

## CSE-GitHub project

### Women safety alert system



### Technical expectations

- 1) woman alert system using Greedy algorithm  
pseudocode

START

Create a list DISTANCES of police stations from the woman

Set MIN = first value in DISTANCES

FOR each distance D in DISTANCES:

    IF D < MIN:

        MIN = D

    ENDIF

ENDFOR

PRINT "Woman should go to the police station", MIN, "meters away."

END

### Algorithm

Step 1: Start.

Step 2: Create a list of distances of all nearby police stations from the woman.

Step 3: Assume the first distance as the minimum (MIN).

Step 4: Repeat for every distance in the list:

- If the current distance is smaller than MIN, update MIN with this distance.

Step 5: After checking all distances, MIN will store the shortest distance.

Step 6: Display that the woman should go to the police station at MIN meters.

Step 7: Stop.

### Code

```
[1]: # Distances of police stations from the woman (in meters)
    police_stations = [500, 200, 150, 170, 400]

    # Greedy choice: choose the minimum distance
    min_distance = police_stations[0]
    for d in police_stations:
        if d < min_distance:
            min_distance = d

    print("The woman should run to the police station", min_distance, "meters away for quickest help.")

The woman should run to the police station 150 meters away for quickest help.
```

2) woman alert system using A\* algorithm  
pseudocode

START

Initialize OPEN list with the start node (Woman)

Initialize CLOSED list as empty

Set  $g(\text{start}) = 0$

Set  $f(\text{start}) = g(\text{start}) + h(\text{start})$

WHILE OPEN list is not empty:

    Select CURRENT node from OPEN with smallest f value

    IF CURRENT is Police:

        Goal reached → break the loop

    Remove CURRENT from OPEN

    Add CURRENT to CLOSED

    FOR each NEIGHBOUR of CURRENT:

        IF NEIGHBOUR is in CLOSED:

            Skip to next neighbour

        Tentative\_G =  $g(\text{CURRENT}) + \text{cost}(\text{CURRENT}, \text{NEIGHBOUR})$

        IF NEIGHBOUR not in OPEN:

            Add NEIGHBOUR to OPEN

        ELSE IF Tentative\_G  $\geq g(\text{NEIGHBOUR})$ :

            Continue to next neighbour

        Set parent(NEIGHBOUR) = CURRENT

        Set  $g(\text{NEIGHBOUR}) = \text{Tentative\_G}$

        Set  $f(\text{NEIGHBOUR}) = g(\text{NEIGHBOUR}) + h(\text{NEIGHBOUR})$

ENDWHILE

Reconstruct the path from Police back to Woman using parent pointers

PRINT "Shortest Path Found"

END

Algorithm

Step 1: Start.

Step 2: Initialize an OPEN list and insert the start node (Woman).

Step 3: Initialize an empty CLOSED list.

Step 4: Set the cost  $g(\text{start}) = 0$ .

Step 5: Calculate  $f(\text{start}) = g(\text{start}) + h(\text{start})$  using the heuristic.

Step 6: Repeat while OPEN list is not empty:

- a) From OPEN list, select the node having the smallest f value. Call it CURRENT.
- b) If CURRENT is Police, stop — goal found.
- c) Remove CURRENT from OPEN and add it to CLOSED.
- d) For each neighbour of CURRENT:
  - If neighbour is in CLOSED, skip it.
  - Compute  $\text{temp\_g} = g(\text{CURRENT}) + \text{cost}(\text{CURRENT}, \text{neighbour})$
  - If neighbour is not in OPEN, add it.
  - Else if  $\text{temp\_g}$  is greater than or equal to  $g(\text{neighbour})$ , skip it.
  - Otherwise update:
    - Set  $\text{parent}(\text{neighbour}) = \text{CURRENT}$
    - Update  $g(\text{neighbour})$
    - Compute  $f(\text{neighbour}) = g(\text{neighbour}) + h(\text{neighbour})$

Step 7: When Police node is reached, reconstruct the path by following parent pointers from Police back to Woman.

Step 8: Output the shortest path and total distance.

