a difference in the world. It is a time to love and be loved, to laugh and to cry, and to make memories that will last a lifetime. Life is a precious gift, and it should be lived to the fullest. So, securing a life is very important and it is done by an emergency **alert triggering system**.

The **emergency alert triggering system** swiftly detects crises, leveraging IoT and AI. Sensor nodes monitor environmental parameters and human activities, transmitting data to a central control unit. Advanced algorithms analyse data, identifying threats and triggering alerts. Multi-channel alerts include SMS notifications, sirens, and social media broadcasts. This system **minimises response time, enhancing situational awareness** and facilitating coordinated emergency management efforts.

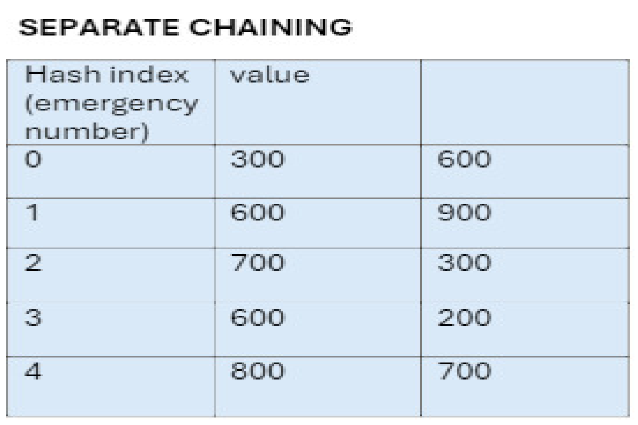
HASH TABLE IMPLEMENTATION:

A **Hash table** is defined as a data structure used to insert, look up, and remove key-value pairs quickly. It operates on the hashing concept, where each key is translated by a hash function into a distinct index in an array. So, the values given by the user are taken and stored in a hash table. Below shows the implementation using **Vectors**, but as we need many emergencies of the same type to be taken we use - **Vector of linked lists (Separate Chaining to avoid collisions). Difference** is shown below →



VECTOR

Hash index (emergency number)	value
0	300
1	600
2	700
3	600
4	800



SEPARATE CHAINING

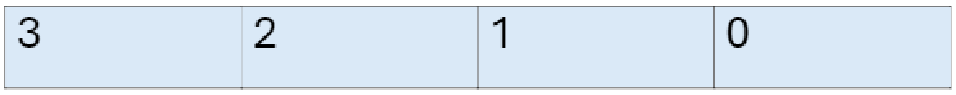
Hash index (emergency number)	value	
0	300	600
1	600	900
2	700	300
3	600	200
4	800	700

https://www.geeksforgeeks.org/hashing-data-structure/

PRIORITY QUEUE IMPLEMENTATION:

A **priority queue** is a type of queue that arranges elements based on their priority values. Elements with higher priority values are typically retrieved before elements with lower priority values. Only the emergencies which are crossing the threshold values are taken into account and put in the priority queue. As the priority is given from down, it will be stored in the descending order and we get the priority list as follows.

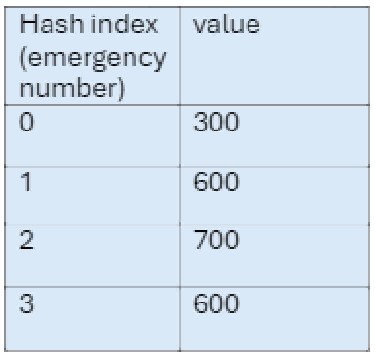
PRIORITY QUEUE



3	2	1	0

Result can be like:

Result:



Hash index (emergency number)	value
0	300
1	600
2	700
3	600

Precautions are also printed.

https://www.geeksforgeeks.org/priority-queue-set-1-introduction/

OVERALL IMPLEMENTATION:

TIME COMPLEXITY: O(n)

SPACE COMPLEXITY: O(n)

Objectives:

1. **Clear Options:**
   Users are presented with well-defined options for setting conditions that trigger alerts.Each option corresponds to a specific emergency scenario (e.g., temperature threshold, motion detection, gas levels).
2. **User-Friendly Interface:**
   The system's interface guides users through the process.Descriptive labels and tooltips ensure users understand the purpose of each trigger option.
3. **Avoiding Confusion:**
   We prioritise simplicity over complexity.Users should not feel overwhelmed by technical jargon or intricate settings.Clear documentation and tooltips help users make informed choices.
4. **Testing and Validation:**
   During development, we rigorously test the alert triggering options.User feedback and usability testing play a crucial role in refining the system.

