

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name warke Purva Dilip

Expt. Title write a program for creating max-heap using
Class SYMCA Batch _____ Performed on INSERT and ADJUST /
HEAPIF

Roll No. _____ Expt. No. _____ Submitted on _____

Remarks Ge Returned on _____

write a algorithm for creating max-heap
using INSERT.

Procedure INSERT_MAX(A,n)

Description :- This procedure rearranges elements such that maximum element is should be at the root or at $A(1)$. Where $A(1:n)$ is array and 'n' is the number of element in array.

Declaration :- integer $A(1:n)$
integer i', j, n

$j \leftarrow n; i \leftarrow \lfloor n/2 \rfloor; item \leftarrow A(n)$

while $i > 0$ and $A(i) < item$, do

$A(j) \leftarrow A(i)$

$j \leftarrow i$

$i \leftarrow i/2$

repeat

$A(j) \leftarrow item$

end INSERT_MAX

for $i \leftarrow 2$ to n do

call INSERT_MAX

repeat.

Incomplete for :

- 1) Algorithm
- 2) Flow Chart
- 3) Programme Listing
- 4) Results
- 5) Comments

Write Algorithm for creating max-heap using
ADJUST, HEAPIFY

Procedure HEAPIFY-MAX(A,n)

Description :- Readjust the element in A(1:n)
'n' is the large non-leaf node

Declaration :- integer n, i

for $i \leftarrow \lfloor n/2 \rfloor$ to 1 by -1 do
| call ADJUST-MAX(A, i, n)
repeat.

end HEAPIFY-MAX

Procedure ADJUST-MAX(A, i, n)

Description :- The complete binary trees with roots
 $A(2*i)$ and $A(2*i+1)$ are combine with A to form
single heap $1 \leq i \leq n$ no node has address greater
than n.

Declaration :- integer i, j, n

j $\leftarrow 2*i$, item $\leftarrow A(i)$
while $j \leq n$ do
| if $i < n$ and $A(j) < A(j+1)$, then
| | j $\leftarrow j+1$
| endif
| if item $> A(j)$
| | then exit loop
| else
| | A($\lfloor j/2 \rfloor$) $\leftarrow A(j)$
| | j $\leftarrow 2*j$
| endif
repeat
A($\lfloor j/2 \rfloor$) \leftarrow item

end ADJUST-MAX

(3)

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name Warke Purva Dilip

Expt. Title Write a program for creating min heap using JNSFR

Class SYMCA Batch _____ Performed on _____ and AJUST / HEAPIFY

Roll No. _____ Expt. No. _____ Submitted on _____

Remarks _____ G.e _____ Returned on _____

Write a algorithm for creating min-heap using insert algorithm.

Procedure INSERT-MIN(A,n)

Description :- This procedure rearranges elements $A(1:n)$ such that minimum element is should be at the root or at $A(1)$. where n is the number of elements in array .

Declaration :- integer $A(1:n)$

integer i, j, n .

$j \leftarrow n$, $i \leftarrow \lfloor n/2 \rfloor$; item $\leftarrow A(n)$

while $i > 0$ and $A(i) > item$, do :

$A(j) \leftarrow A(i)$

$j \leftarrow i$

$i \leftarrow i/2$

repeat

$A(j) \leftarrow item$

END INSERT-MIN

for $i \leftarrow 2$ to n , do

 call INSERT-MIN(A,i)

repeat

Incomplete for :

1) Algorithm

2) Flow Chart

3) Programme Listing

4) Results

5) Comments

Write an algorithm for creating min heap using
ADJUST_HEAPIFY Algorithm.

Procedure ADJUST_HEAPIFY_MIN(A,n)

Description :-

This procedure readjust the elements in A such that to form an heap(min). where n is number of elements and A(1:n) is an array .

