EX.NO: **DATE:**	**8 PUZZLE PROLEM**

AIM:

ALGORITHM:

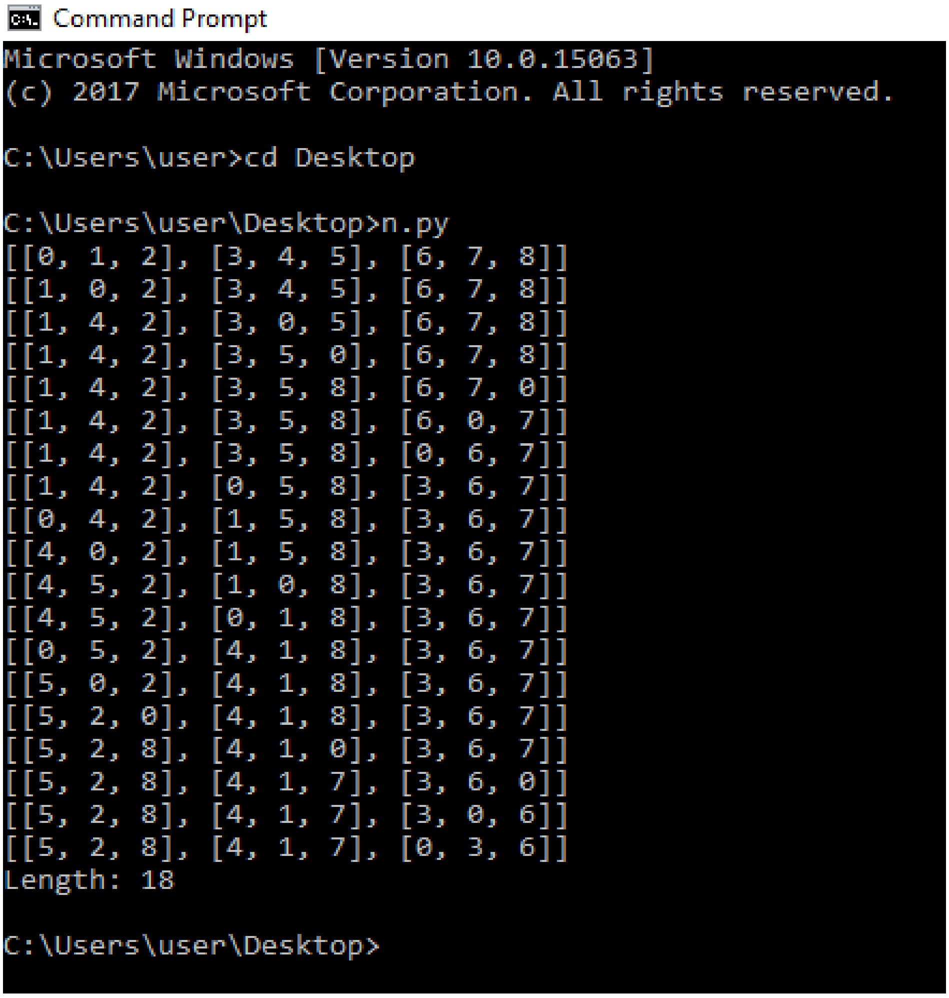
PROGRAM:

```
import random
import math
_goal_state = [[1,2,3],
               [4,5,6],
               [7,8,0]]
def index(item, seq):
    """Helper function that returns -1 for non-found index value of a seq"""
    try:
        return seq.index(item)
    except:
        return -1
class EightPuzzle:
    def_init_(self):
        self._hval = 0
        self._depth = 0
        # parent node in search path
        self._parent = None
        self.adj_matrix = []
        for i in range(3):
            self.adj_matrix.append(_goal_state[i][:])
    def_eq_(self, other):
        if self.__class___!= other._class_:
            return False
        else:
            return self.adj_matrix == other.adj_matrix
    def_str_(self):
        res = ''
        for row in range(3):
```

```
        res += ' '.join(map(str, self.adj_matrix[row]))
        res += '\r\n'
    return res
def _clone(self):
    p = EightPuzzle()
    for i in range(3):
        p.adj_matrix[i] = self.adj_matrix[i][:]
    return p
def _get_legal_moves(self):
    """Returns list of tuples with which the free space may
    be swapped"""
    # get row and column of the empty piece
    row, col = self.find(0)
    free =[]
    # find which pieces can move there
    if row > 0:
        free.append((row - 1, col))
    if col > 0:
        free.append((row, col - 1))
    if row < 2:
        free.append((row + 1, col))
    if col < 2:
        free.append((row, col + 1))
    return free
def _generate_moves(self):
    free = self._get_legal_moves()
    zero = self.find(0)
    def swap_and_clone(a, b):
        p = self._clone()
```

```
            p.swap(a,b)
            p._depth = self._depth + 1
            p._parent = self
            return p
        return map(lambda pair: swap_and_clone(zero, pair), free)
    def _generate_solution_path(self, path):
        if self._parent == None:
            return path
        else:
            path.append(self)
            return self._parent._generate_solution_path(path)
    def solve(self, h):
        """Performs A* search for goal state.
        h(puzzle) - heuristic function, returns an integer
        """
        def is_solved(puzzle):
            return puzzle.adj_matrix == _goal_state
        openl = [self]
        closedl = []
        move_count = 0
        while len(openl) > 0:
            x = openl.pop(0)
            move_count += 1
            if (is_solved(x)):
                if len(closedl) > 0:
                    return x._generate_solution_path([]), move_count
                else:
                    return [x]
            succ = x._generate_moves()
            idx_open = idx_closed = -1
            for move in succ:
                # have we already seen this node?
                idx_open = index(move, openl)
                idx_closed = index(move, closedl)
                hval = h(move)
                fval = hval + move._depth
                if idx_closed == -1 and idx_open == -1:
                    move._hval = hval
                    openl.append(move)
                elif idx_open > -1:
                    copy = openl[idx_open]
                    if fval < copy._hval + copy._depth:
                        # copy move's values over existing
                        copy._hval = hval
                        copy._parent = move._parent
                        copy._depth = move._depth
                elif idx_closed > -1:
                    copy = closedl[idx_closed]
                    if fval < copy._hval + copy._depth:
                        move._hval = hval
                        closedl.remove(copy)
                        openl.append(move)
            closedl.append(x)
            openl = sorted(openl, key=lambda p: p._hval + p._depth)
        # if finished state not found, return failure
        return [], 0
    def shuffle(self, step_count):
        for i in range(step_count):
            row, col = self.find(0)
            free = self._get_legal_moves()
            target = random.choice(free)
            self.swap((row, col), target)
            row, col = target
    def find(self, value):
        """returns the row, col coordinates of the specified value
           in the graph"""
        if value < 0 or value > 8:
            raise Exception("value out of range")
        for row in range(3):
            for col in range(3):
                if self.adj_matrix[row][col] == value:
                    return row, col
    def peek(self, row, col):
        """returns the value at the specified row and column"""
        return self.adj_matrix[row][col]
    def poke(self, row, col, value):
        """sets the value at the specified row and column"""
        self.adj_matrix[row][col] = value
    def swap(self, pos_a, pos_b):
        """swaps values at the specified coordinates"""
        temp = self.peek(*pos_a)
        self.poke(pos_a[0], pos_a[1], self.peek(*pos_b))
        self.poke(pos_b[0], pos_b[1], temp)
def heur(puzzle, item_total_calc, total_calc):
    """
    Heuristic template that provides the current and target position for each number and the
    total function.
    Parameters:
    puzzle - the puzzle
    item_total_calc - takes 4 parameters: current row, target row, current col, target col.
    Returns int.
    total_calc - takes 1 parameter, the sum of item_total_calc over all entries, and returns int.
    This is the value of the heuristic function
    """
    t = 0
    for row in range(3):
        for col in range(3):
            val = puzzle.peek(row, col) - 1
            target_col = val % 3
            target_row = val / 3
            # account for 0 as blank
            if target_row < 0:
                target_row = 2
            t += item_total_calc(row, target_row, col, target_col)
    return total_calc(t)
#some heuristic functions, the best being the standard manhattan distance in this case, as it comes
#closest to maximizing the estimated distance while still being admissible.
def h_manhattan(puzzle):
    return heur(puzzle,
                lambda r, tr, c, tc: abs(tr - r) + abs(tc - c),
                lambda t : t)
def h_manhattan_lsq(puzzle):
    return heur(puzzle,
                lambda r, tr, c, tc: (abs(tr - r) + abs(tc - c))**2,
                lambda t: math.sqrt(t))
def h_linear(puzzle):
    return heur(puzzle,
                lambda r, tr, c, tc: math.sqrt(math.sqrt((tr - r)**2 + (tc - c)**2)), lambda t:
                t)
def h_linear_lsq(puzzle): return
    heur(puzzle,
                lambda r, tr, c, tc: (tr - r)**2 + (tc - c)**2, lambda
                t: math.sqrt(t))
def h_default(puzzle): return 0
def main():
    p = EightPuzzle()
    p.shuffle(20) print (p)
    path, count = p.solve(h_manhattan)
    path.reverse()
    for i in path: print (i)
    print ("Solved with Manhattan distance exploring", count, "states") path,
    count = p.solve(h_manhattan_lsq)
    print ("Solved with Manhattan least squares exploring", count, "states") path,
    count = p.solve(h_linear)
    print ("Solved with linear distance exploring", count, "states") path,
    count = p.solve(h_linear_lsq)
    print ("Solved with linear least squares exploring", count, "states") #
      path, count = p.solve(heur_default)
#   print ("Solved with BFS-equivalent in", count, "moves") if
name______== "_main_":
    main()
```