Scope of the Object:

1. **Hospitals and Healthcare Facilities:**
   Hospitals operate 24/7, handling emergencies, patient care, and critical procedures.The system ensures swift responses to incidents like fire outbreaks, gas leaks, or unauthorised access.Healthcare professionals can focus on patient well-being, knowing that alerts will promptly notify them of any threats.
2. **Fire Prevention and Safety:**
   Sensors monitor temperature, smoke levels, and gas concentrations.When thresholds are breached, alerts trigger evacuation protocols.Simulation mode allows fire drills and training.
3. **Industrial Plants and Laboratories:**
   Chemical plants, research labs, and manufacturing facilities handle hazardous materials. The system mitigates risks by monitoring volatile conditions.
   Alerts prevent accidents and protect workers.

Software Used:

Programming Language: C++ is used mainly because of its efficiency,compatibility and flexibility. It also offers a variety of templates as a part of its Standard Template Library,which provides data structures like priority queues. Leveraging these standard libraries can accelerate development and ensure code reliability.

Integrated Development Environment (IDE) : We utilised VSCODE as our primary Integrated Development Environment (IDE) for C++ programming. VSCODE provided us with a user-friendly interface and a comprehensive set of tools for writing, compiling, and debugging our code. Its cross-platform support allowed our team members to work seamlessly across different operating systems.

Concept Implemented : Priority Queue, Hashtable

Compiler Used : GNU Compiler Collection (GCC)

Version Control System (VCS):

Git is utilised as the Version Control System (VCS) to manage the project's codebase. Git enables tracking of code changes, collaboration among developers, and maintaining a comprehensive history of the project's evolution.

Proposed Model for Emergency Triggering System:

Overview: The emergency triggering system aims to provide swift and effective alerts in critical situations.

User Interface: Intuitive interface with clear options aligned with emergency scenarios.
Descriptive labels and tooltips for easy navigation and understanding.

Alert Triggers: Options include temperature thresholds, motion detection, and other relevant parameters.Each trigger is well-defined to cater to various emergency situations.

Documentation and Guidance: Comprehensive documentation empowers users to make informed decisions.Detailed explanations alongside trigger options enhance understanding.

Testing and Validation: Rigorous testing during development ensures effectiveness and usability.User feedback and usability testing aid in refining the system and addressing any issues.

Continuous Improvement: Continuous monitoring and updates to enhance clarity, simplicity, and user-friendliness.Aim to empower users to efficiently set up alerts tailored to their specific requirements.

Integration and Scalability: Seamless integration with existing emergency response systems.Scalable architecture to accommodate future enhancements and expansions.

This proposed model prioritises simplicity, clarity, and effectiveness, aiming to enhance emergency preparedness and response.

Sample Code:

```
#include <iostream>

#include <vector>

#include <list>

#include <queue> // Include priority_queue header

using namespace std;


int hash_function(int e_num) {

   // Using the division method with 5 as no collisions occur here

   int hash_key = e_num % 5;

   return hash_key;

}

struct Compare {

   bool operator()(const pair<pair<int, int>, int>& a, const pair<pair<int, int>, int>& b) {

      if (a.second != b.second) {

         // Prioritise by emergency number first

         return a.second < b.second; // Higher emergency number means higher priority

      } else if (a.first.first != b.first.first) {

         // If emergency numbers are equal, prioritise by severity

         return a.first.first < b.first.first; // Higher severity means higher priority

      } else {

         // If both emergency numbers and severities are equal, prioritise by distance

         return a.first.second > b.first.second; // Lesser distance means higher priority

      }
```

```
        }

    };

    void hashtable(int hash_key, vector<list<pair<int, int>>>& table,
priority_queue<pair<pair<int, int>, int>, vector<pair<pair<int, int>, int>>, Compare>& pq,
int value, int distance) {

        table[hash_key].push_back({value, distance});

        pq.push({{value, distance}, hash_key});

    }



    int main() {

        cout << "This is the Emergency Alert Triggering System" << endl;

        cout << "The following emergencies are taken into consideration:" << endl;

        cout << "NUMBER| EMERGENCY NAME| THRESHOLD_VALUE|" << endl;

        cout << "  0     FIRE ACCIDENT           300" << endl;

        cout << "  1     ROAD ACCIDENT           200" << endl;

        cout << "  2     HEART STROKE            100" << endl;

        cout << "  3     PREGNANCY ALERT         50" << endl;

        cout << "  4     CHILD ABUSE             25" << endl;

        vector<int> v={300,200,100,50,25};


        // Take the table with size 5 as there are 5 emergencies

        vector<list<pair<int, int>>> table(5);

         priority_queue<pair<pair<int, int>, int>, vector<pair<pair<int, int>, int>>, Compare>
pq; // Priority queue to store emergencies



        // Loop for asking the user to enter the emergencies encountered
```

```
cout << "Type -1 if your emergencies are over, else type 1 to continue." << endl;

int choice, e_num, value, hash_key, distance;

cin >> choice;


if (choice == -1);

else {

   while (choice != -1) {

      cout << "Enter the number which corresponds to your emergency." << endl;

      cin >> e_num;

      if (e_num >= 5)

         cout << "It exceeded the size of hashtable" << endl;

      else {

            cout << "Enter the value; here it means the severity of the emergency. Give the
value according to the dangerous/hazardous situations proning currently." << endl;

         cin >> value;

         cout << "Enter the distance for this emergency:" << endl;

         cin >> distance;

         hash_key = hash_function(e_num);

         if ( v[e_num] < value ) {

            hashtable(hash_key, table, pq, value, distance);

         }

      }


      cout << "Give your choice:" << endl;

      cin >> choice;
```