Declaration :- integer A,n,i

```
for i ← ⌊n/2⌋ to 1 by -1, do
    | call ADJUST-MIN (A,i,n)
repeat
end HEAPIFY-MIN
```

procedure ADJUST-MIN (A,i,n)

Description :-

This procedure ADJUST-MIN the nodes in tree whose root is at location i. n is the number of elements in an array of A(1:n)

Declaration :- integer A,n,
integer i,j,item

```
j ← 2*i, item ← A(i)
while j ≤ n, do
    if j < n and A(j) > A(j+1)
        | j ← j+1
    endif
    if item < A(j), then
        | exit loop
    else
        | A(⌊j/2⌋) ← A(j)
        | j ← j * 2
    endif
repeat
```

A(⌊j/2⌋) ← item

end ADJUST-MIN

DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON

Name Warke Purva Dilip

Expt. Title Write a program for creating Heap sort (Ascending)

Class SYMCA Batch _____ Performed on _____

Roll No. _____ Expt. No. _____ Submitted on _____

Remarks C _____ Returned on _____

write a algorithm to sort an array in ascending order using Heap Sort.

procedure HEAP-SORT(A,n)

Description :-

This procedure sorts the n elements of A(1:n). Heap sort rearrange them in-place into non-decreasing order. where A(1:n) is array contains 'n' number of elements.

Declaration :- integer A,n

integer i.

// Transforms the elements into a heap

call HEAPIFY(A,n)

// interchange the maximum with the elements at the end n and adjust root.

for i ← n to 2 by -1 do

 | call EXCHANGE(A(i),A(1))

 | call ADJUST(A, 1, i-1)

repeat

end HEAP-SORT

Procedure HEAPIFY(A,n)

Description :-

This procedure readjust the elements in A(1:n) such to form an heap(max)

Declaration :- integer n, i.

Incomplete for :

1) Algorithm

2) Flow Chart

3) Programme Listing

4) Results

5) Comments

```

for i ← ⌊n/2⌋ to 1 by -1, do
    |
    | call ADJUST(A,n)
    |
repeat
end HEAPIFY

```

Procedure ADJUST (A,i,n)

Description :-

This procedure is for binary tree whose root is at location i. n is the number of elements is an array A(1:n)

Declaration :- integer i,j , item.

```

j ← 2*i
item ← A(i)
while j ≤ n , do
    if j < n and A(j) < A(j+1) , then
        |
        | j ← j+1
    endif
    if item > A(j) , then
        |
        | exit loop
    else
        |
        | A(⌊j/2⌋) ← A(j)
        |
        | j ← j*2
    endif
repeat
A(⌊j/2⌋) ← item

```

end ADJUST

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name Warke Purva Dilip

Expt. Title write a program for creating Heap sort (Descending)

Class SYMCA Batch _____ Performed on _____

Roll No. _____ Expt. No. _____ Submitted on _____

Remarks _____ Returned on _____

write a algorithm to sort an array in descending order Using Heap sort.

Procedure HEAP-SORT (A,n)

Description :-

This procedure sorts the n elements of A(1:n). heap sort rearrange then in-place into decreasing order. where A(1:n) is array contain 'n' number of elements.

Declaration :- integer A,n
integer i.

~~// transform the elements into a heap~~
call HEAPIFY(A,n)
~~// interchange the minimum with the elements at the end n and adjust root.~~
for i ← n to 2 by -1 do
| call EXCHANGE(A(1), A(i))
| call ADJUST(A, 1, i-1)
repeat
end HEAP-SORT

procedure HEAPIFY(A,n)

Description :-

This procedure readjust the elements in A(1:n) such to form an heap (min)

Declaration :- integer n, i, A

Incomplete for:

- 1) Algorithm
- 2) Flow Chart
- 3) Programme Listing
- 4) Results
- 5) Comments

```

for i ← ⌊n/2⌋ to 1 by -1, do
    | call ADJUST(A, i, n)
repeat
end HEAPIFY

```

Procedure ADJUST (A, i, n)

Description :- This procedure ADJUST the nodes in tree whose root is at location i. 'n' is the number of elements in an array of A(1:n)

Declaration :- integer A, n.
integer i, j, item

```

j ← 2*i, item ← A(i)
while j < n, do
    if j < n and A(j) > A(j+1)
        | j ← j+1
    endif
    if item < A(j), then
        | exit loop
    else
        | A(⌊j/2⌋) ← A(j)
        | j ← j * 2
    endif
repeat
A(⌊j/2⌋) ← item

```

end ADJUST.