OUTPUT:



```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\user>cd Desktop

C:\Users\user\Desktop>n.py
[[0, 1, 2], [3, 4, 5], [6, 7, 8]]
[[1, 0, 2], [3, 4, 5], [6, 7, 8]]
[[1, 4, 2], [3, 0, 5], [6, 7, 8]]
[[1, 4, 2], [3, 5, 0], [6, 7, 8]]
[[1, 4, 2], [3, 5, 8], [6, 7, 0]]
[[1, 4, 2], [3, 5, 8], [6, 0, 7]]
[[1, 4, 2], [3, 5, 8], [0, 6, 7]]
[[1, 4, 2], [0, 5, 8], [3, 6, 7]]
[[0, 4, 2], [1, 5, 8], [3, 6, 7]]
[[4, 0, 2], [1, 5, 8], [3, 6, 7]]
[[4, 5, 2], [1, 0, 8], [3, 6, 7]]
[[4, 5, 2], [0, 1, 8], [3, 6, 7]]
[[0, 5, 2], [4, 1, 8], [3, 6, 7]]
[[5, 0, 2], [4, 1, 8], [3, 6, 7]]
[[5, 2, 0], [4, 1, 8], [3, 6, 7]]
[[5, 2, 8], [4, 1, 0], [3, 6, 7]]
[[5, 2, 8], [4, 1, 7], [3, 6, 0]]
[[5, 2, 8], [4, 1, 7], [3, 0, 6]]
[[5, 2, 8], [4, 1, 7], [0, 3, 6]]
Length: 18

C:\Users\user\Desktop>
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	**INDEX**	**MAXIMUM MARKS**	**MARKS AWARDED**
1.	**AIM**		
2.	**ALGORITHM**		
3.	**PROGRAM**		
4.	**OUTPUT**		
5.	**RESULT**		
6.	**OBSERVATION**		
7.	**VIVA**		
8	**TOTAL**		
SIGNATURE			

RESULT:

EX.NO: **DATE:**	**8 QUEEN PROBLEM**

AIM:

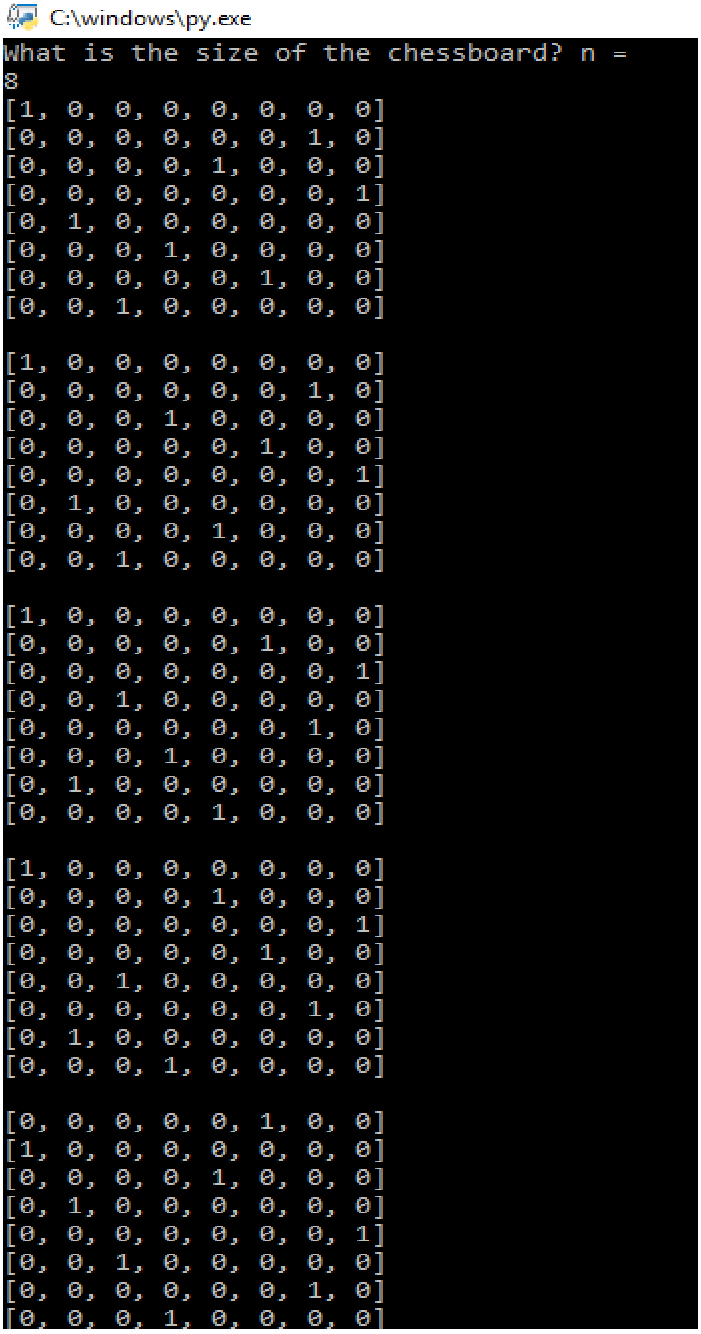
ALGORITHM:

PROGRAM:

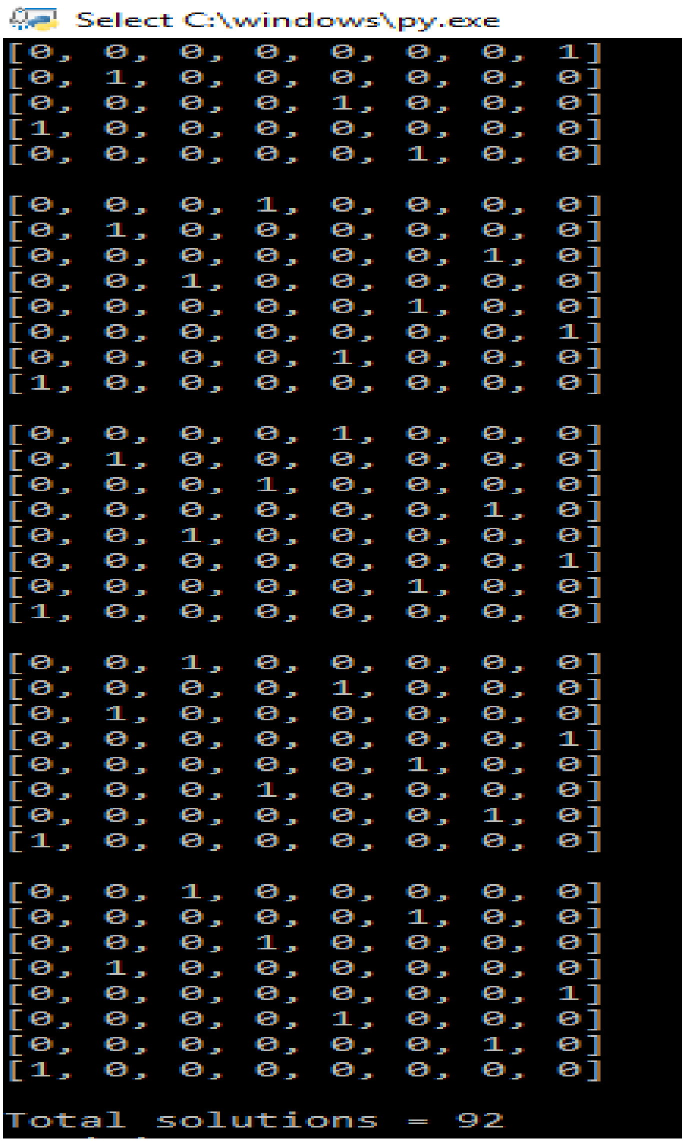
```
import copy
def take_input():
"""Accepts the size of the chess board""" while True:
try:
size = int(input('What is the size of the chessboard? n = \n')) if size == 1:
print("Trivial solution, choose a board size of atleast 4") if size <= 3:
print("Enter a value such that size>=4") continue
return size except ValueError:
print("Invalid value entered. Enter again") def get_board(size):
"""Returns an n by n board""" board = [0]*size
for ix in range(size): board[ix] = [0]*size
return board
def print_solutions(solutions, size):
"""Prints all the solutions in user friendly way""" for sol in solutions:
for row in sol: print(row)
print()
 def is_safe(board, row, col, size):
"""Check if it's safe to place a queen at board[x][y]""" #check row on left side
for iy in range(col):
if board[row][iy] == 1: return False
ix, iy = row, col
while ix >= 0 and iy >= 0: if board[ix][iy] == 1:
return False ix-=1
iy-=1
jx, jy = row,col
while jx < size and jy >= 0: if board[jx][jy] == 1:
return False jx+=1
jy-=1 return True
def solve(board, col, size):
"""Use backtracking to find all solutions""" #base case
if col >= size: return
for i in range(size):
if is_safe(board, i, col, size): board[i][col] = 1
```

```
 if col == size-1: add_solution(board) board[i][col] = 0 return
solve(board, col+1, size) #backtrack
board[i][col] = 0 def add_solution(board):
"""Saves the board state to the global variable 'solutions'""" global solutions
saved_board = copy.deepcopy(board) solutions.append(saved_board)
size = take_input() board = get_board(size) solutions = [] solve(board, 0, size)
print_solutions(solutions, size)
print("Total solutions = {}".format(len(solutions)))
```

OUTPUT:



```
What is the size of the chessboard? n =
8
[1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0]
[0, 0, 0, 0, 1, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1]
[0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 1, 0, 0, 0, 0, 0]

[1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0]
[0, 0, 0, 1, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1]
[0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 0, 0]

[1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0]
[0, 0, 0, 1, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0]

[1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0]
[0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 1, 0, 0]
[1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0]
[0, 0, 0, 1, 0, 0, 0, 0]
```



```
[0, 0, 0, 0, 0, 0, 0, 1]
[0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0]
[1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 1, 0, 0]

[0, 0, 0, 1, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0]
[0, 0, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 1, 0, 0, 0]
[1, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 1, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0]
[0, 0, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 1, 0, 0]
[1, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 1, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0]
[1, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 1, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 1, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0]
[1, 0, 0, 0, 0, 0, 0, 0]

Total solutions = 92
```

SAVEETHA SCHOOL OF ENGINEERING
DEPARTMENT OF CSE
RUBRICS

SLNO	INDEX	MAXIMUM MARKS	MARKS AWARDED
1.	AIM		
2.	ALGORITHM		
3.	PROGRAM		
4.	OUTPUT		
5.	RESULT		
6.	OBSERVATION		
7.	VIVA		
8	TOTAL		
SIGNATURE			

RESULT:

EX.NO: **DATE:**	**BREADTH FIRST SEARCH**

AIM:

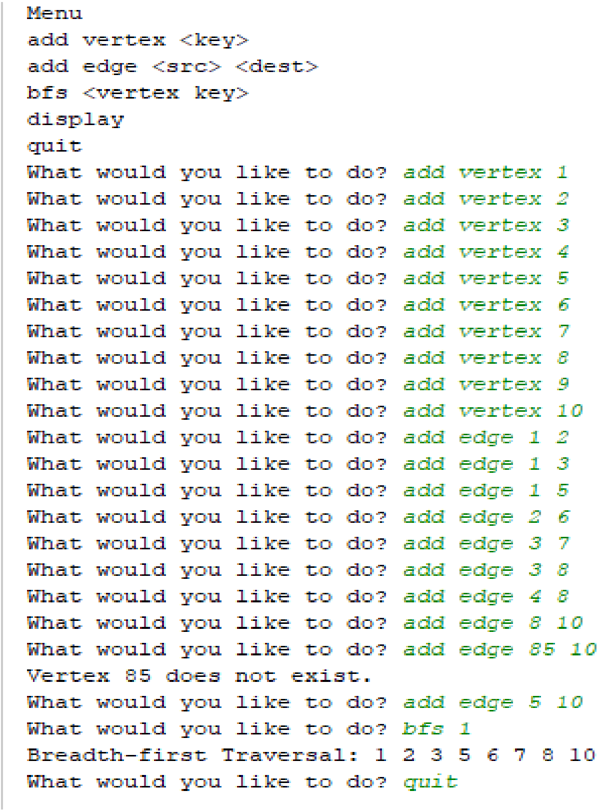
ALGORITHM:

PROGRAM:

```
class Graph:
def init (self):
# dictionary containing keys that map to the corresponding vertex object self.vertices = {}
def add_vertex(self, key):
"""Add a vertex with the given key to the graph.""" vertex = Vertex(key)
self.vertices[key] = vertex
def get_vertex(self, key):
"""Return vertex object with the corresponding key.""" return self.vertices[key]
def contains (self, key): return key in self.vertices
def add_edge(self, src_key, dest_key, weight=1):
"""Add edge from src_key to dest_key with given weight."""
self.vertices[src_key].add_neighbour(self.vertices[dest_key], weight)
def does_edge_exist(self, src_key, dest_key):
"""Return True if there is an edge from src_key to dest_key."""
return self.vertices[src_key].does_it_point_to(self.vertices[dest_key])
 def iter (self):
return iter(self.vertices.values())
class Vertex:
def init (self, key): self.key = key self.points_to = {}
def get_key(self):
"""Return key corresponding to this vertex object.""" return self.key
def add_neighbour(self, dest, weight):
"""Make this vertex point to dest with given edge weight.""" self.points_to[dest] = weight
def get_neighbours(self):
"""Return all vertices pointed to by this vertex.""" return self.points_to.keys()
def get_weight(self, dest):
"""Get weight of edge from this vertex to dest.""" return self.points_to[dest]
 def does_it_point_to(self, dest):
"""Return True if this vertex points to dest.""" return dest in self.points_to
class Queue:
def init (self): self.items = []
def is_empty(self): return self.items == []
def enqueue(self, data): self.items.append(data)
def dequeue(self):
return self.items.pop(0)
def display_bfs(vertex):
"""Display BFS Traversal starting at vertex.""" visited = set()
q = Queue() q.enqueue(vertex)
 visited.add(vertex) while not q.is_empty():
current = q.dequeue() print(current.get_key(), end=' ')
for dest in current.get_neighbours(): if dest not in visited:
visited.add(dest) q.enqueue(dest)
g = Graph() print('Menu')
print('add vertex <key>') print('add edge <src> <dest>') print('bfs <vertex key>') print('display')
print('quit')
while True:
do = input('What would you like to do? ').split()
operation = do[0]
if operation == 'add': suboperation = do[1]
if suboperation == 'vertex':
 key = int(do[2]) if key not in g:
g.add_vertex(key) else:
print('Vertex already exists.') elif suboperation == 'edge':
src = int(do[2]) dest = int(do[3]) if src not in g:
print('Vertex {} does not exist.'.format(src)) elif dest not in g:
print('Vertex {} does not exist.'.format(dest)) else:
if not g.does_edge_exist(src, dest): g.add_edge(src, dest)
else:
print('Edge already exists.')
elif operation == 'bfs': key = int(do[1])
print('Breadth-first Traversal: ', end=") vertex = g.get_vertex(key) display_bfs(vertex)
print()
 elif operation == 'display': print('Vertices: ', end=") for v in g:
print(v.get_key(), end=' ') print()
print('Edges: ') for v in g:
for dest in v.get_neighbours(): w = v.get_weight(dest)
```

```
print('(src={}, dest={}, weight={}) '.format(v.get_key(),

dest.get_key(), w))

print()

elif operation == 'quit':

 break
```

OUTPUT:



```
Menu
add vertex <key>
add edge <src> <dest>
bfs <vertex key>
display
quit
What would you like to do? add vertex 1
What would you like to do? add vertex 2
What would you like to do? add vertex 3
What would you like to do? add vertex 4
What would you like to do? add vertex 5
What would you like to do? add vertex 6
What would you like to do? add vertex 7
What would you like to do? add vertex 8
What would you like to do? add vertex 9
What would you like to do? add vertex 10
What would you like to do? add edge 1 2
What would you like to do? add edge 1 3
What would you like to do? add edge 1 5
What would you like to do? add edge 2 6
What would you like to do? add edge 3 7
What would you like to do? add edge 3 8
What would you like to do? add edge 4 8
What would you like to do? add edge 8 10
What would you like to do? add edge 85 10
Vertex 85 does not exist.
What would you like to do? add edge 5 10
What would you like to do? bfs 1
Breadth-first Traversal: 1 2 3 5 6 7 8 10
What would you like to do? quit
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	INDEX	MAXIMUM MARKS	MARKS AWARDED
1.	AIM		
2.	ALGORITHM		
3.	PROGRAM		
4.	OUTPUT		
5.	RESULT		
6.	OBSERVATION		
7.	VIVA		
8	TOTAL		
SIGNATURE			

RESULT:

EX.NO: DATE:	DEPTH FIRST SEARCH

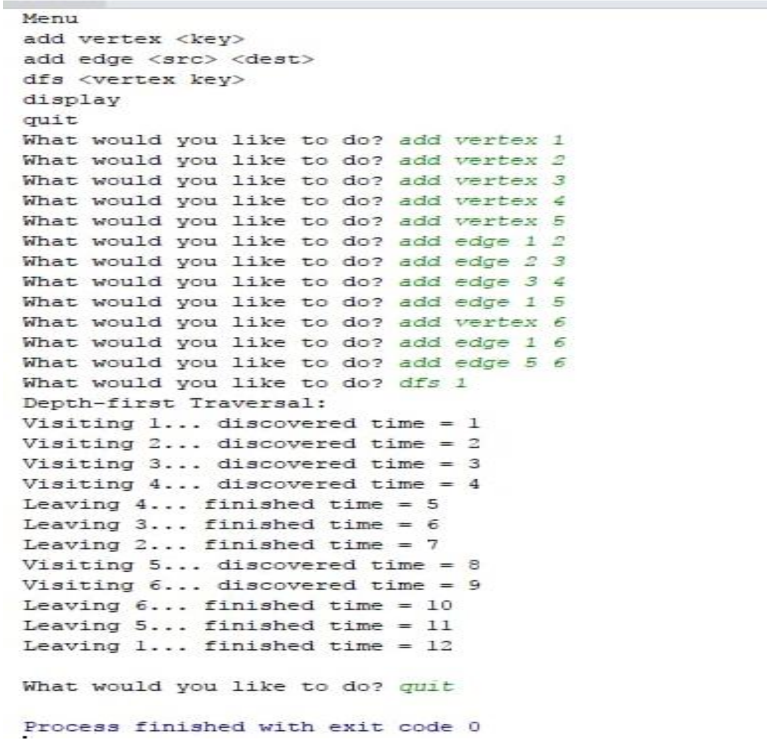
AIM:

ALGORITHM:

PROGRAM:

```
class Graph:
    def __init__(self):
        # dictionary containing keys that map to the corresponding vertex object
        self.vertices = {}
def add_vertex(self, key):
        """Add a vertex with the given key to the graph."""
        vertex = Vertex(key)
     self.vertices[key] = vertex
def get_vertex(self, key):
        """Return vertex object with the corresponding key."""
        return self.vertices[key]
def __contains__(self, key):
        return key in self.vertices
  def add_edge(self, src_key, dest_key, weight=1):
        """Add edge from src_key to dest_key with given weight."""
        self.vertices[src_key].add_neighbour(self.vertices[dest_key], weight)
 def does_edge_exist(self, src_key, dest_key):
        """Return True if there is an edge from src_key to dest_key."""
        return self.vertices[src_key].does_it_point_to(self.vertices[dest_key])
 def __iter__(self):
        return iter(self.vertices.values())
class Vertex:
    def __init__(self, key):
        self.key = key
        self.points_to = {}
 def get_key(self):
        """Return key corresponding to this vertex object."""
        return self.key
def add_neighbour(self, dest, weight):
        """Make this vertex point to dest with given edge weight."""
        self.points_to[dest] = weight
    def get_neighbours(self):
        """Return all vertices pointed to by this vertex."""
        return self.points_to.keys()
   def get_weight(self, dest):
        """Get weight of edge from this vertex to dest."""
        return self.points_to[dest]
  def does_it_point_to(self, dest):
        """Return True if this vertex points to dest."""
        return dest in self.points_to
class Stack:
    def __init__(self):
        self.items = []
     def is_empty(self):
        return self.items == []
    def push(self, data):
        self.items.append(data)
  def pop(self):
        return self.items.pop()
def display_dfs(v):
    visited = set()
    s = Stack()
    s.push(vertex)
    while not s.is_empty():
        current = s.pop()
if current in visited:
            continue
        print(current.get_key(), end=' ')
        visited.add(current)
        for dest in current.get_neighbours():
            if dest not in visited:
                s.push(dest)
 g = Graph()
print('Menu')
print('add vertex <key>')
print('add edge <src> <dest>')
print('dfs <vertex key>')
print('display')
print('quit')
 while True:
    do = input('What would you like to do? ').split()
    operation = do[0]
    if operation == 'add':
        suboperation = do[1]
        if suboperation == 'vertex':
            key = int(do[2])
            if key not in g:
                g.add_vertex(key)
            else:print('Vertex already exists.')
        elif suboperation == 'edge':
            src = int(do[2])
            dest = int(do[3])
            if src not in g:
                print('Vertex {} does not exist.'.format(src))
            elif dest not in g:
                print('Vertex {} does not exist.'.format(dest))
            else: if not g.does_edge_exist(src, dest):
                    g.add_edge(src, dest)
                else:
                    print('Edge already exists.')
    elif operation == 'dfs':
        key = int(do[1])
        print('Depth-first Traversal: ', end='')
        vertex = g.get_vertex(key)
        display_dfs(vertex) print()
   elif operation == 'display':
        print('Vertices: ', end='')
        for v in g:
            print(v.get_key(), end=' ')
    print()
        print('Edges: ')
        for v in g:
            for dest in v.get_neighbours():
                w = v.get_weight(dest)
                print('(src={}, dest={}, weight={}) '.format(v.get_key(), dest.get_key(), w))
        print() elif operation == 'quit':break
```

OUTPUT:



```
Menu
add vertex <key>
add edge <src> <dest>
dfs <vertex key>
display
quit
What would you like to do? add vertex 1
What would you like to do? add vertex 2
What would you like to do? add vertex 3
What would you like to do? add vertex 4
What would you like to do? add vertex 5
What would you like to do? add edge 1 2
What would you like to do? add edge 2 3
What would you like to do? add edge 3 4
What would you like to do? add edge 1 5
What would you like to do? add vertex 6
What would you like to do? add edge 1 6
What would you like to do? add edge 5 6
What would you like to do? dfs 1
Depth-first Traversal:
Visiting 1... discovered time = 1
Visiting 2... discovered time = 2
Visiting 3... discovered time = 3
Visiting 4... discovered time = 4
Leaving 4... finished time = 5
Leaving 3... finished time = 6
Leaving 2... finished time = 7
Visiting 5... discovered time = 8
Visiting 6... discovered time = 9
Leaving 6... finished time = 10
Leaving 5... finished time = 11
Leaving 1... finished time = 12

What would you like to do? quit

Process finished with exit code 0
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	INDEX	MAXIMUM MARKS	MARKS AWARDED
1.	AIM		
2.	ALGORITHM		
3.	PROGRAM		
4.	OUTPUT		
5.	RESULT		
6.	OBSERVATION		
7.	VIVA		
8	TOTAL		
SIGNATURE			

RESULT:

EX.NO: **DATE:**	**TRAVELLING SALESMAN PROBLEM**

AIM:

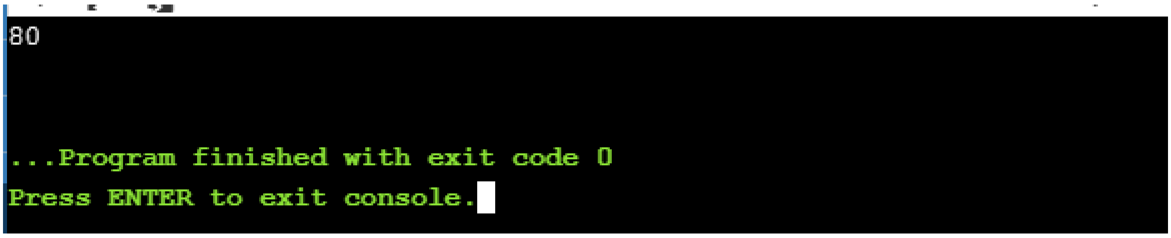
ALGORITHM:

PROGRAM:

```
from sys import maxsize
V = 4
def travellingSalesmanProblem(graph, s):
        vertex = []
        for i in range(V):
                if i != s: vertex.append(i)
        min_path = maxsize
        while True:
                current_pathweight = 0
                k = s
                for i in range(len(vertex)):
                        current_pathweight += graph[k][vertex[i]]
                        k = vertex[i]
                current_pathweight += graph[k][s]
                min_path = min(min_path, current_pathweight)
                if not next_permutation(vertex):
                        break
        return min_path
def next_permutation(L):
        n = len(L)
        i = n - 2
        while i >= 0 and L[i] >= L[i + 1]:
                i -= 1
        if i == -1:
                return False
        j = i + 1
        while j < n and L[j] > L[i]:
                j += 1  j -= 1
        L[i], L[j] = L[j], L[i]
        left = i + 1
        right = n - 1
```

```
while left < right:

L[left], L[right] = L[right], L[left]
        left += 1
        right -= 1

    return True
if __name__ == "__main__":

    # matrix representation of graph
    graph = [[0, 10, 15, 20], [10, 0, 35, 25],
            [15, 35, 0, 30], [20, 25, 30, 0]]
    s = 0
    print(travellingSalesmanProblem(graph, s))
```

OUTPUT:



```
80


...Program finished with exit code 0
Press ENTER to exit console.
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	**INDEX**	**MAXIMUM MARKS**	**MARKS AWARDED**
1.	**AIM**		
2.	**ALGORITHM**		
3.	**PROGRAM**		
4.	**OUTPUT**		
5.	**RESULT**		
6.	**OBSERVATION**		
7.	**VIVA**		
8	**TOTAL**		
SIGNATURE			

RESULT:

EX.NO: **DATE:**	**MIN-MAX ALGORITHM**

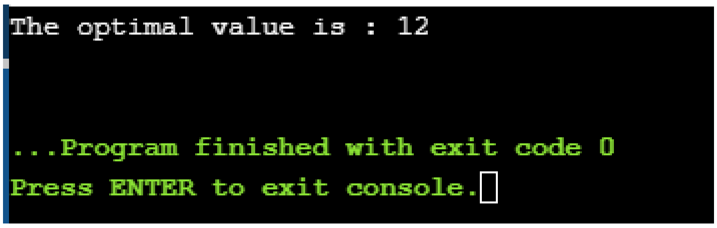
AIM:

ALGORITHM:

PROGRAM:

```
import math
def minimax (curDepth, nodeIndex, maxTurn, scores, targetDepth):
case : targetDepth reached
        if (curDepth == targetDepth):
                return scores[nodeIndex]
                if (maxTurn):
                return max(minimax(curDepth + 1, nodeIndex * 2,
                False, scores, targetDepth), minimax(curDepth + 1, nodeIndex * 2 + 1,
                False, scores, targetDepth))
        else: return min(minimax(curDepth + 1, nodeIndex * 2,
                True, scores, targetDepth), minimax(curDepth + 1, nodeIndex * 2 + 1,
                True, scores, targetDepth))
scores = [3, 5, 2, 9, 12, 5, 23, 23]
treeDepth = math.log(len(scores), 2)
print("The optimal value is : ", end = "")
print(minimax(0, 0, True, scores, treeDepth))
```

OUTPUT:



```
The optimal value is : 12


...Program finished with exit code 0
Press ENTER to exit console.
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	**INDEX**	**MAXIMUM MARKS**	**MARKS AWARDED**
1.	**AIM**		
2.	**ALGORITHM**		
3.	**PROGRAM**		
4.	**OUTPUT**		
5.	**RESULT**		
6.	**OBSERVATION**		
7.	**VIVA**		
8	**TOTAL**		
SIGNATURE			

RESULT:

EX.NO: **DATE:**	**FAMILY TREE**

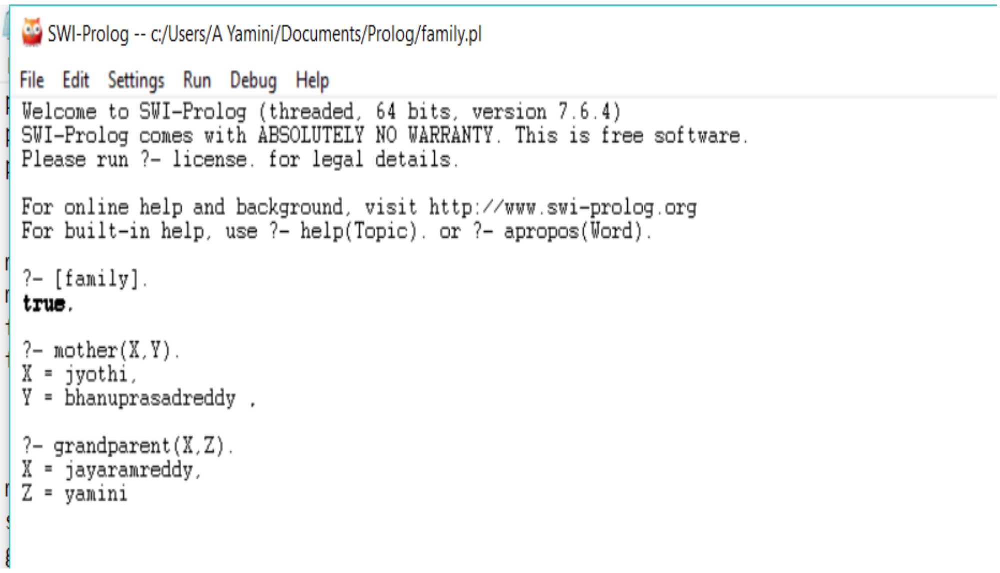
AIM:

ALGORITHM:

PROGRAM:

```
parent(jayaramreddy,bhanuprasadreddy).
parent(jyothi,bhanuprasadreddy).
parent(bhanuprasadreddy,yamini).
male(jayaramreddy).
male(bhanuprasadreddy).
female(jyothi).
female(yamini).
mother(X,Y):-parent(X,Y),female(X).
sister(X,Y):-parent(Z,X),parent(Z,Y),female(X).
grandparent(X,Z):-parent(X,Y),parent(Y,Z).
```

OUTPUT:



SWI-Prolog -- c:/Users/A Yamini/Documents/Prolog/family.pl
File Edit Settings Run Debug Help

```
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- [family].
true.