Step 9: End.

### Code

```
[1]: # Graph distances (cost)
graph = {
    'Woman': {'Shop': 2, 'Crowd': 1},
    'Shop': {'Road': 2},
    'Crowd': {'Park': 4},
    'Road': {'Police': 1},
    'Park': {'Police': 2},
    'Police': {}
}

# Heuristic guess (straight-line distance to Police)
h = {
    'Woman': 4,
    'Shop': 2,
    'Crowd': 3,
    'Road': 1,
    'Park': 2,
    'Police': 0
}

start = 'Woman'
goal = 'Police'
open_set = [start]
g = {start: 0}
came = []

while open_set:
    # find node with smallest f = g + h
    current = open_set[0]
    for node in open_set:
        if g.get(node, 999) + h[node] < g.get(current, 999) + h[current]:
            current = node
    if current == goal:
        break
```

```
    if current == goal:
        break
    open_set.remove(current)
    for neigh in graph[current]:
        temp = g[current] + graph[current][neigh]
        if temp < g.get(neigh, 999):
            g[neigh] = temp
            came[neigh] = current
            if neigh not in open_set:
                open_set.append(neigh)

# path reconstruction
path = [goal]
while path[-1] != start:
    path.append(came[path[-1]])
path.reverse()
print("Shortest path:", path)
print("Total distance:", g[goal])
print("Woman reached the police and informed them about the danger.")
```

```
Shortest path: ['Woman', 'Shop', 'Road', 'Police']
Total distance: 5
Woman reached the police and informed them about the danger.
```

3)Using the concept of exchanging values for woman security

### Pseudocode

```
START
Set WOMAN_MESSAGE = "Help Needed"
Set POLICE_MESSAGE = "On the Way"
PRINT both messages before exchange
Set TEMP = WOMAN_MESSAGE
Set WOMAN_MESSAGE = POLICE_MESSAGE
Set POLICE_MESSAGE = TEMP
PRINT both messages after exchange
END
```

### Algorithm

Step 1: Start  
Step 2: Assign the woman's message as "Help Needed".  
Step 3: Assign the police message as "On the Way".  
Step 4: Display both messages before swapping.  
Step 5: Store the woman's message in a temporary variable TEMP.  
Step 6: Assign the police message to WOMAN\_MESSAGE.  
Step 7: Assign TEMP (old woman message) to POLICE\_MESSAGE.  
Step 8: Display both messages after swapping.  
Step 9: Stop.

### Code

```
1: # Woman wants to send a help alert to Police
   woman_message = "Help Needed"
   police_message = "On the Way"

   print("Before exchange:")
   print("Woman:", woman_message)
   print("Police:", police_message)

   # Exchange (swap) the values
   temp = woman_message
   woman_message = police_message
   police_message = temp

   print("\nAfter exchange:")
   print("Woman received:", woman_message)
   print("Police received:", police_message)

Before exchange:
Woman: Help Needed
Police: On the Way

After exchange:
Woman received: On the Way
Police received: Help Needed
```

### 4)Using the concept of Base conversion

#### Pseudocode

```
START
Set SECRET_CHAR = '7'
PRINT "Woman sent:", SECRET_CHAR
Compute ALERT_LEVEL = ASCII(SECRET_CHAR) - ASCII('0')
PRINT "Police converted to number:", ALERT_LEVEL
IF ALERT_LEVEL > 5 THEN
    PRINT "High alert! Police rushing."
```

```

ELSE
    PRINT "Low alert. Patrol sent."
ENDIF
END

```

### Algorithm

Step 1: Start.

Step 2: Take the woman's secret alert code as a character (e.g., '7').

Step 3: Display the character.

Step 4: Convert the character into a number using:  
 $\text{ASCII}(\text{character}) - \text{ASCII}('0')$

Step 5: Store the result in ALERT\_LEVEL.

Step 6: Display the converted number.

Step 7: If ALERT\_LEVEL > 5, show High Alert message.

Step 8: Else show Low Alert message.

Step 9: Stop.

### Code

```

[4]: # Woman sends a secret character code to the police
secret_char = '7' # Woman's alert level as a character

print("Woman sent character code:", secret_char)

# Base conversion: character to number
# ord('0') = 48, ord('7') = 55 → 55 - 48 = 7
alert_level = ord(secret_char) - ord('0')

print("Police converted it to number:", alert_level)

# Police reacts based on number
if alert_level > 5:
    print("Police: High alert! Heading to help the woman.")
else:
    print("Police: Low alert. Sending nearest patrol team.")

Woman sent character code: 7
Police converted it to number: 7
Police: High alert! Heading to help the woman.

```

5) using if-elif conditional statements

### Pseudocode

```

START
Set SECRET_CHAR = '4'
PRINT the secret character
Convert ALERT = ASCII(SECRET_CHAR) - ASCII('0')
PRINT the alert level
IF ALERT >= 8 THEN
    PRINT "Extreme danger! Immediate action."
ELSE IF ALERT >= 5 THEN
    PRINT "High alert. Police moving."
ELSE IF ALERT >= 3 THEN
    PRINT "Moderate alert. Patrol alerted."
ELSE
    PRINT "Low alert. Situation monitored."

```

ENDIF  
END

### Algorithm

- Step 1: Start.
- Step 2: Take the secret alert character from the woman.
- Step 3: Display the character.
- Step 4: Convert the character to number:  
 $\text{alert} = \text{ASCII}(\text{character}) - \text{ASCII}('0')$
- Step 5: Display the converted alert value.
- Step 6: If  $\text{alert} \geq 8$ , show Extreme danger message.
- Step 7: Else if  $\text{alert} \geq 5$ , show High alert message.
- Step 8: Else if  $\text{alert} \geq 3$ , show Moderate alert message.
- Step 9: Else, show Low alert.
- Step 10: Stop.

### Code

```
[5]: # Woman sends a secret character alert code
secret_char = '4' # example character

print("Woman sent code:", secret_char)

# Convert character to number
alert = ord(secret_char) - ord('0')

print("Converted alert level:", alert)

# IF - ELIF - ELSE conditions
if alert >= 8:
    print("Police: Extreme danger! Team dispatched immediately!")
elif alert >= 5:
    print("Police: High alert! Moving towards the woman.")
elif alert >= 3:
    print("Police: Moderate alert. Nearby patrol alerted.")
else:
    print("Police: Low alert. Monitoring the situation.")

Woman sent code: 4
Converted alert level: 4
Police: Moderate alert. Nearby patrol alerted.
```

6) using the concept of while loop

### Pseudocode

```
START
LOOP Forever:
    INPUT SECRET_CHAR
    ALERT = ASCII(SECRET_CHAR) - ASCII('0')
    PRINT ALERT
    IF ALERT >= 8 THEN
        PRINT "Extreme danger"
        BREAK the loop
    ELSE IF ALERT >= 5 THEN
        PRINT "High alert"
        BREAK the loop
    ELSE IF ALERT >= 3 THEN
        PRINT "Moderate alert"
    ELSE
        PRINT "Low alert. Ask woman to send again"
    ENDIF
ENDLOOP
```

END

### Algorithm

- Step 1: Start.
- Step 2: Begin a continuous loop using while.
- Step 3: Take a character input from the woman.
- Step 4: Convert character to number using:  
alert ASCII (character)  
ASCII('0')
- Step 5: If alert  $\geq 8$ :  
Print "Extreme danger" and break.
- Step 6: Else if alert  $\geq 5$ :  
Print "High alert" and break.
- Step 7: Else if alert  $\geq 3$ :  
Print "Moderate alert".
- Step 8: Else:  
Print "Low alert, woman can send again".
- Step 9: Loop continues until a break happens.
- Step 10: Stop.

### Code

```
[?]: # Woman keeps sending alert codes until police get a high alert

while True:
    secret_char = input("Woman sends alert code (0-9): ")

    # Prevent empty input
    if secret_char == "":
        print("Error: You must enter a number from 0 to 9.")
        continue

    # Take only the first character
    secret_char = secret_char[0]

    # If not a digit, show error
    if not ('0' <= secret_char <= '9'):
        print("Invalid input! Please enter digits only (0-9).")
        continue

    # Character to number conversion
    alert = ord(secret_char) - ord('0')
    print("Converted alert level:", alert)

    # Conditions
    if alert >= 8:
        print("Police: Extreme danger! Rushing immediately!")
        break
    elif alert >= 5:
        print("Police: High alert! Police moving towards the woman.")
        break
    elif alert >= 3:
        print("Police: Moderate alert. Asking woman to stay calm.")
    else:
        print("Police: Low alert. Monitoring. Woman can send again.")

    print() # blank line for clarity
```

Woman sends alert code (0-9):  
Error: You must enter a number from 0 to 9.  
Woman sends alert code (0-9): 5  
Converted alert level: 5  
Police: High alert! Police moving towards the woman.

### 7) Using GCD Pseudocode

DISPLAY start message  
WHILE True

```

READ a
IF a is empty OR non-digit
    DISPLAY error
    CONTINUE
IF a = 0
    DISPLAY exit
    BREAK
READ b
IF b is empty OR non-digit
    DISPLAY error
    CONTINUE
IF b = 0
    DISPLAY exit
    BREAK
SET x = a
SET y = b
WHILE x != y
    IF x > y
        x = x - y
    ELSE
        y = y - x
SET gcd = x
DISPLAY gcd
IF gcd >= 10
    DISPLAY high alert
    BREAK
ELSE IF gcd >= 5
    DISPLAY moderate alert
ELSE
    DISPLAY low alert
END WHILE

```

#### Algorithm

Step 1: Start  
 Step 2: Display the project title  
 Step 3: Enter a while loop (infinite loop)  
 Step 4: Take first number as input  
 Step 5: Validate input (not empty, digits only)  
 Step 6: If input = 0, stop program  
 Step 7: Take second number with same validations  
 Step 8: If input = 0, stop program  
 Step 9: Assign  $x = a$  and  $y = b$   
 Step 10: Repeat until both values become equal:  
 If  $x > y \rightarrow x = x - y$   
 Else  $\rightarrow y = y - x$   
 Step 11: When equal,  $x$  is the GCD  
 Step 12: Display GCD  
 Step 13:  
 If  $GCD \geq 10 \rightarrow$  High Alert  $\rightarrow$  break

Else if  $GCD \geq 5 \rightarrow$  Moderate Alert

Else  $\rightarrow$  Low Alert

Step 14: Loop continues

Step 15: Stop

### Code

```
[8]: # Woman Safety Alert + GCD Finder
print("WOMAN SAFETY SYSTEM - GCD CALCULATOR")
print("Enter two positive numbers to calculate GCD.")
print("Enter 0 anytime to stop.\n")
while True:
    a = input("Enter first number: ")
    # Prevent empty input
    if a == "":
        print("Error: Input cannot be empty.")
        continue
    # Convert to int safely
    if not a.isdigit():
        print("Please enter digits only.")
        continue
    a = int(a)
    if a == 0:
        print("Exiting system... Stay safe!")
        break
    b = input("Enter second number: ")
    if b == "":
        print("Error: Input cannot be empty.")
        continue
    if not b.isdigit():
        print("Please enter digits only.")
        continue
    b = int(b)
    if b == 0:
        print("Exiting system... Stay safe!")
        break
    # Calculate GCD using repeated subtraction method
    x, y = a, b
    while x != y:
        if x > y:
            x = x - y
        else:
            if x > y:
                x = x - y
            else:
                y = y - x
    gcd = x
    print("GCD of", a, "and", b, "is:", gcd)
    # Police alert based on GCD result
    if gcd >= 10:
        print("Police: High Alert Triggered based on GCD result!")
        break
    elif gcd >= 5:
        print("Police: Moderate Alert. Monitoring the situation.\n")
    else:
        print("Police: Low Alert. Woman may continue sending values.\n")

WOMAN SAFETY SYSTEM - GCD CALCULATOR
Enter two positive numbers to calculate GCD.
Enter 0 anytime to stop.

Enter first number: 5
Enter second number: 8
GCD of 5 and 8 is: 1
Police: Low Alert. Woman may continue sending values.

Enter first number: 6
Enter second number: 4
GCD of 6 and 4 is: 2
Police: Low Alert. Woman may continue sending values.

Enter first number: 0
Exiting system... Stay safe!
```

8) using Fibonacci series  
Pseudocode

DISPLAY intro message

WHILE True

    READ a

    IF empty OR non-digit

        DISPLAY error

```

    CONTINUE
  IF a = 0
    BREAK
  READ b
  IF empty OR non-digit
    DISPLAY error
    CONTINUE
  IF b = 0
    BREAK
  DEFINE fib(n)
    IF n = 1 OR n = 2
      RETURN 1
    SET f1 = 1, f2 = 1
    LOOP from 3 to n
      f3 = f1 + f2
      f1 = f2
      f2 = f3
    RETURN f2
  SET fib_a = fib(a)
  SET fib_b = fib(b)
  SET alert = fib_a + fib_b
  IF alert >= 50
    DISPLAY extreme alert
    BREAK
  ELSE IF alert >= 20
    DISPLAY high alert
  ELSE IF alert >= 10
    DISPLAY moderate alert
  ELSE
    DISPLAY low alert
END WHILE

```

### Algorithm

Step 1: Start  
 Step 2: Display system instructions  
 Step 3: Enter infinite while loop  
 Step 4: Take first Fibonacci position input  
 Step 5: Validate → if empty or non-numeric, ask again  
 Step 6: If input = 0, stop program  
 Step 7: Take second Fibonacci position input  
 Step 8: Validate → if empty or non-numeric, ask again  
 Step 9: If input = 0, stop program  
 Step 10: Compute Fibonacci at position a  
 Step 11: Compute Fibonacci at position b  
 Step 12: Add both Fibonacci values → store in alert  
 Step 13:  
 If alert  $\geq 50$  → Extreme danger → break  
 If alert  $\geq 20$  → High alert

If alert  $\geq 10 \rightarrow$  Moderate alert

Else  $\rightarrow$  Low alert

Step 14: Repeat loop

Step 15: Stop

### Code

```
[9]: # Women Safety Alert System using Fibonacci Series

print("WOMEN SAFETY SYSTEM - FIBONACCI ALERT")
print("Enter two positions of Fibonacci series.")
print("Enter 0 anytime to stop.\n")

while True:
    a = input("Enter first Fibonacci position: ")

    if a == "":
        print("Error: Input cannot be empty.")
        continue

    if not a.isdigit():
        print("Please enter numbers only.")
        continue

    a = int(a)
    if a == 0:
        print("Exiting system... Stay safe!")
        break

    b = input("Enter second Fibonacci position: ")

    if b == "":
        print("Error: Input cannot be empty.")
        continue

    if not b.isdigit():
        print("Please enter numbers only.")
        continue

    b = int(b)
    if b == 0:
```

```

b = int(b)
if b == 0:
    print("Exiting system... Stay safe!")
    break

# Function to compute Fibonacci number at position n
def fib(n):
    if n == 1 or n == 2:
        return 1
    f1, f2 = 1, 1
    for _ in range(3, n + 1):
        f3 = f1 + f2
        f1 = f2
        f2 = f3
    return f2

fib_a = fib(a)
fib_b = fib(b)

print("Fibonacci at position", a, "=", fib_a)
print("Fibonacci at position", b, "=", fib_b)

# Combined alert value
alert = fib_a + fib_b
print("Combined Alert Value =", alert)

# Police Alert Levels
if alert >= 50:
    print("Police: EXTREME DANGER! Rushing Immediately!")
    break
elif alert >= 20:
    print("Police: High Alert! Moving towards location.\n")
elif alert >= 10:
    print("Police: Moderate Alert. Stay calm.\n")
else:
    print("Police: Low Alert. Monitoring... Woman can send again.\n")

```

WOMEN SAFETY SYSTEM - FIBONACCI ALERT  
Enter two positions of Fibonacci series.  
Enter 0 anytime to stop.