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name Warke Purva Dilip

Expt. Title write a program to find solution of knapsack instant

Class SYMCA Batch _____ Performed on _____

Roll No. _____ Expt. No. _____ Submitted on _____

Remarks SJ Returned on _____

write a algorithm for greedy knapsack method.

Procedure GREEDY_KNAPSACK(P, W, M, X, n)

assuming the data given is sorted according to P/w ratio.

Description :- $P(i:n)$ and $W(i:n)$ contain the profits and weights resp. of n th objects ordered so that $P(i)/W(i) \geq P(i+1)/W(i+1)$. M is the max of size and $X(i:n)$ is the solution vector.

Declaration :- real $P(1:n), W(1:n), X(1:n), M, Cu$
integer i, n .

```

 $x \leftarrow 0$ 
 $Cu \leftarrow M$ 
for  $i \leftarrow 1$  to  $n$  do
    if  $W(i) > Cu$ , then
        | exit loop
    else
        |  $x(i) \leftarrow 1$ 
        |  $Cu \leftarrow Cu - W(i)$ 
    endif
repeat
if  $i \leq n$ , then
    |  $x(i) \leftarrow Cu / W(i)$ 
endif

```

end GREEDY-KNAPSACK.

Incomplete for :

- 1) Algorithm
- 2) Flow Chart
- 3) Programme Listing
- 4) Results
- 5) Comments

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name Warka Purva Dilip

Expt. Title Write a program to find minimum - cost Spanning Trees.

Class SYMCA Batch _____ Performed on _____

Roll No. _____ Expt. No. _____ Submitted on _____

Remarks _____ Returned on _____

write a Algorithm to find minimum - cost Spanning Trees (Prim's & Kruskal's Algorithm)

2) Prim's Algorithm:-

Procedure PRIM (E, COST, n, T, mincost)

Description :— E is the set of edges in G. COST(n,n) is Cost adjacency matrix of an n vertex graph such that COST(i,j) is either a positive real number or $+\infty$ if no edge (i,j) exists. A minimum spanning tree is computed and stored as a set of edges in the array T(1:n,2), T(i,1), T(i,2) is an edge in the minimum cost spanning tree. The final cost is assigned to min cost.

Declaration :- real COST(n,n), mincost
integer NEAR ϵ , n, i, j, k, l,
T(1:n-1,2).

$(k, l) \leftarrow$ edge with mincost.

mincost \leftarrow COST(k,l)

$(T(1,1), T(1,2)) \leftarrow (k, l)$

// fill up near array.

for i \leftarrow 1 to n do

 if COST(i,l) $<$ COST(i,k)

 NEAR(i) \leftarrow l

 else

 NEAR(i) \leftarrow k

 endif

repeat

Incomplete for :

1) Algorithm

2) Flow Chart

3) Programme Listing

4) Results

5) Comments

$\text{NEAR}(k) \leftarrow 0$
 $\text{NEAR}(l) \leftarrow 0$

// find out remaining $(n-2)$ edges of the tree

for $i \leftarrow 2$ to $n-1$ do

 Let j be an index such that $\text{NEAR}(j) \neq 0$
 and $\text{COST}(j, \text{NEAR}(j))$ is minimum.

$(T(i,1), T(i,2)) \leftarrow (j, \text{NEAR}(j))$

$\text{mincost} \leftarrow \text{mincost} + \text{COST}(j, \text{NEAR}(j))$

$\text{NEAR}(j) \leftarrow 0$

// update NEAR array

for $k \leftarrow 1$ to n do

 if $\text{NEAR}(k) \neq 0$ and $\text{COST}(k, \text{NEAR}(k)) >$
 $(\text{COST}(k, j))$

$\text{NEAR}(k) \leftarrow j$

 endif

repeat

repeat

if $\text{mincost} \geq \infty$, then

 print ("NO spanning Tree")

endif

end PRIM

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name Worke Purva Dilip

Expt. Title Write a program to implement Union & find operation

Class SYMCA Batch _____ Performed on _____

Roll No. _____ Expt. No. _____ Submitted on _____

Remarks _____ Returned on _____

Procedure $U(i, j)$

Description :- Replace the disjoint sets with roots $i, j, i \neq j$ by their union.

Declaration :- integer i, j

$PARENT(i) \leftarrow j$

end U

Procedure $F(i)$

Description :- find the root of the tree containing ele i .

Declaration :- integer i, j .

$j \leftarrow i$

while $PARENT(j) > 0$

$j \leftarrow PARENT$

repeat

return(j)

end F

Procedure $UNION(i, j)$

Description :- UNION sets with roots i and j .
 $i \neq j$, using the weighting rule.

$PARENT(i) = -COUNT(i)$ &

$PARENT(j) = -COUNT(j)$

Declaration :- integer i, j, x

Incomplete for :

- 1) Algorithm
- 2) Flow Chart
- 3) Programme Listing
- 4) Results
- 5) Comments

```

 $x \leftarrow \text{PARENT}(i) + \text{PARENT}(j)$ 
if ( $\text{PARENT}(i) > \text{PARENT}(j)$ ) then
     $\text{PARENT}(i) \leftarrow j;$ 
     $\text{PARENT}(j) \leftarrow x;$ 
else
     $\text{PARENT}(j) \leftarrow i;$ 
     $\text{PARENT}(i) \leftarrow x;$ 
endif
end UNION

```

procedure FIND(i)

Description :- Find the root of the tree containing element i , use the collapsing rule to collapse all nodes from i to the root.

Declaration :- integer i, j, k

```

 $j \leftarrow i$  // find first root of tree
while  $\text{PARENT}(j) > 0$ 
     $j \leftarrow \text{PARENT}(j);$ 
repeat
 $k \leftarrow i$ 
while  $k \neq j$  do // collapse nodes from  $k$  to root  $j$ 
    temp  $\leftarrow \text{PARENT}(k);$ 
     $\text{PARENT}(k) \leftarrow j;$ 
     $k \leftarrow \text{temp};$ 
repeat
return(j)
end FIND

```

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name Warke Purva Dilip

Expt. Title write a program for Merge Sort.

Class SYMCA Batch _____ Performed on _____

Roll No. _____ Expt. No. _____ Submitted on _____

Remarks _____ Returned on _____

Write an Algorithm for merge sort.

Procedure MERGE-SORT (low, high)

Description :- A[low:high] is a global array to be sorted. In this case the list is already sorted.

Declaration :- integer A, low, high

if low < high , then

mid \leftarrow (low + high)/2

call MERGE-SORT (A , low, mid)

call MERGE-SORT (A , mid+1, high)

Call MERGE (A , low, mid , high)

end if

end MERGE-SORT

Procedure MERGE (A,low,mid ,high)

Description :- This process merge two sublist A(low:mid) & B(mid+1 : high) it uses auxiliary B(low:high) & sorted.

Declaration :- Global A(low:mid) & A(mid+1, high)
integer A, low , mid , high
local integer i,j,k

$i \leftarrow low$

$j \leftarrow mid+1$

$k \leftarrow low$

Incomplete for :

1) Algorithm

2) Flow Chart

3) Programme Listing

4) Results

5) Comments

```
while i <= mid and j <= high, do
    if A(i) < A(j), then
        B(k) ← A(i)
        i ← i + 1
    else
        B(k) ← A(j)
        j ← j + 1
    endif
repeat      k ← k + 1
if i <= mid
    while i <= mid, do
        B(k) ← A(i)
        i ← i + 1
        k ← k + 1
    repeat
else
    while j <= high, do
        B(k) ← A(j)
        j ← j + 1
        k ← k + 1
    repeat
endif
for k ← low to high, do
    A(k) ← B(k)
repeat
end MERGE
```

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name Warke Purva Dilip

Expt. Title Write a program to find Minimum - Cost Spanning Tree.

Class SYMCA Batch _____ Performed on _____

Roll No. _____ Expt. No. _____ Submitted on _____

Remarks _____ Returned on _____

Procedure KRUSKAL(E, cost, n, t, mincost)

Description :- E is the set of edges in G.
G has n vertices. COST(u,v) is the COST of edge COST(u,v).
t is the set of edges in the minimum spanning tree. & mincost is COST.

Declaration :- real mincost, COST(1:n, 1:n)
integer PARENT(1:n),
T(1:n-1, 2), n.

Construct a heap out of the edge costs using Heapify;

for i ← 1 to n do PARENT(i) ← -1;

i ← emincost ← 0;

while i < n-1 and HEAP is not empty do

Delete a minimum cost edge (u,v) from the heap and reheapify using Adjust;

j ← Find(u); k ← find(v);

if (j ≠ k) then

i ← i + 1;

t(i, 1) ← t(i, 2) ← v;

mincost ← mincost + cost(u,v);

UNION(j, k)

endif

repeat

if i ≠ n-1, then

print ("NO spanning Tree")

endif

end KRUSKAL.

Incomplete for :

1) Algorithm

2) Flow Chart

3) Programme Listing

4) Results

5) Comments

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name warke Purva Dilip

Expt. Title Write a program to find shortest Path using single source shortest path.
Class SY MCA Batch _____ Performed on _____

Roll No. _____ **Expt. No.** _____ **Submitted on** _____

Remarks _____ Returned on _____

Write a algorithm to find shortest Path using Single Source Shortest Path.

Procedure SHORTEST-PATH (v , COST, DIST, n)

Description:-

$\text{DIST}(j)$, j is $1 \leq j \leq n$ is set to the length of the shortest path from vector v to vertex j in a diagraph G with n vertices. $\text{DIST}(v)$ is set to zero. G is represented by its cost adjacency matrix $\text{COST}(1:n, 1:n)$.

Declaration :- boolean S(1:n),
real COST(1:n,1:n), DIST(1:n)
integer n, u, v, w, num, i.

// Initialize set S to empty & DIST using current edges.

for $i \leftarrow 1$ to n do

$s(i) \leftarrow 0;$

$\text{DIST}(i) \leftarrow \text{COST}(v, i)$

repeat

$s(v) \leftarrow 1;$

$\text{DIST}(v) \leftarrow 0$

for num \leftarrow 2 to $n-1$ do

1) choose u from among those vertices
 not in S such that
 $\text{PSET}(u) = \min\{\text{PSET}(w)\}$ and $s(u) = \alpha^*$

$S(u) \leftarrow 1$ // put u in S .

Incomplete for :

- 1) Algorithm
 - 2) Flow Chart
 - 3) Programme Listing
 - 4) Results
 - 5) Comments

for (each w adjacent to u with $S(w)=0$) do
 // update distances.
 if ($DIST(w) > DIST(u) + COST(u,w)$) then
 $DIST(w) \leftarrow DIST(u) + COST(u,w);$
 endif
repeat
repeat
end SHORTEST_PATH

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name Warke Purva Dilip

Expt. Title Write program for Searching element from given array using binary search for n=1000, 2000, 3000 find exact time of execution
Class SYMCA Batch 1 Performed on 10/10/2023

Roll No. _____ Expt. No. _____ Submitted on _____

Remarks _____ Returned on _____

BINARY SEARCH :-

Procedure BINSEARCH(A,n,x,j)

Description: - Given an Array A[1:n] of elements in nondecreasing order, $n \geq 0$, determine whether x is present and if so, return j such that $x = A[j]$; else return 0.

Declaration: - integer low, high, mid, j, n

low $\leftarrow 1$; high $\leftarrow n$

while low \leq high, do

mid $\leftarrow [(low + high)/2]$

case

: $x < A(mid)$: high $\leftarrow mid - 1$

: $x > A(mid)$: low $\leftarrow mid + 1$

: else

j $\leftarrow mid$; return

endcase

repeat

j $\leftarrow 0$

END BINSEARCH

Incomplete for :

- 1) Algorithm
- 2) Flow Chart
- 3) Programme Listing
- 4) Results
- 5) Comments

Procedure BIN_SEARCH1 (A, n, x, j)

Description :-

Declaration :- integer low, high, mid, j, n

low \leftarrow 1; high \leftarrow n + 1

while low < high - 1, do

 mid $\leftarrow \lfloor (\text{low} + \text{high})/2 \rfloor$

 if x < A(mid), then

 high \leftarrow mid

 else

 low \leftarrow mid

 endif

repeat

 if x = A(low), then

 j \leftarrow low

 else

 j \leftarrow 0

 endif

END BIN_SEARCH1

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name Warke Purva Dilip

Name Warke , Date 01/01/2023
Expt. Title Write a program to find minimum & maximum from a given array
Perfomed on S Y M C A

Class SYMCA Batch _____ Performed on _____

Roll No. _____ Expt. No. _____ Submitted on _____

Remarks _____ Returned on _____

Wavelength minimum and

write an algorithm to find minimum and maximum from given array.

Procedure MAXMIN (p,q , max , min)

Description :- A(1:n) is a global array.

The effect is to set max and min to the largest and smallest values in $A[P:q]$, respectively.

Declaration :- integer p,q
1 ≤ p ≤ q ≤ n

If $p = q$, then
 $\max \leftarrow \min \leftarrow A(p)$

else

if $p = q - 1$, then

if $A(p) > A(q)$,

$\max \leftarrow A(p)$

1

$$m \alpha \leftarrow A(g)$$

$\min \leftarrow A(p)$

endif

else.

$$m \leftarrow (p+q)/2$$

call MAXMIN(p,m,fmax,fmin)

call MAXMIN ($m+1, q, h_{\max}, h_{\min}$)

if $f_{max} > h_{max}$, then

10

$$m_9 x \leftarrow b m_9 x$$

endif

Incomplete for :

- 1) Algorithm
 - 2) Flow Chart
 - 3) Programme Listing
 - 4) Results
 - 5) Comments

```
    if fmin < hmin , then  
        min ← fmin  
    else  
        min ← hmin  
    endif  
endif  
endif
```

end MAXMIN

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name Warke Purva Dilip

Expt. Title Write a program to implement breadth & depth first search

Class SYMCA Batch B4 Performed on _____

Roll No. 136 Expt. No. _____ Submitted on _____

Remarks _____ Returned on _____

Breadth first search :-

Procedure BFS(v)

Object Description:- A breadth first search of G is carried out beginning at vertex v . All vertices visited are marked as VISITED(i). The graph G and array VISITED are global and VISITED is initialised to 0.

Declaration :-

VISITED(v) $\leftarrow 1$

$u \leftarrow v$

Initialize Q to be empty

loop

for all vertices w adjacent from u do

 if $VISITED(w) = 0$, then

 call ADDQ(w, Q)

 $VISITED(w) \leftarrow 1$

 endif

repeat

 if Q is empty then

 return

 end if

 call DELETEQ(u, Q)

repeat

end BFS.

Incomplete for :

- 1) Algorithm
- 2) Flow Chart
- 3) Programme Listing
- 4) Results
- 5) Comments

Depth First Search (Recursive) :-

Procedure DFS(v)

Description :- Given an undirected or directed graph $G = (V, E)$ with n vertices & an array VISITED(n) initially set to 0.

Declaration :-

VISITED(v) $\leftarrow 1$

for each vertex w adjacent from v do

 if VISITED(w) = 0, then

 call DFS(w)

 endif

repeat

end DFS

Depth First Search (Non-Recursive) :-

Procedure NR-DFS(v)

VISITED(v) $\leftarrow 1$

$u \leftarrow v$

//initialise STACK to be empty

loop

 for all w adjacent from u do

 PUSH(w)

 VISITED(w) $\leftarrow 1$

 repeat

 if STACK $= u$ empty, then

 return

 else

$u \leftarrow \text{POP}()$

 endif

 repeat

end NR-DFS

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name Warke Purva Dilip

Expt. Title write a program to find shortest path using all pair path

Class SYMCA Batch 84 Performed on _____

Roll No. 136 Expt. No. _____ Submitted on _____

Remarks _____ Returned on _____

write an Algorithm for find shortest path using all pair path.

Procedure ALL-PATHS (COST, A, n)

Description :- COST(n, n) is the cost adjacency matrix of a graph with n vertices; A(i, j) is the cost of a shortest path from v_i to v_j . $COST(i, i) = 0$, $1 \leq i \leq n$

Declaration :- integer i, j, k, n;
real COST(n, n) A(n, n)

```

for i ← 1 to n do
    for j ← 1 to n do
        | A(i, j) ← COST(i, j)
repeat
repeat
for k ← 1 to n do
    for i ← 1 to n do
        for j ← 1 to n do
            | A(i, j) ← min{A(i, j),
                           A(i, k) + A(k, j)}
repeat
repeat
repeat
repeat

```

Incomplete for :

1) Algorithm

2) Flow Chart

3) Programme Listing

4) Results

5) Comments

end ALL-PATHS

**DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF MANAGEMENT AND RESEARCH, JALGAON**

Name Warke Purva Dilip

Expt. Title write a program to find all solutions for 8-queen problem using backtracking.
Class SYMCA Batch B4 Performed on _____

Roll No. 136 Expt. No. _____ Submitted on _____
Remarks _____ Returned on _____

write an algorithm to find all solutions for 8-queen problem using backtracking.

Procedure NQUEENS(n)

Description :- Using backtracking, this procedure prints all possible placements of n queens on an $n \times n$ chessboard so that they are nonattacking

Declaration :- integer $k, n, x(1:n)$

$k \leftarrow 1; x(k) \leftarrow 0$ // start with first row & 0th column

while $k > 0$ do // try all possible solutions.

$x(k) \leftarrow x(k) + 1$

while $x(k) \leq n$ and NOT PLACE(k) do

$x(k) \leftarrow x(k) + 1$ //try next column & pos.

repeat

if $x(k) \leq n$, then

if $k = n$, then

print(x)

else

$k \leftarrow k + 1$

$x(k) \leftarrow 0$

endif

else ~~else~~

$k \leftarrow k - 1$

endif

repeat

end NQUEENS

Incomplete for :

- 1) Algorithm
- 2) Flow Chart
- 3) Programme Listing
- 4) Results
- 5) Comments

Procedure PLACE(k)

Description :- return true if a queen can be placed in k^{th} row and $x(k)^{\text{th}}$ column. otherwise it returns false. x is a global array whose first k values having set $\text{ABS}(x)$ returns absolute value of x

Declaration :- global $x(1:n)$

 integer i,k

for $i \leftarrow 1$ to $k-1$. do

 if $(x(i) = x(k))$ OR

$\text{ABS}(i-k) = \text{ABS}(x(i)-x(k))$, then

 return (false)

 endif

repeat

return (true)

end PLACE