?- mother(X,Y).
X = jyothi,
Y = bhanuprasadreddy .

?- grandparent(X,Z).
X = jayaramreddy,
Z = yamini
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	INDEX	MAXIMUM MARKS	MARKS AWARDED
1.	AIM		
2.	ALGORITHM		
3.	PROGRAM		
4.	OUTPUT		
5.	RESULT		
6.	OBSERVATION		
7.	VIVA		
8	TOTAL		
SIGNATURE			

RESULT:

EX.NO: **DATE:**	**FACTORIAL**

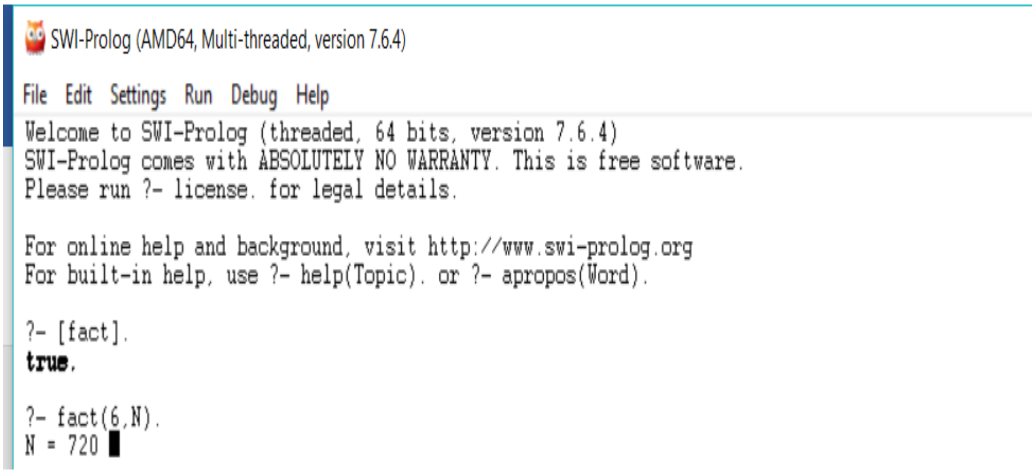
AIM:

ALGORITHM:

PROGRAM:

```
fact(0, 1).

fact(N, F) :-

(% The below is for +ve factorialN > 0->

(N1 is N - 1,

fact(N1,F1),

F is N * F1);

% The below is for -ve factorial

N<0->(N1 is N+1,fact(N1, F1),F is N * F1)).
```

OUTPUT:


SWI-Prolog (AMD64, Multi-threaded, version 7.6.4)
File Edit Settings Run Debug Help

```
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- [fact].
true.

?- fact(6,N).
N = 720
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	**INDEX**	**MAXIMUM MARKS**	**MARKS AWARDED**
1.	**AIM**		
2.	**ALGORITHM**		
3.	**PROGRAM**		
4.	**OUTPUT**		
5.	**RESULT**		
6.	**OBSERVATION**		
7.	**VIVA**		
8	**TOTAL**		
SIGNATURE			

RESULT:

EX.NO: **DATE:**	**GCD OF TWO NUMBERS**

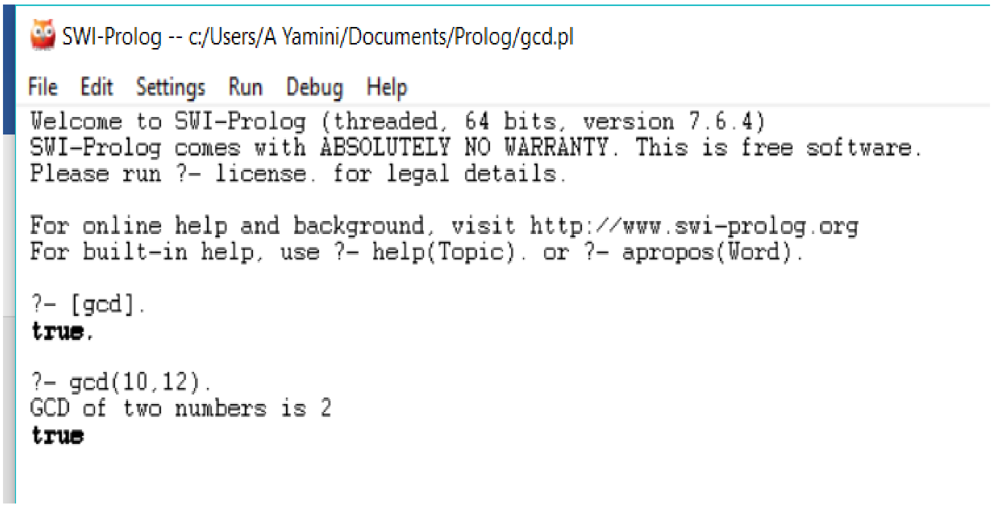
AIM:

ALGORITHM:

PROGRAM:

```
gcd(X,Y):-X=Y,write('GCD of two numbers is '),write(X);
 X=0,write('GCD of two numbers is '),write(Y);
 Y=0,write('GCD of two numbers is '),write(X);
 Y>X,Y1 is Y-X,gcd(X,Y1);
 X>Y,Y1 is X-Y,gcd(Y1,Y).
```

OUTPUT:



SWI-Prolog -- c:/Users/A Yamini/Documents/Prolog/gcd.pl
File Edit Settings Run Debug Help

```
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- [gcd].
true.

?- gcd(10,12).
GCD of two numbers is 2
true
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	**INDEX**	**MAXIMUM MARKS**	**MARKS AWARDED**
1.	**AIM**		
2.	**ALGORITHM**		
3.	**PROGRAM**		
4.	**OUTPUT**		
5.	**RESULT**		
6.	**OBSERVATION**		
7.	**VIVA**		
8	**TOTAL**		
SIGNATURE			

RESULT:

EX.NO: **DATE:**	**INPUT FROM USER**

AIM:

ALGORITHM:

PROGRAM:

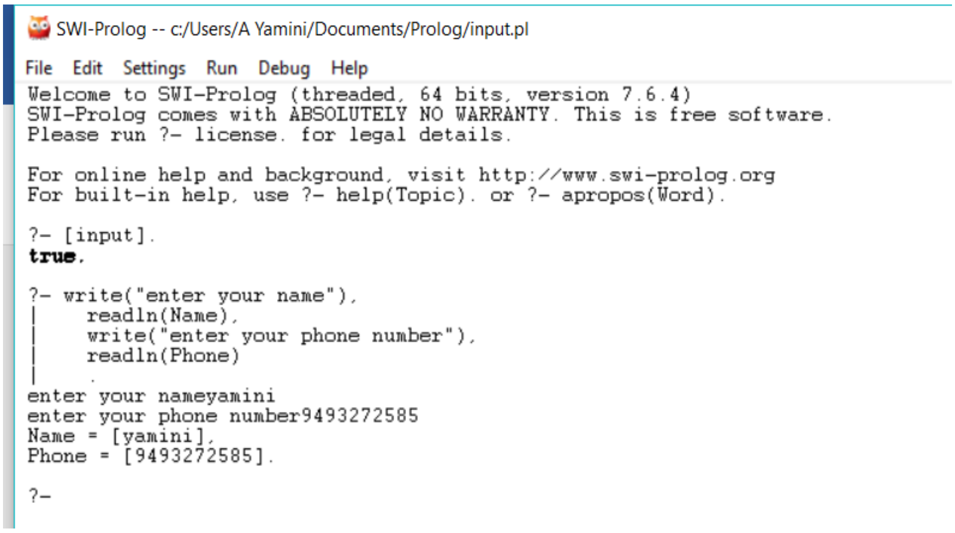
reference("yamini", "9493272585").

reference("radhika", "8919666297").

reference("hemanth", "9642499090").

reference("jayaram reddy", "9490013093").

OUTPUT:



SWI-Prolog -- c:/Users/A Yamini/Documents/Prolog/input.pl
File Edit Settings Run Debug Help

```
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- [input].
true.

?- write("enter your name"),
|    readln(Name),
|    write("enter your phone number"),
|    readln(Phone)
|    .
enter your nameyamini
enter your phone number9493272585
Name = [yamini],
Phone = [9493272585].

?-
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	**INDEX**	**MAXIMUM MARKS**	**MARKS AWARDED**
1.	**AIM**		
2.	**ALGORITHM**		
3.	**PROGRAM**		
4.	**OUTPUT**		
5.	**RESULT**		
6.	**OBSERVATION**		
7.	**VIVA**		
8	**TOTAL**		
SIGNATURE			

RESULT:

EX.NO: **DATE:**	**OUTPUT FROM USER**

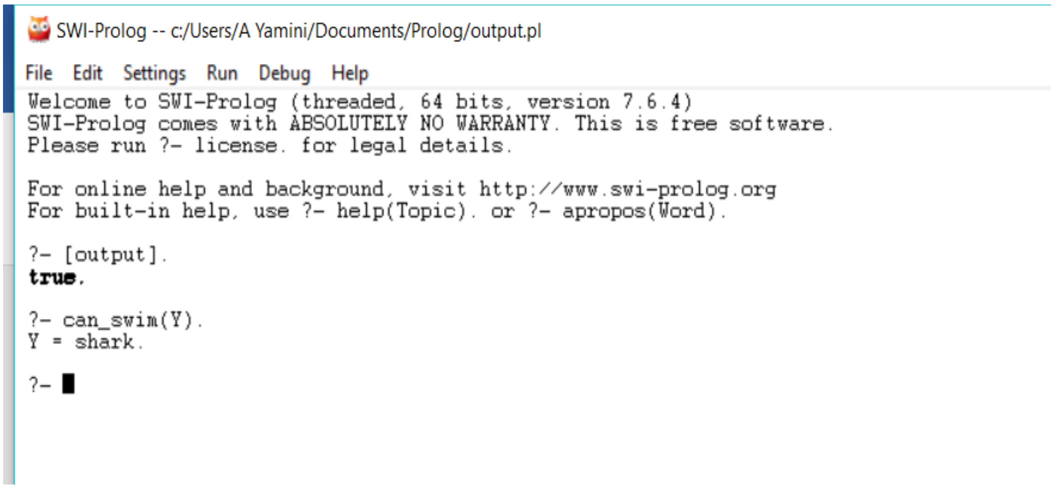
AIM:

ALGORITHM:

PROGRAM:

```
type(ungulate,animal).

type(fish,animal).

is_a(zebra,ungulate).

is_a(herring,fish).

is_a(shark,fish).

lives(zebra,on_land).

lives(frog,on_land).

lives(frog,in_water).

lives(shark,in_water).

can_swim(Y):- type(X, animal), is_a(Y,X), lives(Y, in_water).
```

OUTPUT:


SWI-Prolog -- c:/Users/A Yamini/Documents/Prolog/output.pl
File Edit Settings Run Debug Help

```
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- [output].
true.

?- can_swim(Y).
Y = shark.

?- 
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	**INDEX**	**MAXIMUM MARKS**	**MARKS AWARDED**
1.	**AIM**		
2.	**ALGORITHM**		
3.	**PROGRAM**		
4.	**OUTPUT**		
5.	**RESULT**		
6.	**OBSERVATION**		
7.	**VIVA**		
8	**TOTAL**		
SIGNATURE			

RESULT:

EX.NO: 12 **DATE:**	**MONKEY BANANA**

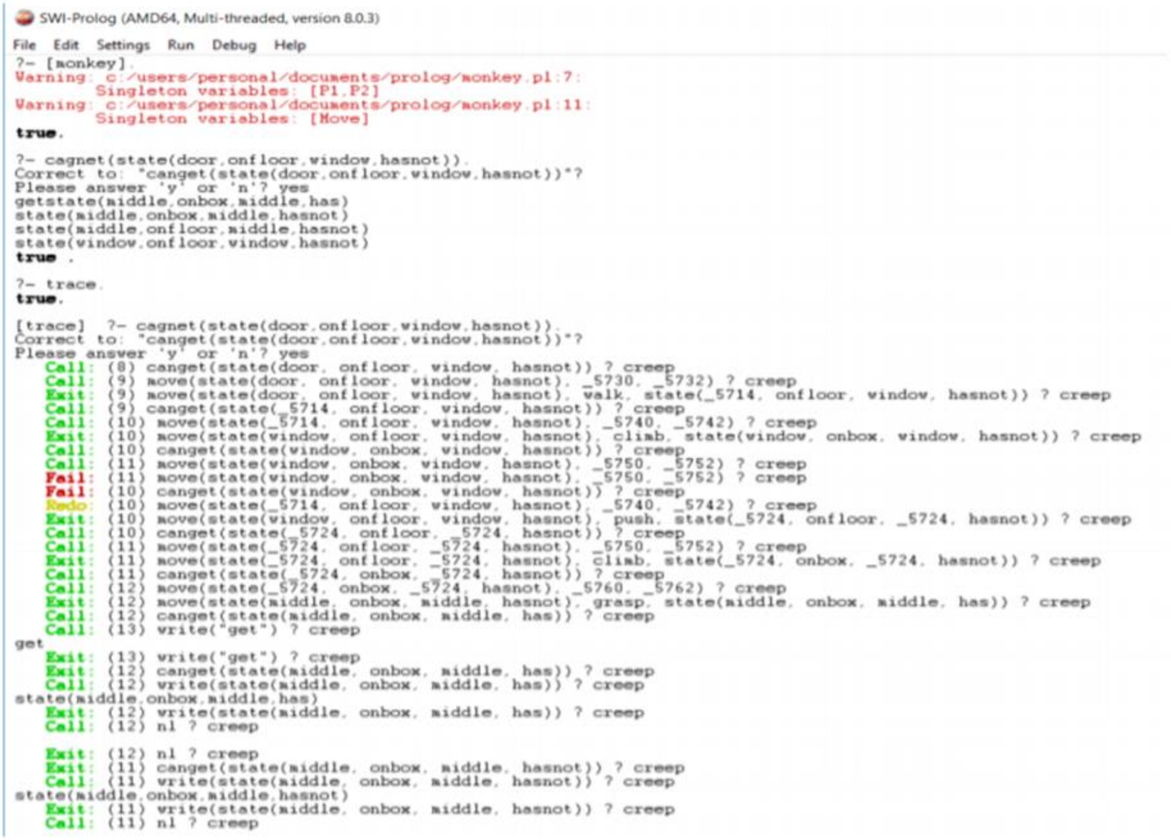
AIM:

ALGORITHM:

PROGRAM:

```
move(state(middle,onbox,middle,hasnot),
   grasp,state(middle,onbox,middle,has)).
move(state(P,onfloor,P,hasnot),climb,
   state(P,onbox,P,hasnot)).
move(state(P,onfloor,P,hasnot),push,
   state(P1,onfloor,P1,hasnot)).
move(state(P1,onfloor,B,hasnot),walk,
   state(P2,onfloor,B,hasnot)).
canget(state(_,_,_,has)) :-
     write("get").
canget(State1) :-
move(State1,Move,State2),
    canget(State2),
      write(State2),nl.
```

OUTPUT:



SWI-Prolog (AMD64, Multi-threaded, version 8.0.3)

```
?- [monkey].
Warning: c:/users/personal/documents/prolog/monkey.pl:7:
        Singleton variables: [P1,P2]
Warning: c:/users/personal/documents/prolog/monkey.pl:11:
        Singleton variables: [Move]
true.

?- cagnet(state(door,onfloor,window,hasnot)).
Correct to: "canget(state(door,onfloor,window,hasnot))"?
Please answer 'y' or 'n'? yes
getstate(middle,onbox,middle,has)
state(middle,onbox,middle,hasnot)
state(middle,onfloor,middle,hasnot)
state(window,onfloor,window,hasnot)
true .

?- trace.
true.

[trace]  ?- cagnet(state(door,onfloor,window,hasnot)).
Correct to: "canget(state(door,onfloor,window,hasnot))"?
Please answer 'y' or 'n'? yes
   Call: (8) canget(state(door, onfloor, window, hasnot)) ? creep
   Call: (9) move(state(door, onfloor, window, hasnot), _5730, _5732) ? creep
   Exit: (9) move(state(door, onfloor, window, hasnot), walk, state(_5714, onfloor, window, hasnot)) ? creep
   Call: (9) canget(state(_5714, onfloor, window, hasnot)) ? creep
   Call: (10) move(state(_5714, onfloor, window, hasnot), _5740, _5742) ? creep
   Exit: (10) move(state(window, onfloor, window, hasnot), climb, state(window, onbox, window, hasnot)) ? creep
   Call: (10) canget(state(window, onbox, window, hasnot)) ? creep
   Call: (11) move(state(window, onbox, window, hasnot), _5750, _5752) ? creep
   Fail: (11) move(state(window, onbox, window, hasnot), _5750, _5752) ? creep
   Fail: (10) canget(state(window, onbox, window, hasnot)) ? creep
   Redo: (10) move(state(_5714, onfloor, window, hasnot), _5740, _5742) ? creep
   Exit: (10) move(state(window, onfloor, window, hasnot), push, state(_5724, onfloor, _5724, hasnot)) ? creep
   Call: (10) canget(state(_5724, onfloor, _5724, hasnot)) ? creep
   Call: (11) move(state(_5724, onfloor, _5724, hasnot), _5750, _5752) ? creep
   Exit: (11) move(state(_5724, onfloor, _5724, hasnot), climb, state(_5724, onbox, _5724, hasnot)) ? creep
   Call: (11) canget(state(_5724, onbox, _5724, hasnot)) ? creep
   Call: (12) move(state(_5724, onbox, _5724, hasnot), _5760, _5762) ? creep
   Exit: (12) move(state(middle, onbox, middle, hasnot), grasp, state(middle, onbox, middle, has)) ? creep
   Call: (12) canget(state(middle, onbox, middle, has)) ? creep
   Call: (13) write("get") ? creep
get
   Exit: (13) write("get") ? creep
   Exit: (12) canget(state(middle, onbox, middle, has)) ? creep
   Call: (12) write(state(middle, onbox, middle, has)) ? creep
state(middle,onbox,middle,has)
   Exit: (12) write(state(middle, onbox, middle, has)) ? creep
   Call: (12) nl ? creep

   Exit: (12) nl ? creep
   Exit: (11) canget(state(middle, onbox, middle, hasnot)) ? creep
   Call: (11) write(state(middle, onbox, middle, hasnot)) ? creep
state(middle,onbox,middle,hasnot)
   Exit: (11) write(state(middle, onbox, middle, hasnot)) ? creep
   Call: (11) nl ? creep
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	INDEX	MAXIMUM MARKS	MARKS AWARDED
1.	AIM		
2.	ALGORITHM		
3.	PROGRAM		
4.	OUTPUT		
5.	RESULT		
6.	OBSERVATION		
7.	VIVA		
8	TOTAL		
SIGNATURE			

RESULT:

EX.NO: **DATE:**	**LIST**

AIM:

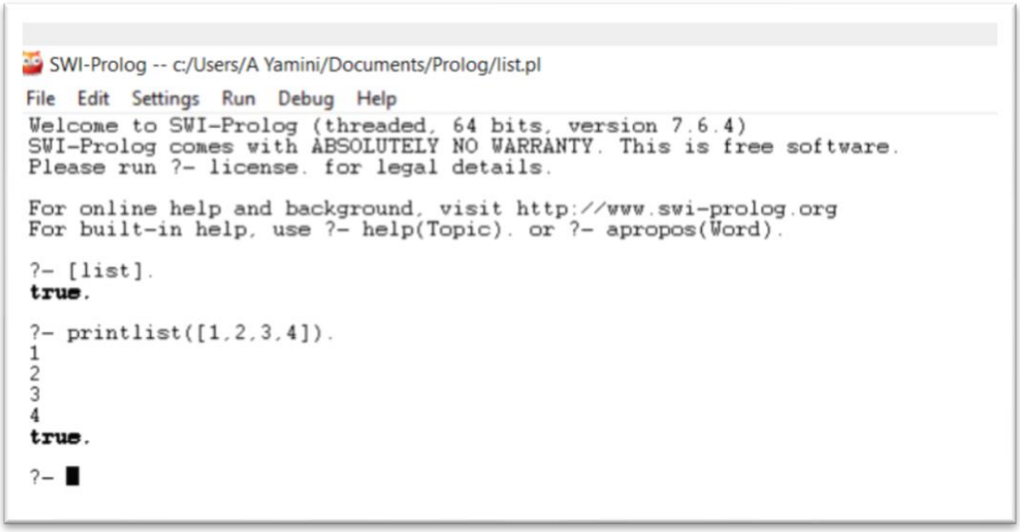
ALGORITHM:

PROGRAM:

11 A) PRINT LIST:

```
printlist([]).
    printlist([X|List]) :-
        write(X),nl,
        printlist(List).
```

OUTPUT:



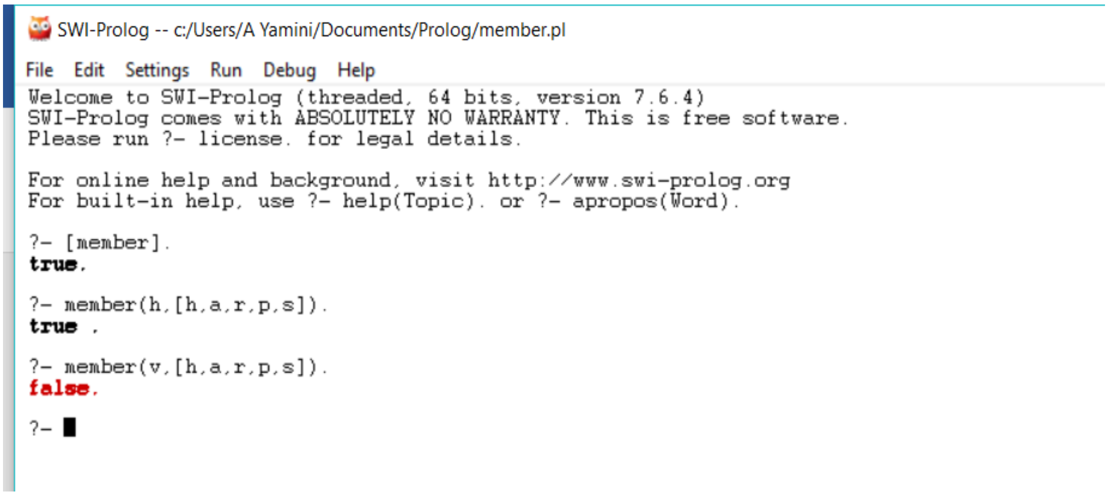
SWI-Prolog -- c:/Users/A Yamini/Documents/Prolog/list.pl
File Edit Settings Run Debug Help

```
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- [list].
true.

?- printlist([1,2,3,4]).
1
2
3
4
true.

?- ▮
```

11 B) MEMBER IS PRESENT OR NOT IN A LIST PROGRAM:

```
member(X,List):-
        delete(X,List,_).
    delete(X,[X|Tail],Tail).
    delete(X,[Y|Tail1],[Y|Tail2]):-
        delete(X,Tail1,Tail2).
```

OUTPUT:



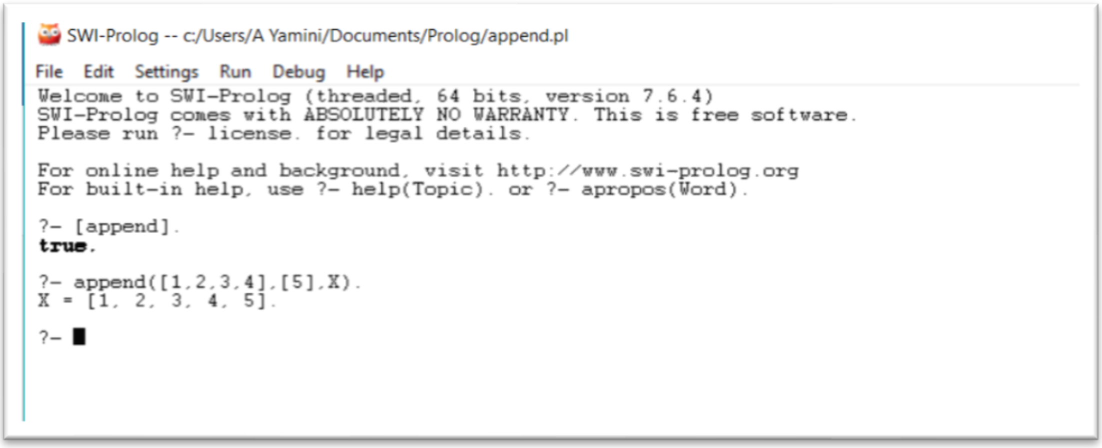
SWI-Prolog -- c:/Users/A Yamini/Documents/Prolog/member.pl
File Edit Settings Run Debug Help

```
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- [member].
true.

?- member(h,[h,a,r,p,s]).
true .

?- member(v,[h,a,r,p,s]).
false.

?- ▮
```

11 C) APPEND LIST:

```
append([],L,L).
 append([X|L1],L2,[X|L3]) :- append(L1,L2,L3).
```

OUTPUT:



SWI-Prolog -- c:/Users/A Yamini/Documents/Prolog/append.pl
File Edit Settings Run Debug Help

```
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- [append].
true.

?- append([1,2,3,4],[5],X).
X = [1, 2, 3, 4, 5].