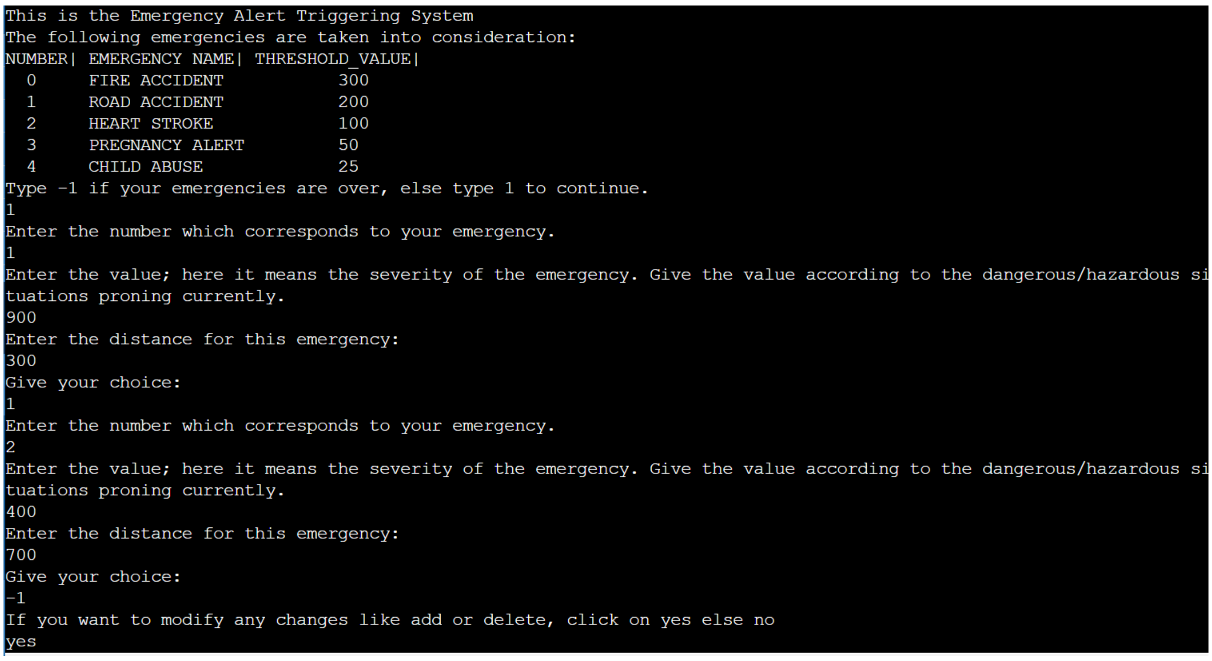
```
    }

  }

  string s;

   cout << "If you want to modify any changes like add or delete, click on yes else no" <<
endl;

  cin >> s;


  if (s == "no");

  else {

    int c = 1;

    while (c) {

      cout << "Click 1 to insert emergency:" << endl;

      cout << "Click 2 to delete emergency:" << endl;

      cout << "Click 3 to search emergency:" << endl;


      int option;

      cin >> option;

      switch (option) {

        case 1:

          cout << "Enter the number which corresponds to your emergency." << endl;

          cin >> e_num;

          if (e_num >= 5)

            cout << "It exceeded the size of hashtable" << endl;

          else {
                cout << "Enter the value; here it means the severity of the emergency. Give the value according to the dangerous/hazardous situations proning currently." << endl;

                cin >> value;

                cout << "Enter the distance for this emergency:" << endl;

                cin >> distance;

                hash_key = hash_function(e_num);

                hashtable(hash_key, table, pq, value, distance);

            }


            break;

        case 2:

                cout << "Enter the number which corresponds to your emergency to be deleted." << endl;

            cin >> e_num;

            // Assuming you want to clear the list for that index

            table[e_num].clear();

            break;

        case 3:

                cout << "Enter the number which corresponds to your emergency to be searched." << endl;

            cin >> e_num;

            cout << e_num << " ";

            for (pair<int, int> val : table[e_num]) {

                cout << "Severity: " << val.first << ", Distance: " << val.second << endl;

            }

            cout << endl;
```

```
                    break;

                default:

                    cout << "Invalid entry!! Please give it correctly." << endl;

            }

            cout << "If you want to continue, press 1 else 0. Note that only 5 slots are available for insertion." << endl;

            cin >> c;

        }

    }


    cout << "Final hashtable emergencies:" << endl;

    for (int i = 0; i < 5; i++) {

        cout << i << " ";

        for (pair<int, int> val : table[i]) {

            cout << "  Severity: " << val.first << ", Distance: " << val.second << endl;

        }

        cout << endl;

    }

    cout << "Priority Queue of Emergencies (Descending Order of Severity):" << endl;

    while (!pq.empty()) {

        pair<pair<int, int>, int> top = pq.top();

        pq.pop();

        cout << "Emergency Number: " << top.second << ", Severity: " << top.first.first << ", Distance: " << top.first.second << endl;

        cout << "Precautions to be taken:" << endl;
```