Enter first Fibonacci position: 4  
Enter second Fibonacci position: 7  
Fibonacci at position 4 = 3  
Fibonacci at position 7 = 13  
Combined Alert Value = 16  
Police: Moderate Alert. Stay calm.

Enter first Fibonacci position: 0  
Exiting system... Stay safe!

## 9)Using Arrays

### Pseudocode

```

DISPLAY intro message
WHILE True
    READ data
    IF data empty
        DISPLAY error
        CONTINUE
    IF data equals "0"
        DISPLAY exit message

```

```

    BREAK
  SPLIT data into parts
  CREATE empty array alerts
  SET valid = True
  FOR each element p in parts
    IF p is not digit
      SET valid = False
      BREAK
    ELSE
      APPEND integer p to alerts
  IF valid is False
    DISPLAY error
    CONTINUE
  DISPLAY alerts array
  SET max_alert = alerts[0]
  FOR each value v in alerts
    IF v > max_alert
      SET max_alert = v
  DISPLAY max_alert
  IF max_alert >= 9
    DISPLAY extreme alert
    BREAK
  ELSE IF max_alert >= 6
    DISPLAY high alert
  ELSE IF max_alert >= 3
    DISPLAY moderate alert
  ELSE
    DISPLAY low alert
END WHILE

```

### Algorithm

```

Step 1: Start
Step 2: Display system instructions
Step 3: Begin an infinite while loop
Step 4: Take space-separated input of alert values
Step 5: If input empty → show error and continue
Step 6: If input is "0" → terminate program
Step 7: Split input into tokens
Step 8: Convert each token into an integer and store in array alerts
Step 9: If any token is invalid (non-digit) → show error
Step 10: Initialize max_alert with first value of array
Step 11: Traverse entire array to find maximum alert value
Step 12:
  If max_alert ≥ 9 → Extreme danger → break
  Else if ≥ 6 → High alert
  Else if ≥ 3 → Moderate alert
  Else → Low alert
Step 13: Loop repeats until extreme alert is found

```

## Step 14: Stop

## Code

```
# Women Safety Alert System using Arrays

print("WOMEN SAFETY SYSTEM - ARRAY BASED ALERT")
print("Woman will enter multiple alert values in an array.")
print("System will check strongest alert and notify police.\n")

while True:
    data = input("Enter alert values separated by space (or 0 to quit): ")

    if data == "":
        print("Error: Input cannot be empty.")
        continue

    # If user wants to exit
    if data.strip() == "0":
        print("System exiting.. Stay Safe!")
        break

    # Convert input to array
    parts = data.split()
    alerts = []

    valid = True
    for p in parts:
        if not p.isdigit():
            valid = False
            break
        alerts.append(int(p))

    if not valid:
        print("Please enter numbers only.")
        continue

    print("Alert array =", alerts)

    print("Alert array =", alerts)

    # Find maximum alert value in array
    max_alert = alerts[0]
    for val in alerts:
        if val > max_alert:
            max_alert = val

    print("Highest Alert =", max_alert)

    # Police Response Conditions
    if max_alert >= 9:
        print("Police: EXTREME Danger! Rushing Immediately!")
        break
    elif max_alert >= 6:
        print("Police: HIGH Alert! Officers moving now.\n")
    elif max_alert >= 3:
        print("Police: MODERATE Alert. Stay calm.\n")
    else:
        print("Police: LOW Alert. Monitoring situation...\n")

WOMEN SAFETY SYSTEM - ARRAY BASED ALERT
Woman will enter multiple alert values in an array.
System will check strongest alert and notify police.

Enter alert values separated by space (or 0 to quit): 8
Alert array = [8]
Highest Alert = 8
Police: HIGH Alert! Officers moving now.

Enter alert values separated by space (or 0 to quit): 0
System exiting.. Stay Safe!
```