?-
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	INDEX	MAXIMUM MARKS	MARKS AWARDED
1.	AIM		
2.	ALGORITHM		
3.	PROGRAM		
4.	OUTPUT		
5.	RESULT		
6.	OBSERVATION		
7.	VIVA		
8	TOTAL		
SIGNATURE			

RESULT:

EX.NO: 13 **DATE:**	**MEDICAL DIAGNOSIS**

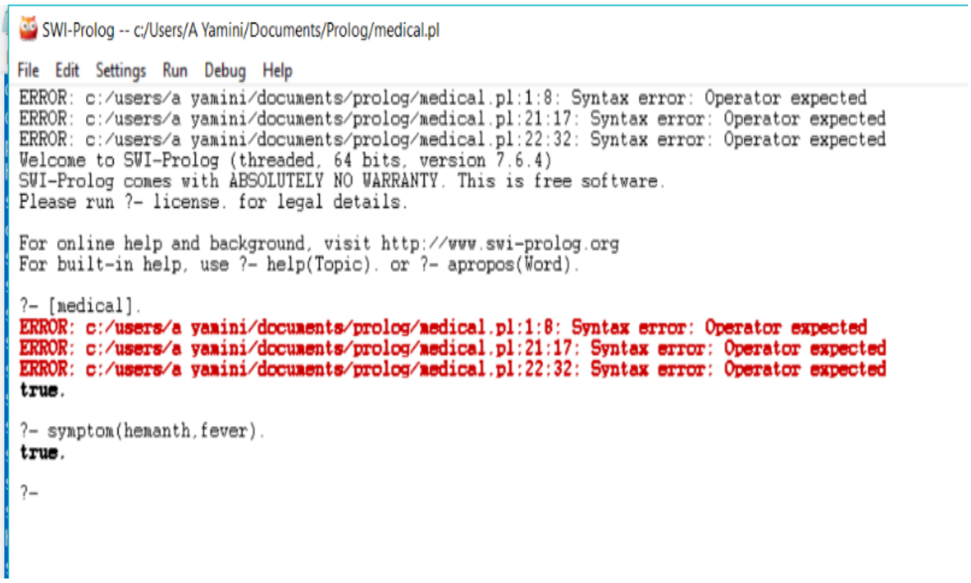
AIM:

ALGORITHM:

PROGRAM:

```
domains
disease,indication,name=symbol
predicates
hypothesis(name,disease)
symptom(name,indication)
clauses
symptom(yamini,fever).
symptom(yamini,rash) .
symptom(yamini,headache).
symptom(yamini,runn_nose).
symptom(hemanth,chills).
symptom(hemanth,fever).
symptom(hemnth,headache).
symptom(radhika,runny_nose).
symptom(radhika,rash).
symptom(radhika,flu).
hypothesis(Patient,measels):
symptom(Patient,fever),
symptom(Patient,cough),
symptom(Patient,conjunctivitis),
symptom(Patient,r ash).
hypothesis(Patient,german_measl es):
symptom(Patient,f ev er ),
symptom(Patient,headache),
symptom(Patient,runny_nose),
symptom(Patient,rash).
```

OUTPUT:



SWI-Prolog -- c:/Users/A Yamini/Documents/Prolog/medical.pl
File Edit Settings Run Debug Help

```
ERROR: c:/users/a yamini/documents/prolog/medical.pl:1:8: Syntax error: Operator expected
ERROR: c:/users/a yamini/documents/prolog/medical.pl:21:17: Syntax error: Operator expected
ERROR: c:/users/a yamini/documents/prolog/medical.pl:22:32: Syntax error: Operator expected
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- [medical].
ERROR: c:/users/a yamini/documents/prolog/medical.pl:1:8: Syntax error: Operator expected
ERROR: c:/users/a yamini/documents/prolog/medical.pl:21:17: Syntax error: Operator expected
ERROR: c:/users/a yamini/documents/prolog/medical.pl:22:32: Syntax error: Operator expected
true.

?- symptom(hemanth,fever).
true.

?-
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	INDEX	MAXIMUM MARKS	MARKS AWARDED
1.	AIM		
2.	ALGORITHM		
3.	PROGRAM		
4.	OUTPUT		
5.	RESULT		
6.	OBSERVATION		
7.	VIVA		
8	TOTAL		
SIGNATURE			

RESULT:

EX.NO: **DATE:**	**DECISION TREE**

AIM:

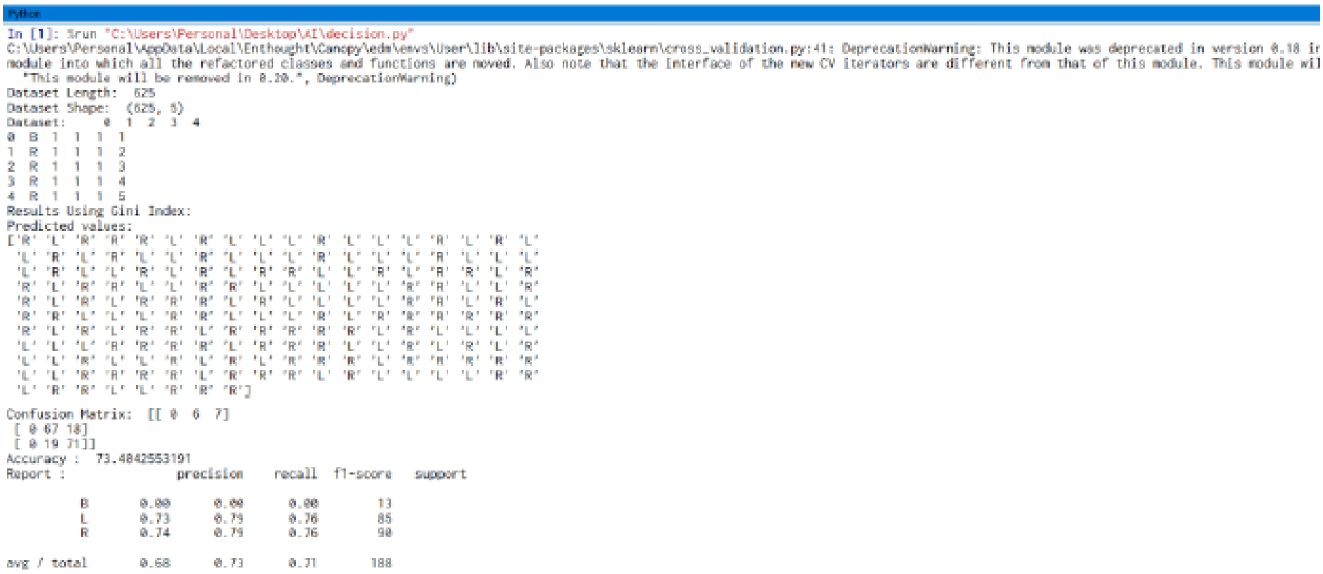
ALGORITHM:

PROGRAM:

```
import numpy as np
import pandas as pd
from sklearn.metrics import confusion_matrix
from sklearn.cross_validation import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
def importdata():
balance_data = pd.read_csv(
'https://archive.ics.uci.edu/ml/machine-learning-'+
'databases/balance-scale/balance-scale.data',
sep= ',', header = None)
print ("Dataset Length: ", len(balance_data))
print ("Dataset Shape: ", balance_data.shape)
print ("Dataset: ",balance_data.head())
return balance_data
def splitdataset(balance_data):
X = balance_data.values[:, 1:5]
Y = balance_data.values[:, 0]
X_train, X_test, y_train, y_test = train_test_split(
X, Y, test_size = 0.3, random_state = 100)
return X, Y, X_train, X_test, y_train, y_test
def train_using_gini(X_train, X_test, y_train):
clf_gini = DecisionTreeClassifier(criterion = "gini",
random_state = 100,max_depth=3, min_samples_leaf=5)
clf_gini.fit(X_train, y_train)
return clf_gini
def tarin_using_entropy(X_train, X_test, y_train):
clf_entropy = DecisionTreeClassifier(
criterion = "entropy", random_state = 100,
max_depth = 3, min_samples_leaf = 5)
clf_entropy.fit(X_train, y_train)
return clf_entropy
```

```
def prediction(X_test, clf_object):
y_pred = clf_object.predict(X_test)
print("Predicted values:")
print(y_pred)
return y_pred
def cal_accuracy(y_test, y_pred):
print("Confusion Matrix: ",
confusion_matrix(y_test, y_pred))
print ("Accuracy : ",
accuracy_score(y_test,y_pred)*100)
print("Report : ",
classification_report(y_test, y_pred))
# Driver code
def main():
# Building Phase
data = importdata()
X, Y, X_train, X_test, y_train, y_test = splitdataset(data)
clf_gini = train_using_gini(X_train, X_test, y_train)
clf_entropy = tarin_using_entropy(X_train, X_test, y_train)
# Operational Phase
print("Results Using Gini Index:")
# Prediction using gini
y_pred_gini = prediction(X_test, clf_gini)
cal_accuracy(y_test, y_pred_gini)
print("Results Using Entropy:")
# Prediction using entropy
y_pred_entropy = prediction(X_test, clf_entropy)
cal_accuracy(y_test, y_pred_entropy)
# Calling main function
if __name__=="__main__":
main()
```

OUTPUT:



```
In [1]: %run "C:\Users\Personal\Desktop\AI\decision.py"
C:\Users\Personal\AppData\Local\Enthought\Canopy\edm\envs\User\lib\site-packages\sklearn\cross_validation.py:41: DeprecationWarning: This module was deprecated in version 0.18 ir
module into which all the refactored classes and functions are moved. Also note that the interface of the new CV iterators are different from that of this module. This module wil
  "This module will be removed in 0.20.", DeprecationWarning)
Dataset Length:  625
Dataset Shape:  (625, 5)
Dataset:     0  1  2  3  4
0  B  1  1  1  1
1  R  1  1  1  2
2  R  1  1  1  3
3  R  1  1  1  4
4  R  1  1  1  5
Results Using Gini Index:
Predicted values:
['R' 'L' 'R' 'R' 'R' 'L' 'R' 'L' 'L' 'L' 'R' 'L' 'L' 'L' 'R' 'L' 'R' 'L'
 'L' 'R' 'L' 'R' 'L' 'L' 'R' 'L' 'L' 'L' 'R' 'L' 'L' 'L' 'R' 'L' 'L' 'L'
 'L' 'R' 'L' 'L' 'R' 'L' 'R' 'L' 'R' 'R' 'L' 'L' 'R' 'L' 'R' 'R' 'L' 'R'
 'R' 'L' 'R' 'R' 'L' 'L' 'R' 'R' 'L' 'L' 'L' 'L' 'L' 'R' 'R' 'L' 'L' 'R'
 'R' 'L' 'R' 'L' 'R' 'R' 'R' 'L' 'R' 'L' 'L' 'L' 'L' 'R' 'R' 'L' 'R' 'L'
 'R' 'R' 'L' 'L' 'L' 'R' 'R' 'L' 'L' 'L' 'R' 'L' 'R' 'R' 'R' 'R' 'R' 'R'
 'R' 'L' 'R' 'L' 'R' 'R' 'L' 'R' 'R' 'R' 'R' 'R' 'L' 'R' 'L' 'L' 'L' 'L'
 'L' 'L' 'L' 'R' 'R' 'R' 'R' 'L' 'R' 'R' 'R' 'L' 'L' 'R' 'L' 'R' 'L' 'R'
 'L' 'L' 'R' 'L' 'L' 'R' 'L' 'R' 'L' 'R' 'R' 'R' 'L' 'R' 'R' 'R' 'R' 'R'
 'L' 'L' 'R' 'R' 'R' 'R' 'L' 'R' 'R' 'R' 'L' 'R' 'L' 'L' 'L' 'L' 'R' 'R'
 'L' 'R' 'R' 'L' 'L' 'R' 'R' 'R']

Confusion Matrix:  [[ 0  6  7]
 [ 0 67 18]
 [ 0 19 71]]
Accuracy :  73.4042553191
Report :              precision    recall  f1-score   support

          B       0.00      0.00      0.00        13
          L       0.73      0.79      0.76        85
          R       0.74      0.79      0.76        90

avg / total       0.68      0.73      0.71       188
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	INDEX	MAXIMUM MARKS	MARKS AWARDED
1.	AIM		
2.	ALGORITHM		
3.	PROGRAM		
4.	OUTPUT		
5.	RESULT		
6.	OBSERVATION		
7.	VIVA		
8	TOTAL		
SIGNATURE			

RESULT:

EX.NO: **DATE:**	**NEURAL NETWORK**

AIM:

ALGORITHM:

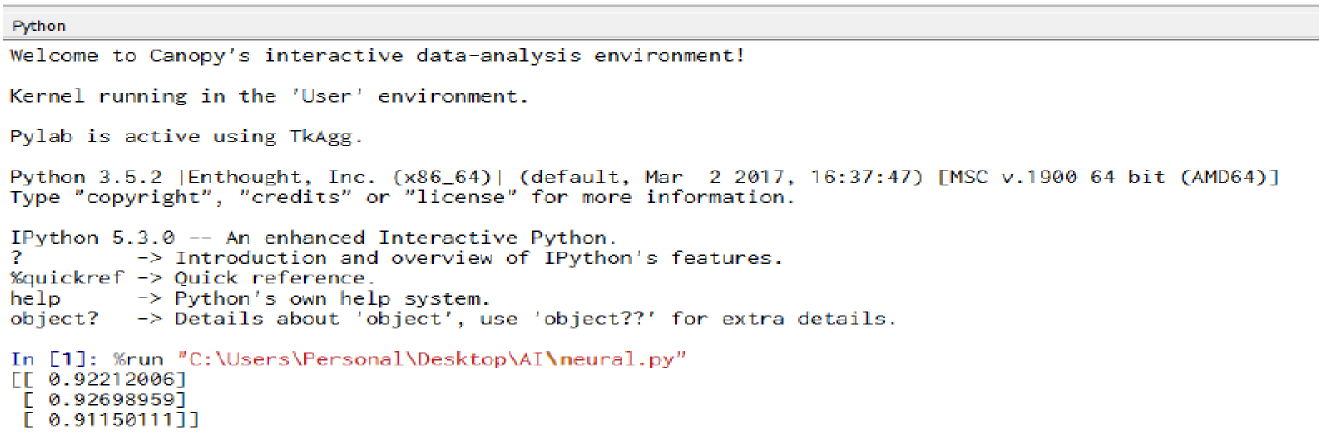
PROGRAM:

```
import numpy as np
import pandas as pd
from sklearn.metrics import confusion_matrix
from sklearn.cross_validation import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
# Function importing Dataset
def importdata():
balance_data = pd.read_csv(
'https://archive.ics.uci.edu/ml/machine-learning-'+
'databases/balance-scale/balance-scale.data',
sep= ',', header = None)
# Printing the dataswet shape
print ("Dataset Length: ", len(balance_data))
print ("Dataset Shape: ", balance_data.shape)
# Printing the dataset obseravtions
print ("Dataset: ",balance_data.head())
return balance_data
# Function to split the dataset
def splitdataset(balance_data):
# Seperating the target variable
X = balance_data.values[:, 1:5]
Y = balance_data.values[:, 0]
# Spliting the dataset into train and test
X_train, X_test, y_train, y_test = train_test_split(
X, Y, test_size = 0.3, random_state = 100)
return X, Y, X_train, X_test, y_train, y_test
# Function to perform training with giniIndex.
def train_using_gini(X_train, X_test, y_train):
```

```
clf_gini = DecisionTreeClassifier(criterion = "gini",
random_state = 100,max_depth=3, min_samples_leaf=5)
# Performing training
clf_gini.fit(X_train, y_train)
return clf_gini
# Function to perform training with entropy.
def tarin_using_entropy(X_train, X_test, y_train):
# Decision tree with entropy
clf_entropy = DecisionTreeClassifier(
criterion = "entropy", random_state = 100,
max_depth = 3, min_samples_leaf = 5)
# Performing training
clf_entropy.fit(X_train, y_train)
return clf_entropy
# Function to make predictions
def prediction(X_test, clf_object):
# Predicton on test with giniIndex
y_pred = clf_object.predict(X_test)
print("Predicted values:")
print(y_pred)
return y_pred
# Function to calculate accuracy
def cal_accuracy(y_test, y_pred):
print("Confusion Matrix: ",
confusion_matrix(y_test, y_pred))
print ("Accuracy : ",
accuracy_score(y_test,y_pred)*100)
print("Report : ",
classification_report(y_test, y_pred))
# Driver code
def main():
data = importdata()
X, Y, X_train, X_test, y_train, y_test = splitdataset(data)
```

```
clf_gini = train_using_gini(X_train, X_test, y_train)
clf_entropy = tarin_using_entropy(X_train, X_test, y_train)
print("Results Using Gini Index:")
# Prediction using gini
y_pred_gini = prediction(X_test, clf_gini)
cal_accuracy(y_test, y_pred_gini)
print("Results Using Entropy:")
# Prediction using entropy
y_pred_entropy = prediction(X_test, clf_entropy)
cal_accuracy(y_test, y_pred_entropy)
# Calling main function
if __name__=="__main__":
main()
```

OUTPUT:



```
Welcome to Canopy's interactive data-analysis environment!

Kernel running in the 'User' environment.

Pylab is active using TkAgg.

Python 3.5.2 |Enthought, Inc. (x86_64)| (default, Mar  2 2017, 16:37:47) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 5.3.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

In [1]: %run "C:\Users\Personal\Desktop\AI\neural.py"
[[ 0.92212006]
 [ 0.92698959]
 [ 0.91150111]]
```

SAVEETHA SCHOOL OF ENGINEERING DEPARTMENT OF CSE RUBRICS			
SLNO	**INDEX**	**MAXIMUM MARKS**	**MARKS AWARDED**
1.	**AIM**		
2.	**ALGORITHM**		
3.	**PROGRAM**		
4.	**OUTPUT**		
5.	**RESULT**		
6.	**OBSERVATION**		
7.	**VIVA**		
8	**TOTAL**		
SIGNATURE			

RESULT:

EX.NO: **DATE:**	**SEO WEBPAGE USING WORDPRESS**

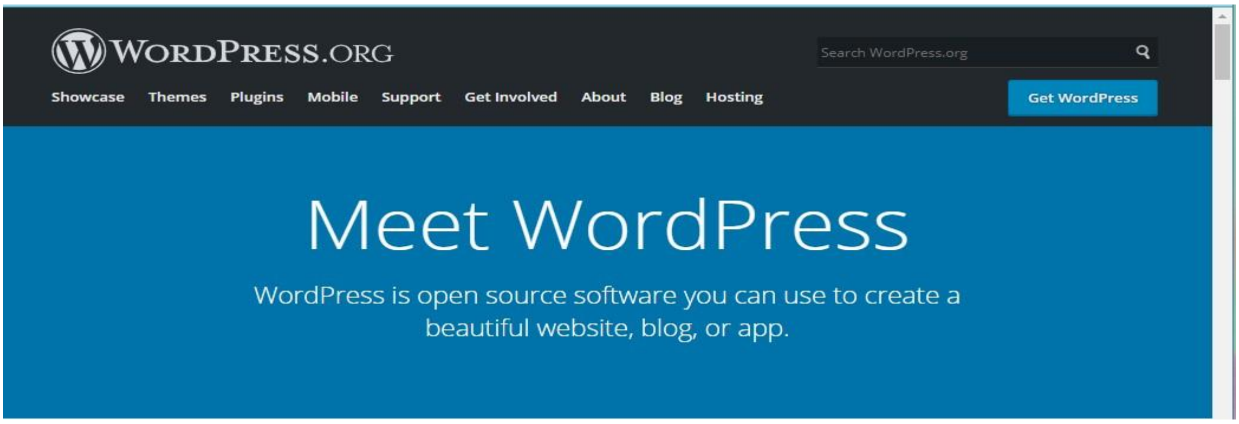
STEP I:- CHOOSE WORDPRESS AS YOUR WEBSITE PLATFORM

there are many website platforms that you can use when building a new site – Content Management Systems (CMS) is what they’re usually called.

The idea of a CMS is to give you some easy-to-use tools so that you’re able to edit your site’s content without any knowledge of coding. For the most part – from the user’s pointofview–thoseCMSlook much like the familiar interfaces at Facebook or Google Docs. You basically create new pages or documents,andthenhavethempublishedto the web.

key details aboutWordPress:

- it’s open source
- it’s free
- it’s the ultimate DIY solution for website building
- it’s extra versatile – can run any type of website
- it’s fast, optimized, and secure
- it’s SEO-ready – makes promotion easier



Meet WordPress

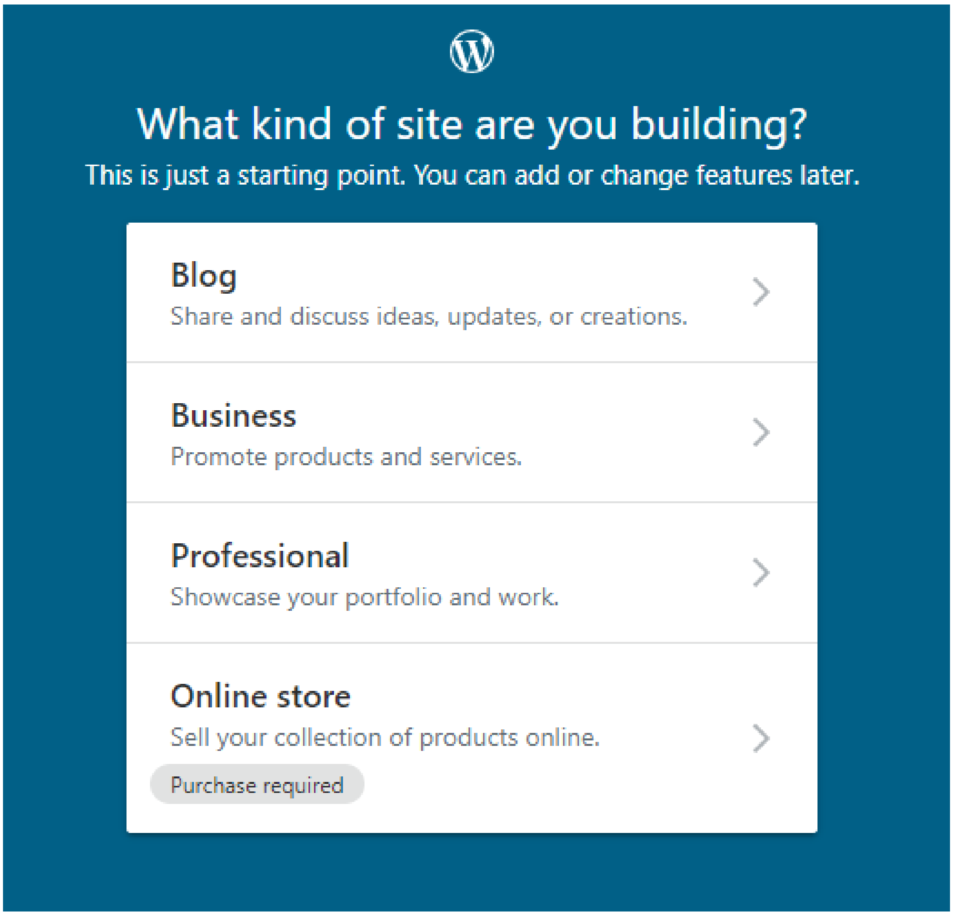
WordPress is open source software you can use to create a beautiful website, blog, or app.

STEP 2: PICK A NAME FOR YOUR WEBSITE

there are nearly 2 billion websites online on the web. Meaning, staying original can be quite challenging. It's a really good idea to construct our website's name (and thus your domain name) around either the name of your organization (the most obvious approach) or a phrase that's associated with the niche you're in, but with some added words for better brandability.

In short, a good domain name should be:

- brandable – unique sounding, like nothing else that's out there in the market
- easy to memorize
- short – those are also easier to memorize
- easy to type and hard to mix up – you don't want people to be wondering how to spell your site's name
- including niche-related keywords – for instance, if you do anything with , it would be cool to have "pizza" somewhere in the name of the site; it works the same in non-pizza industries aswell.



What kind of site are you building?

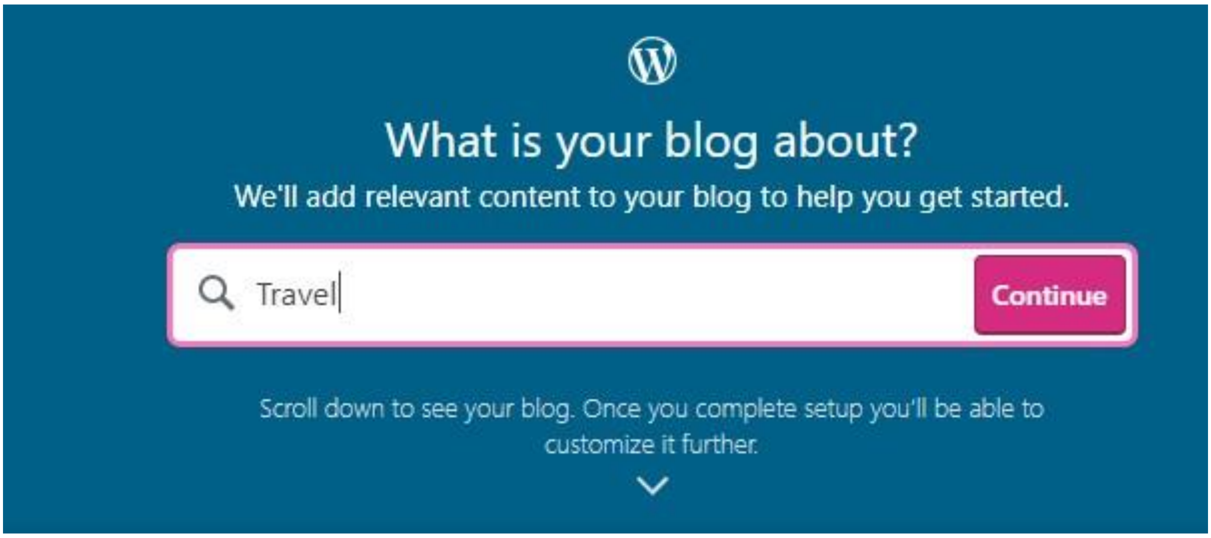
This is just a starting point. You can add or change features later.

Blog
Share and discuss ideas, updates, or creations.

Business
Promote products and services.

Professional
Showcase your portfolio and work.