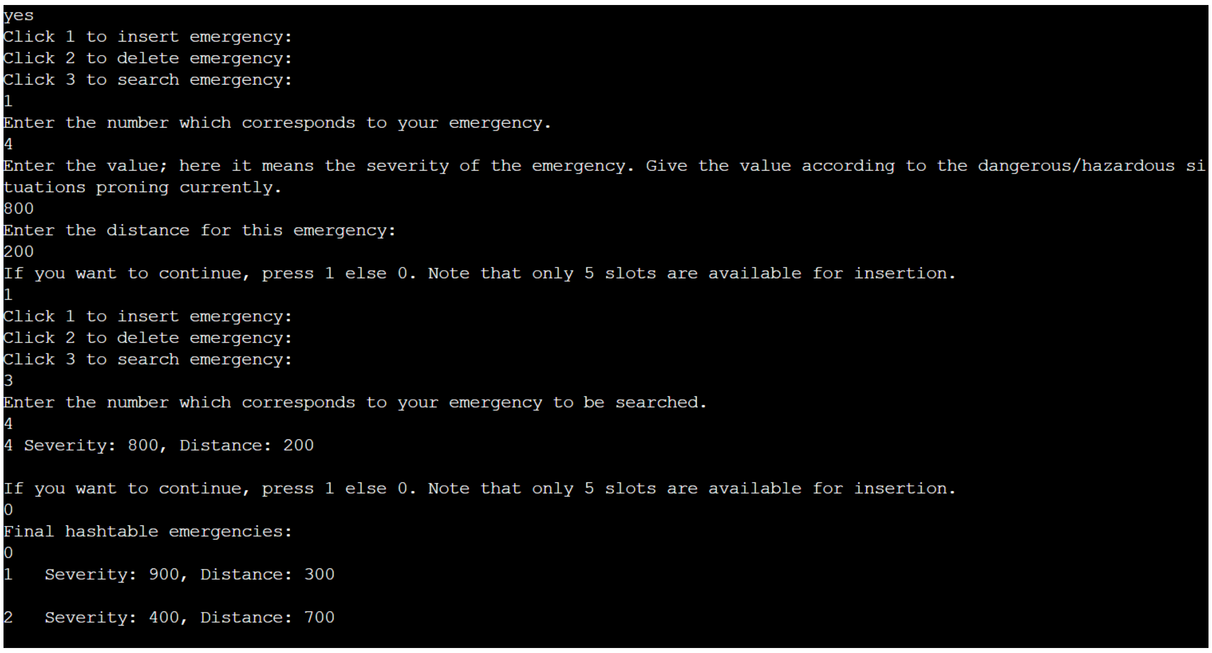
```
if(top.second==0)
{
    cout<<"Evacuate the area immediately and call emergency services."<<endl;
    cout<<"Use fire extinguishers if safe to do so and if trained."<<endl;
    cout<<"Stay low to the ground if smoke is present to minimise inhalation."<<endl;
    cout<<"Call : 101"<<endl;
}
else if(top.second==1)
{
    cout<<"Assess the scene for safety hazards and ensure personal safety."<<endl;
    cout<<"Call emergency services and provide accurate location details."<<endl;
    cout<<"Administer basic first aid if trained, while awaiting medical help."<<endl;
    cout<<"Clear the area of bystanders if possible to prevent further accidents."<<endl;
    cout<<"Call : 108"<<endl;
}
  else if(top.second==2)
{
    cout<<"Recognize symptoms such as chest pain, shortness of breath, and dizziness."<<endl;
    cout<<"Call emergency services immediately and provide necessary medical history."<<endl;
    cout<<"Assist the individual to sit or lie down comfortably while awaiting help."<<endl;
    cout<<"Administer CPR if trained and necessary, following current guidelines."<<endl;
    cout<<"Call : 108"<<endl;
```

```
            }
                else if(top.second==3)
            {
               cout<<"Stay calm and reassure the pregnant individual."<<endl;
                     cout<<"Assist them to a comfortable position, preferably lying on their left side."<<endl;
                     cout<<"Assist the individual to sit or lie down comfortably while awaiting help."<<endl;
                cout<<"Monitor vital signs and provide reassurance until help arrive"<<endl;
                cout<<"Call : 108"<<endl;
            }
                else if(top.second==4)
            {
               cout<<"Ensure the safety of the child by removing them from harm's way."<<endl;
                      cout<<"Document any visible injuries or signs of abuse discreetly and accurately."<<endl;
               cout<<"Report the incident to the appropriate authorities immediately."<<endl;
                   cout<<"Provide emotional support to the child and reassure them that they are safe."<<endl;
                cout<<"Call : 112"<<endl;
            }
        }
        return 0;
    }
```

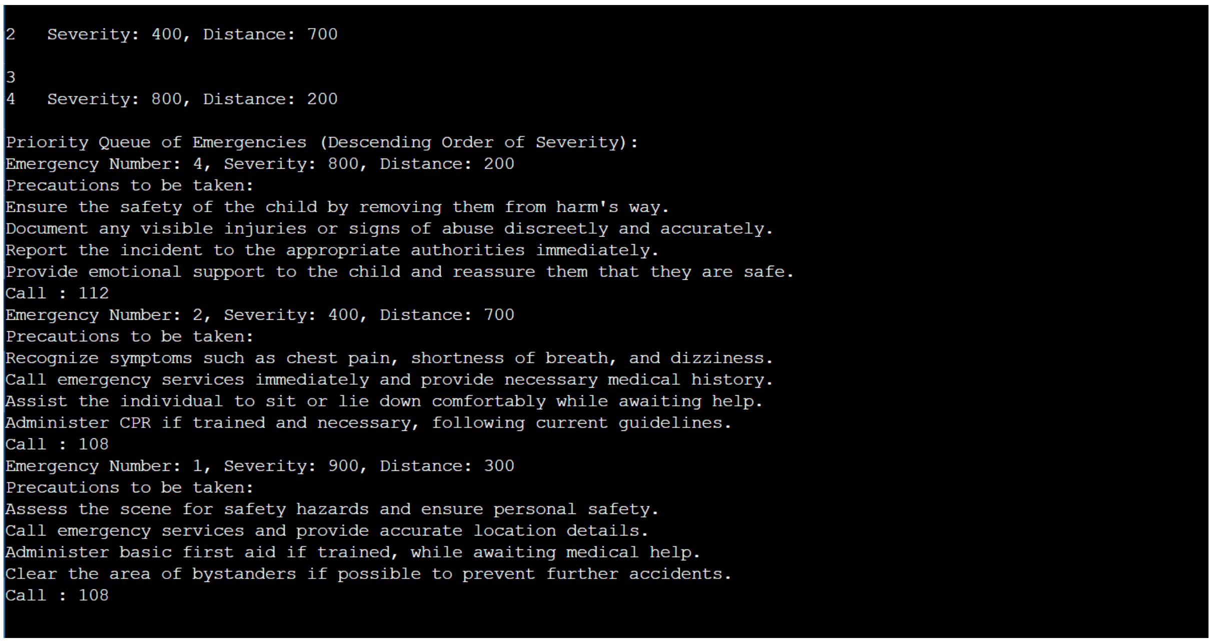
Result/Output Screenshots:



```
This is the Emergency Alert Triggering System
The following emergencies are taken into consideration:
NUMBER| EMERGENCY NAME| THRESHOLD_VALUE|
  0     FIRE ACCIDENT           300
  1     ROAD ACCIDENT           200
  2     HEART STROKE            100
  3     PREGNANCY ALERT         50
  4     CHILD ABUSE             25
Type -1 if your emergencies are over, else type 1 to continue.
1
Enter the number which corresponds to your emergency.
1
Enter the value; here it means the severity of the emergency. Give the value according to the dangerous/hazardous si
tuations proning currently.
900
Enter the distance for this emergency:
300
Give your choice:
1
Enter the number which corresponds to your emergency.
2
Enter the value; here it means the severity of the emergency. Give the value according to the dangerous/hazardous si
tuations proning currently.
400
Enter the distance for this emergency:
700
Give your choice:
-1
If you want to modify any changes like add or delete, click on yes else no
yes
```



```
yes
Click 1 to insert emergency:
Click 2 to delete emergency:
Click 3 to search emergency:
1
Enter the number which corresponds to your emergency.
4
Enter the value; here it means the severity of the emergency. Give the value according to the dangerous/hazardous si
tuations proning currently.
800
Enter the distance for this emergency:
200
If you want to continue, press 1 else 0. Note that only 5 slots are available for insertion.
1
Click 1 to insert emergency:
Click 2 to delete emergency:
Click 3 to search emergency:
3
Enter the number which corresponds to your emergency to be searched.
4
4 Severity: 800, Distance: 200

If you want to continue, press 1 else 0. Note that only 5 slots are available for insertion.
0
Final hashtable emergencies:
0
1   Severity: 900, Distance: 300

2   Severity: 400, Distance: 700
```


```
2   Severity: 400, Distance: 700

3
4   Severity: 800, Distance: 200

Priority Queue of Emergencies (Descending Order of Severity):
Emergency Number: 4, Severity: 800, Distance: 200
Precautions to be taken:
Ensure the safety of the child by removing them from harm's way.
Document any visible injuries or signs of abuse discreetly and accurately.
Report the incident to the appropriate authorities immediately.
Provide emotional support to the child and reassure them that they are safe.
Call : 112
Emergency Number: 2, Severity: 400, Distance: 700
Precautions to be taken:
Recognize symptoms such as chest pain, shortness of breath, and dizziness.
Call emergency services immediately and provide necessary medical history.
Assist the individual to sit or lie down comfortably while awaiting help.
Administer CPR if trained and necessary, following current guidelines.
Call : 108
Emergency Number: 1, Severity: 900, Distance: 300
Precautions to be taken:
Assess the scene for safety hazards and ensure personal safety.
Call emergency services and provide accurate location details.
Administer basic first aid if trained, while awaiting medical help.
Clear the area of bystanders if possible to prevent further accidents.
Call : 108
```

Conclusion:

In **conclusion, the Emergency Alert Triggering System** stands as a vigilant guardian, ever-watchful and swift in its response to emergent threats. Through its capacity to detect critical conditions and issue timely alerts, this system plays a pivotal role in empowering individuals, organisations, and entire communities to take proactive measures in the face of adversity. Whether deployed in hospitals to ensure patient safety, in industrial plants to mitigate risks, or in educational institutions to enhance emergency preparedness, the system's ability to deliver early warnings is paramount in minimising the impact of emergencies and preserving lives.

In embracing this technology, we acknowledge its transformative potential not only to inform but also to protect. Every second gained through early alerting mechanisms is a precious opportunity to enact life-saving measures and mitigate the consequences of disasters. As we continue to refine and innovate upon these systems, let us remain steadfast in our commitment to leveraging technology for the greater good, ensuring that no warning goes unheard and no life is needlessly endangered. For in the realm of emergency response, each passing moment holds the potential to make a profound difference, reaffirming the importance of prioritising preparedness, resilience, and the preservation of human life above all else. In the future, soon we are going to improve it as an **WEB APPLICATION.**

References (web site URLs):

FEMA (Federal Emergency Management Agency): https://www.fema.gov/

International Association of Emergency Managers (IAEM): https://www.iaem.org/

National Emergency Management Association (NEMA): https://www.nemaweb.org/