Online store
Sell your collection of products online.
Purchase required



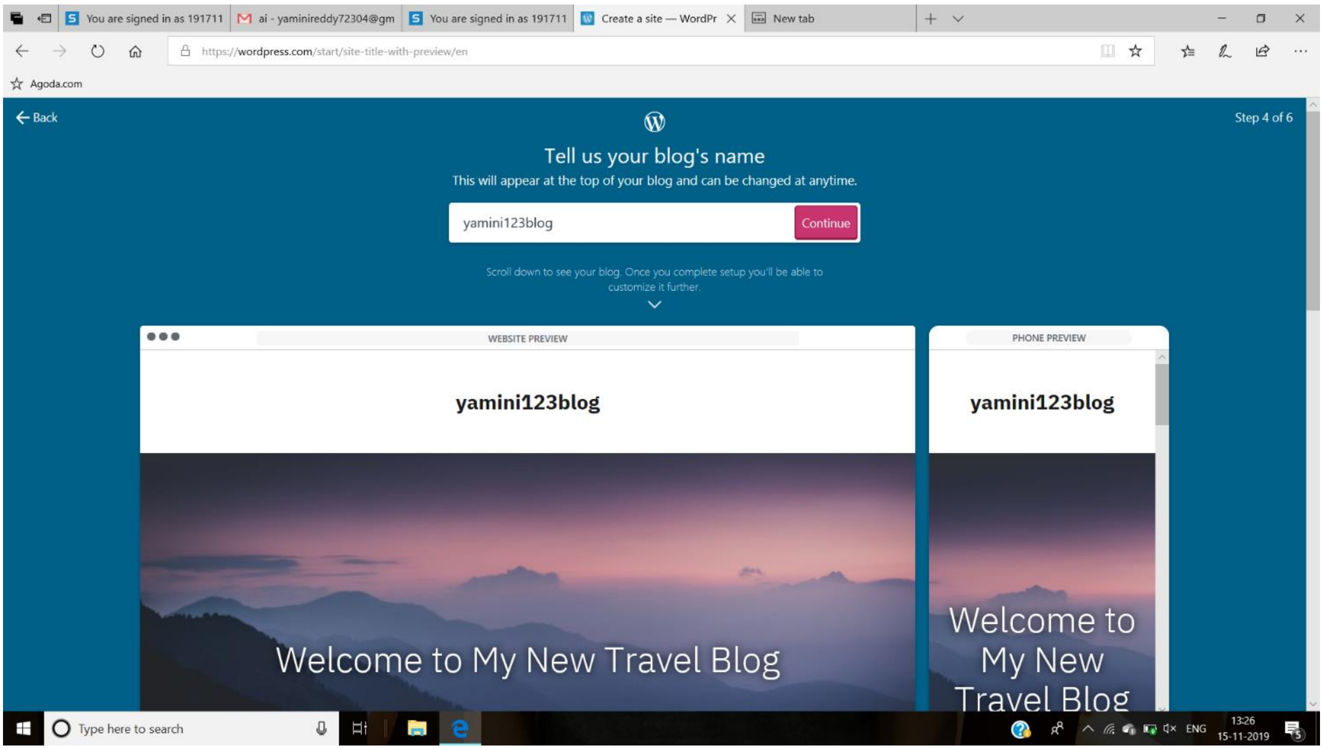
What is your blog about?

We'll add relevant content to your blog to help you get started.

Travel

Scroll down to see your blog. Once you complete setup you'll be able to customize it further.

STEP III:-CREATE A BLOG



You are signed in as 191711 | ai - yaminireddy72304@gm | You are signed in as 191711 | Create a site — WordPr | New tab

https://wordpress.com/start/site-title-with-preview/en

Tell us your blog's name

This will appear at the top of your blog and can be changed at anytime.

yamini123blog

Scroll down to see your blog. Once you complete setup you'll be able to customize it further.

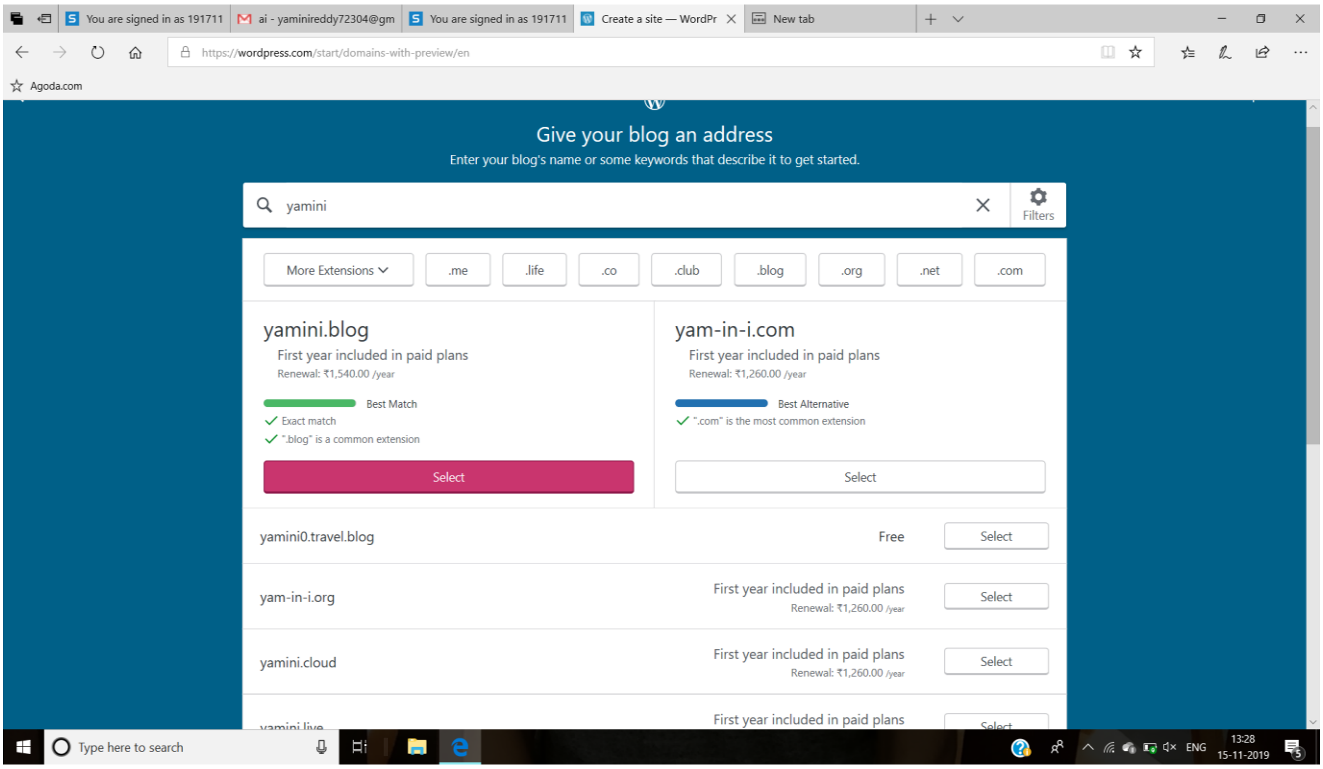
yamini123blog

Welcome to My New Travel Blog

yamini123blog

Welcome to My New Travel Blog

STEP IV:-UPDATE BLOG



https://wordpress.com/start/domains-with-preview/en

Give your blog an address

Enter your blog's name or some keywords that describe it to get started.

yamini

More Extensions | .me | .life | .co | .club | .blog | .org | .net | .com

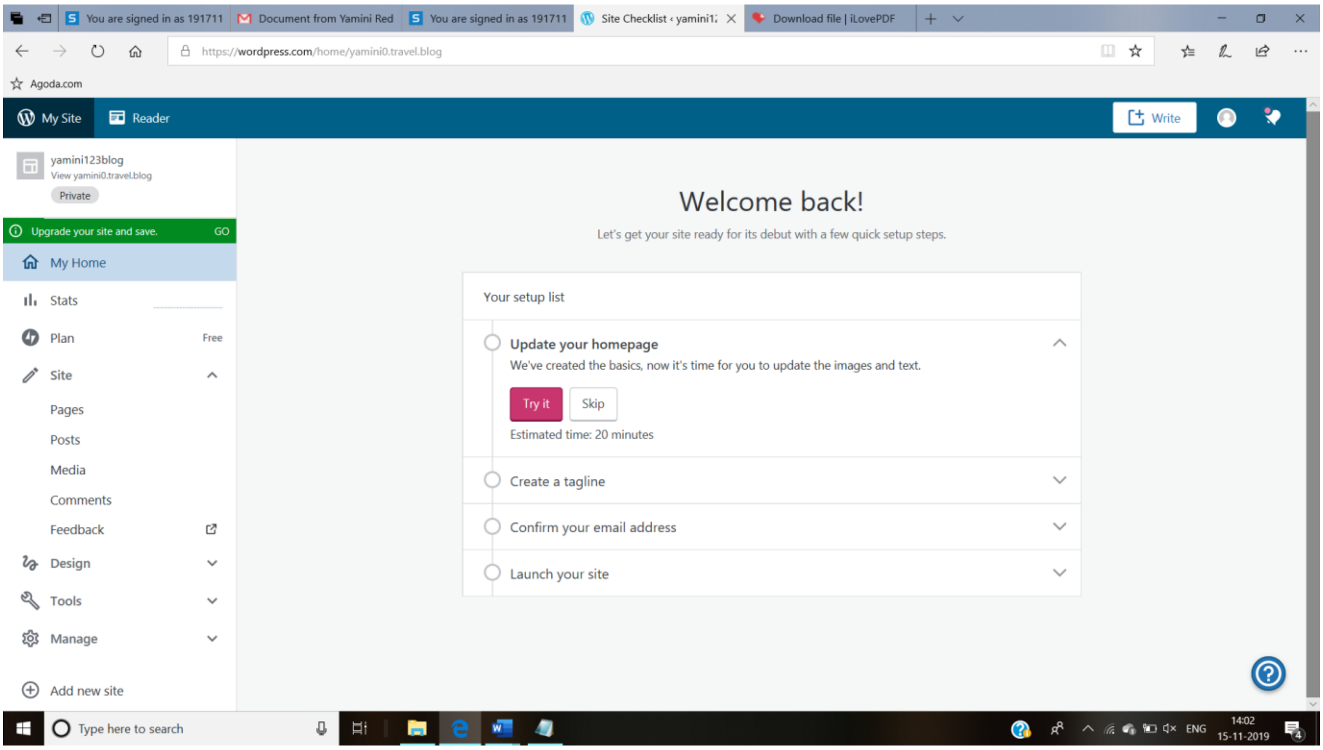
yamini.blog
First year included in paid plans
Renewal: ₹1,540.00 /year
Best Match
✓ Exact match
✓ ".blog" is a common extension
Select

yam-in-i.com
First year included in paid plans
Renewal: ₹1,260.00 /year
Best Alternative
✓ ".com" is the most common extension
Select

yamini0.travel.blog	Free	Select
yam-in-i.org	First year included in paid plans Renewal: ₹1,260.00 /year	Select
yamini.cloud	First year included in paid plans Renewal: ₹1,260.00 /year	Select
yamini.live	First year included in paid plans	Select

STEP V:CONSIDER STARTING A BLOG

A blog (as well as marketing through content – aka "TRAVEL" in general) is among the most effective ways to promote not only your website but also any products that you might want to sell through that website.



https://wordpress.com/home/yamini0.travel.blog

yamini123blog
View yamini0.travel.blog
Private

Upgrade your site and save. GO

Welcome back!

Let's get your site ready for its debut with a few quick setup steps.

Your setup list

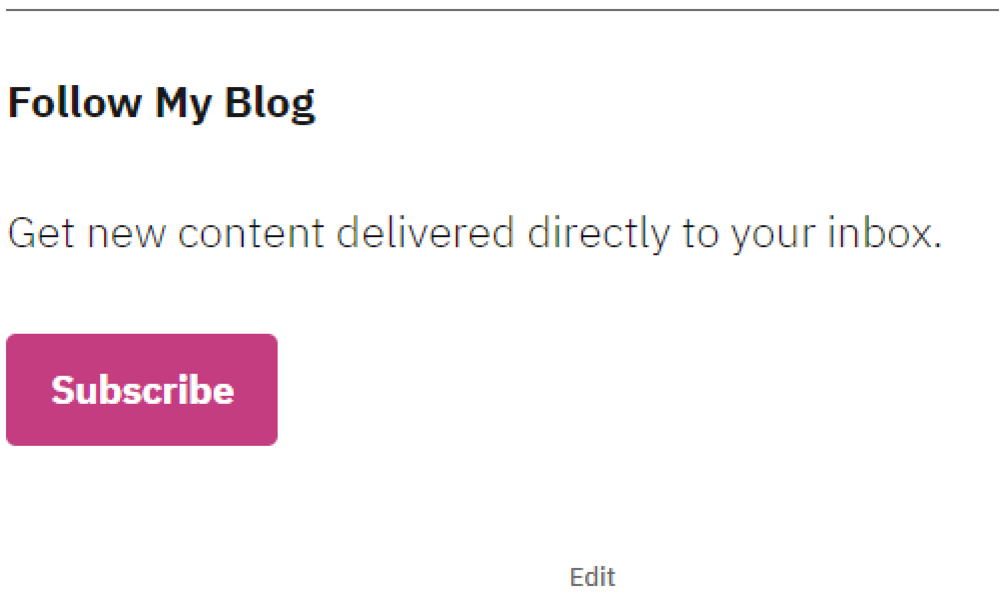
○ Update your homepage
We've created the basics, now it's time for you to update the images and text.
Try it | Skip
Estimated time: 20 minutes

○ Create a tagline

○ Confirm your email address

○ Launch your site

BLOG SITE : https://wordpress.com/home/yamini0.travel.blog



Follow My Blog

Get new content delivered directly to your inbox.

<table>
<tr><th colspan="4">SAVEETHA SCHOOL OF ENGINEERING
DEPARTMENT OF CSE
RUBRICS</th></tr>
<tr><td>SLNO</td><td>INDEX</td><td>MAXIMUM MARKS</td><td>MARKS AWARDED</td></tr>
<tr><td>1.</td><td>AIM</td><td></td><td></td></tr>
<tr><td>2.</td><td>ALGORITHM</td><td></td><td></td></tr>
<tr><td>3.</td><td>PROGRAM</td><td></td><td></td></tr>
<tr><td>4.</td><td>OUTPUT</td><td></td><td></td></tr>
<tr><td>5.</td><td>RESULT</td><td></td><td></td></tr>
<tr><td>6.</td><td>OBSERVATION</td><td></td><td></td></tr>
<tr><td>7.</td><td>VIVA</td><td></td><td></td></tr>
<tr><td>8</td><td>TOTAL</td><td></td><td></td></tr>
<tr><td colspan="2">SIGNATURE</td><td colspan="2"></td></tr>
</table>

RESULT: