www.MyPlacementPrep.com

Pro Material Series

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

1. Addition Of Two Matrices In C:

1. 2. 3. 4. 5. 6. 7. 8. 9. 10. 11. 12. 13. 14. 15. 16. 17. 18. 19. 20. 21. 22. 23. 24. 25. 26. 27. 28. 29. 30. 31. 32.

```
#include <stdio.h>

int main() {

int m, n, c, d, first[1 0][10], second[10][10], sum[ 10][10];

printf("Enter the number of rows and columns of matrix\n"); scanf( "%d%d", & m, &n);
printf("Enter the elements of first matrix\n");

for (c = 0; c < m; c++) for (d = 0; d < n; d++)

scanf("%d", &first[c][d]);

printf("Enter the elements of second matrix\n");

for (c = 0; c < m; c++) for (d = 0 ; d < n; d++)

scanf("%d", &second[ c][d]);

printf("Sum of entered matrices:-\n");

for (c = 0; c < m; c++) { for (d = 0 ; d < n; d++) {

sum[c][d] = first[c][d] + second[c][d];

printf("%d\t", sum[c][d]); }
```

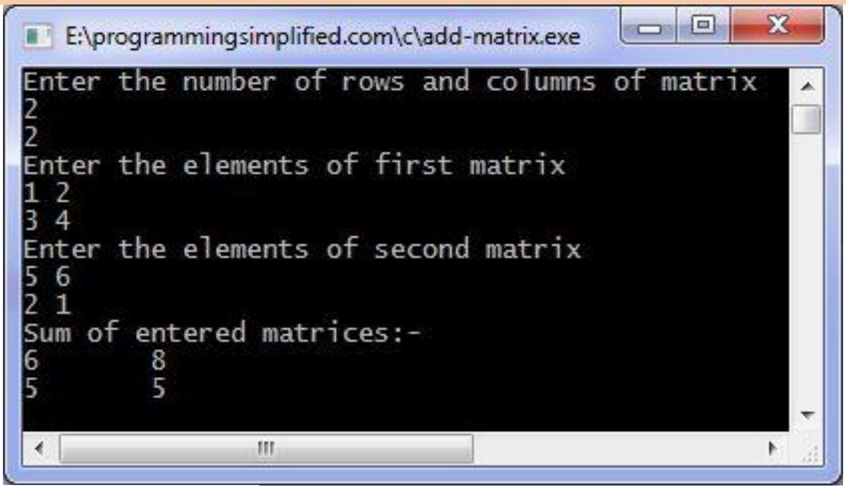
```
printf("\n"); }

return 0; }
```

Wipro Elite NLTH Coding / Programming Questions

Output:

2. Program to find the average of n (n < 10) numbers using arrays

```
#include <stdio.h> int main()
{

int marks[10], i, n, sum = 0, average; printf("Enter n: ");
scanf("%d", &n);
for(i=0; i<n; ++i)

{
printf("Enter number%d: ",i+1); scanf("%d", &marks[i]);
sum += marks[i];

     }
     average = sum/n;

printf("Average = %d", average); return 0;

}
```



FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

Output:

```
Enter n: 5
Enter number1: 45
Enter number2: 35
Enter number3: 38
Enter number4: 31
Enter number5: 49
Average = 39
```

3. C program To Implement Linked List

1. 2. 3. 4. 5. 6. 7. 8. 9. 10. 11. 12. 13. 14. 15. 16. 17. 18. 19. 20. 21. 22. 23. 24.

```
#include <stdio.h> #include <stdlib.h>

struct node {
int data;
struct node *next;
```

```
};

struct node *start = NULL; void insert_at_begin(int) ; void
insert_at_end(int); void traverse();

void delete_from_begin(); void delete_from_end(); int count = 0;

int main () {
int input, data;

for (;;) {
printf("1. Insert an element at beginning of linked list.\n"); printf("2.
Insert an element at end of linked list.\n"); printf("3. Traverse linked
list.\n");
printf("4. Delete element from beginning.\n");
```

FRESHERS NOW

25. 26. 27. 28. 29. 30. 31. 32. 33. 34. 35. 36. 37. 38. 39. 40. 41. 42. 43. 44. 45. 46. 47. 48. 49. 50. 51. 52. 53. 54. 55. 56. 57. 58. 59. 60.

Wipro Elite NLTH Coding / Programming Questions

```
printf("5. Delete element from end.\n"); printf("6. Exit\n");

scanf("%d", &input);

if (input == 1 ) {
printf("Enter value of element\n"); scanf("%d", &data);
insert_at_begin(d ata);

}
else if (input == 2) {

printf("Enter value of element\n"); scanf("%d", &data); insert_at_end(
data);
```

```
}
else if (input == 3)

traverse();
else if (input == 4)

delete_from_begin(); else if (input == 5)

delete_from_end(); else if (input == 6)

break; else

printf("Please enter valid input.\n"); }

return 0; }

void insert_at_begin(int x) { struct node *t;

t = (struct node*)malloc(sizeof(struct node)); count++;
```


61. 62. 63. 64. 65. 66. 67. 68. 69. 70. 71. 72. 73. 74. 75. 76. 77. 78. 79. 80. 81. 82. 83. 84. 85. 86. 87. 88. 89. 90. 91. 92. 93. 94. 95. 96.

Wipro Elite NLTH Coding / Programming Questions

```
if (start == NULL) { start = t; start->data = x; start->next = NULL;
return;

}

t->data = x; t->next = start; start = t;

}
void insert_at_end(int x) {
```

```
struct node *t, *temp;
t = (struct node*)malloc(sizeof(struct node));

count++;

if (start == NULL) { start = t; start->data = x; start->next = NULL;
return;

}

temp = start;

while (temp->next != NULL) temp = temp->next;

temp->next = t; t->data = x; t->next = NULL;

}
void traverse() {
```

Wipro Elite NLTH Coding / Programming Questions

```
97. struct node *t; 98.
99. t = start;
100.

101.        if (t == NULL) {
102.        printf("Linked list is empty.\n");
103.        return;
104.        }

105.
106. printf("There are %d elements in linked list.\n", count) ; 107.

108.        while (t->next != NULL) {
109.        printf("%d\n" , t- >data);
110.        t = t->next;
```

```
}
printf("%d\n", t->data);
}

void delete_from_begin() {
struct node *t;
int n;

if (start == NULL) {
printf("Linked list is already empty.\n");
return;
}

n = start->d ata;
t = start->next;
free(start);
start = t;
count--;

printf("%d deleted from beginning successfully.\n", n);
}
```



```
void delete_from_end() {
struct node *t, *u;
```

```
int n;

if (start == NULL) {
printf("Linked list is already empty.\n");
return;
}

count--;
if

}

t = start;

while (t->next != NULL) {
u = t;
t = t->next;
}

n = t->data;
u->next = NULL;
free(t);

printf("%d deleted from end successfully.\n", n) ;
}
```

4. Operations On Linked List

```
(start->next == NULL) { n = start->data; free(start);
start = NULL;

printf("%d deleted from end successfully.\n", n); return;
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
#include<stdio.h> #include<stdlib.h>

struct node {

int data;

struct node *next; };

void display(struct node* head) {

struct node *temp = head; printf("\n\nList elements are - \n");
while(temp != NULL)
{

printf("%d --->",temp->data); temp = temp->next;

} }

void insertAtMiddle(struct node *head, int position, int value) {
struct node *temp = head;
struct node *newNode;
newNode = malloc(sizeof(struct node));

newNode->data = value;

int i;

for(i=2; inext != NULL) { temp = temp->next;
}

}
newNode->next = temp->next; temp->next = newNode;

}
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
void insertAtFront(struct node** headRef, int value) { struct node*
head = *headRef;

struct node *newNode;
newNode = malloc(sizeof(struct node)); newNode->data = value;
newNode->next = head;
head = newNode;

    *headRef = head;
}

void insertAtEnd(struct node* head, int value){ struct node *newNode;
newNode = malloc(sizeof(struct node)); newNode->data = value;

newNode->next = NULL;

struct node *temp = head; while(temp->next != NULL){

temp = temp->next; }

temp->next = newNode; }

void deleteFromFront(struct node** headRef){ struct node* head =
*headRef;
head = head->next;
*headRef = head;

}

void deleteFromEnd(struct node* head){ struct node* temp = head;
while(temp->next->next!=NULL){

temp = temp->next; }
```

Wipro Elite NLTH Coding / Programming Questions

```
temp->next = NULL; }

void deleteFromMiddle(struct node* head, int position){ struct node*
temp = head;
int i;
for(i=2; inext != NULL) {

temp = temp->next;

} }

temp->next = temp->next->n ext; }

int main() {
/* Initialize nodes */ struct node *head;
struct node *one = NULL; struct node *two = NULL; struct node *three
= NULL;

/* Allocate memory */
one = malloc(sizeof(struct node)); two = malloc(sizeof(struct node));
three = malloc(sizeof(struct node));

/* Assign data values */ one->data = 1; two->data = 2; three->data =
3;

/* Connect nodes */ one->next = two; two->next = three; three->next =
NULL;
```

Wipro Elite NLTH Coding / Programming Questions

```
/* Save address of first node in head */ head = one;

display(head); // 1 --->2 --->3 ---> insertAtFront(&head, 4);

display(head); // 4 --->1 --->2 --->3 ---> deleteFromFront(&head);

display(head); // 1 --->2 --->3 ---> insertAtEnd(head, 5);

display(head); // 1 --->2 --->3 --->5 ---> deleteFromEnd(head);

display(head); // 1 --->2 --->3 --->

int position = 3;
insertAtMiddle(head, position, 10); display(head); // 1 --->2 --->10
--->3 --->

deleteFromMiddle(head, position);

display(head); // Output:

List elements are - 1 --->2 --->3 --->

List elements are - 4 --->1 --->2 --->3

List elements are - 1 --->2 --->3 --->

List elements are -

1 --->2 --->3 --->

}

--->
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
1 --->2 --->3 --->5 --->

List elements are - 1 --->2 --->3 --->

List elements are -
1 --->2 --->10 --->3 --->

List elements are - 1 --->2 --->3 --->
```

5. Circular Linked List

```
#include <stdio.h> #include <string.h> #include <stdlib.h> #include
<stdbool.h>

struct node { int data;

int key;
struct node *next;

};
struct node *head = NULL;

struct node *current = NULL;

bool isEmpty() { return head == NULL;

}
int length() {

int length = 0;

//if list is empty if(head == NULL) {

return 0; }

current = head->next;
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
while(current != head) { length++;

current = current->n ext; }

return length; }

//insert link at the first location

void insertFirst(int key, int data) {

//create a link
struct node *link = (struct node*) malloc(sizeof(struct node)); link->key =
key;
link->data = data;

if (isEmpty()) { head = link; head->next = head;

} else {
//point it to old first node link->next = head;

//point first to new first node

head = link; }

}

//delete first item

struct node * deleteFirst() { //save reference to first link

struct node *tempLink = head;

if(head->next == head) { head = NULL;
return tempLink;

}
//mark next to first link as first

head = head->next;

//return the deleted link return tempLink;
```

```
}
```

Wipro Elite NLTH Coding / Programming Questions

```
//display the list

void printList() {
struct node *ptr = head;

printf("\n[ ");
//start from the beginning

if(head != NULL) {

while(ptr->next != ptr) { printf("(%d,%d) ",ptr- >key,ptr->data); ptr = ptr->
next;

} }

printf(" ]"); }

void main() { insertFirst(1,10); insertFirst(2,20); insertFirst(3,30);
insertFirst(4,1); insertFirst(5,40); insertFirst(6,56);

printf("Original List: "); //print list

printList();

while(!isEmpty()) {
struct node *temp = deleteFirst(); printf("\nDeleted value:"); printf(
"(%d,%d) ",temp->key,temp->data);

}

printf("\nList after deleting all items: ");

printList(); }
```

Output:

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
Original List:
[ (6,56) (5,40) (4,1) (3,30) (2,20) ] Deleted value:(6,56)
Deleted value:(5,40)
Deleted value:(4,1)
Deleted value:(3,30)
Deleted value:(2,20)
Deleted value:(1,10)
List after deleting all items:
[]
```

6.
```
#include <stdio.h> #include <string.h> #include <stdlib.h> #include
<stdbool.h>

struct node { int data;

int key;

struct node *next;

struct node *prev; };

//this link always point to first Link

struct node *head = NULL;
//this link always point to last Link

struct node *last = NULL; struct node *current = NULL;

//is list empty

bool isEmpty() { return head == NULL;

}

int length() {
int length = 0; struct node *current;

for(current = head; current != NULL; current = current->next) { length++;
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
}
return length;

}

//display the list in from first to last

void displayForward() { //start from the beginning

struct node *ptr = head; //navigate till the end of the list

printf("\n[ ");

while(ptr != NULL) {
printf("(%d,%d) ",ptr->k ey,ptr->data); ptr = ptr->next;

}
printf(" ]");

}

//display the list from last to first

void displayBackward() { //start from the last

struct node *ptr = last;
//navigate till the start of the list

printf("\n[ "); while(ptr != NULL) {

//print data
printf("(%d,%d) ",ptr->k ey,ptr->data);

//move to next item ptr = ptr ->prev;

} }
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
//insert link at the first location

void insertFirst(int key, int data) {

//create a link
struct node *link = (struct node*) malloc(sizeof(struct node)); link->key =
key;
link->data = data;

if(isEmpty()) {
//make it the last link last = link;

} else {
//update first prev link head->prev = link;

}
//point it to old first link

link->next = head;

//point first to new first link

head = link; }

//insert link at the last location

void insertLast(int key, int data) {

//create a link
struct node *link = (struct node*) malloc(sizeof(struct node)); link->key =
key;
link->data = data;

if(isEmpty()) {
//make it the last link last = link;

} else {
//make link a new last link last->next = link;
```

```
//mark old last node as prev of new link

link->prev = last; }

//point last to new last node
```

Wipro Elite NLTH Coding / Programming Questions

```
last = link; }

//delete first item

struct node* deleteFirst() { //save reference to first link

struct node *tempLink = head;

//if only one link if(head->next == NULL){

last = NULL; } else {

head->next->prev = NULL; }

head = head->next; //return the deleted link return tempLink;

}

//delete link at the last location

struct node* deleteLast() { //save reference to last link struct node *
tempLink = last;

//if only one link if(head->next == NULL) {

head = NULL; } else {

last->prev->next = NULL; }

last = last->prev;
```

```
//return the deleted link

return tempLink; }

//delete a link with given key

struct node* delete(int key) {
```

Wipro Elite NLTH Coding / Programming Questions

```
//start from the first link struct node* current = head; struct node*
previous = NULL;

//if list is empty if(head == NULL) { return NULL;

}

//navigate through list while(current->key != key) {

//if it is last node

if(current->next == NULL) { return NULL;

} else {
//store reference to current link previous = current;

//move to next link

current = current->next; }

}

//found a match, update the link if(current == head) {

//change first to point to next link

head = head->next; } else {

//bypass the current link
```

```
current->prev->next = current->next; }

if(current == last) {
//change last to point to prev link last = current->prev;

} else {
current->next->prev = current->prev;

}
return current;

}
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
bool insertAfter(int key, int newKey, int data) { //start from the first link
struct node *current = head;

//if list is empty if(head == NULL) {

return false; }

//navigate through while(current->key

//if it is last if(current->next == NULL) {

return false; } else {

} }

//move to current =

next link

current->next;

list

!= key) {
```

```
node

//create a link
struct node *newLink = (struct node*) malloc(sizeof(struct node)); newLink->
key = newKey;
newLink->data = data;

if(current == last) { newLink->next = NULL; last = newLink;

} else {
newLink->next = current->next; current->next->prev = newLink;

}

newLink->prev = current; current->next = newLink; return true;

}

void main() { insertFirst(1,10); insertFirst(2,20);
```


Wipro Elite NLTH Coding / Programming Questions

```
insertFirst(3,30); insertFirst(4,1); insertFirst(5,40); insertFirst(6,56);

printf("\nList (First to Last): "); displayForward();

printf("\n");
printf("\nList (Last to first): "); displayBackward();

printf("\nList , after deleting first record: "); deleteFirst();
displayForward();

printf("\nList , after deleting last record: "); deleteLast();
displayForward();

printf("\nList , insert after key(4) : "); insertAfter(4,7, 13);
displayForward();

printf("\nList , after delete key(4) : "); delete(4);
displayForward();
```

```
}
```

Output:

```
List (First to Last):
[ (6,56) (5,40) (4,1) (3,30) (2,20) (1,10) ]
List (Last to first):
[ (1,10) (2,20) (3,30) (4,1) (5,40) (6,56) ]
List , after deleting first record:
[ (5,40) (4,1) (3,30) (2,20) (1,10) ]
List , after deleting last record:
[ (5,40) (4,1) (3,30) (2,20) ]
List , insert after key(4) :
[ (5,40) (4,1) (4,13) (3,30) (2,20) ]
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
List , after delete key(4) :
[ (5,40) (4,13) (3,30) (2,20) ]
```

7. Topological Sort Program In C Language

```
#include <stdio.h> int main(){

    int i,j,k,n,a[10][10],indeg[10],flag[10],count=0;

printf("Enter the no of vertices:\n"); scanf("%d",&n);

printf("Enter the adjacency matrix:\n"); for(i=0;i<n;i++){

printf("Enter row %d\n",i+1); for(j=0;j<n;j++)

scanf("%d",&a[i][j]);

}

    for(i=0;i<n;i++){
        indeg[i]=0;

flag[i]=0; }
```

```
    for(i=0;i<n;i++)
        for(j=0;j<n;j++)
            indeg[i]=indeg[i]+a[j][i];

printf("\nThe topological order is:");

    while(count<n){
        for(k=0;k<n;k++){

if((indeg[k]==0) && (flag[k]==0)){ printf("%d ",(k+1));
flag [k]=1;

}
```

Wipro Elite NLTH Coding / Programming Questions

```
      for(i=0;i<n;i++){
          if(a[i][k]==1)

} }

count++; }

return 0; }
```

Output:

Enter the no of vertices:
4
Enter the adjacency matrix: Enter row 1

0 110 Enter row 2 0 001 Enter row 3

```
indeg[k]--;
```

Wipro Elite NLTH Coding / Programming Questions

0 001
Enter row 4
0 000
The topological order is:1 2 3 4

8. String Processing & Manipulation In C Language

```
#include <stdio.h> #include <string.h> int main(void)
{

//variable
char str[100], tmp; int i, len, mid;

//input
printf("Enter a string: "); gets(str);

//find number of characters len = strlen(str);
mid = len/2;

//reverse
for (i = 0; i < mid; i++) {

tmp = str[len - 1 - i]; str[len - 1 - i] = str[i]; str[i] = tmp;
```

Wipro Elite NLTH Coding / Programming Questions

```
}
```

```
//output
printf("Reversed string: %s\n", str);

printf("End of code\n");

return 0; }
```

Output:

Enter a string: Hello World Reversed string: dlroW olleH End of code

9.Stacks & Queues Program In C Language i) Stack:

```
#include <stdio.h>
int MAXSIZE = 8;

int stack[8]; int top = -1; int isempty() { if(top == -1)

return 1;
```

Wipro Elite NLTH Coding / Programming Questions

```
else return 0;

}
int isfull() {

if(top == MAXSIZE) return 1;

else return 0;

}
int peek() {

return stack[top]; }

int pop() { int data;
```

```
if(!isempty()) {
data = stack[top]; top = top - 1;
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
return data; } else

{
printf("Could not retrieve data, Stack is empty.\n");

} }

int push(int data) { if(!isfull()) {

top = top + 1;

stack[top] = data; } else {

printf("Could not insert data, Stack is full.\n"); }

}
int main() {

// push items on to the stack push(3);
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
push(5); push(9); push(1); push(12); push(15);
```

```
printf("Element at top of the stack: %d\n" ,peek()); printf("Elements: \n");
// print stack data

while(!isempty()) { int data = pop(); printf("%d\n",data);

}
printf("Stack full: %s\n" , isfull()?"true":"false"); printf("Stack empty: %s\n" ,
isempty()?"true":"false");

return 0; }
```

Output:

Wipro Elite NLTH Coding / Programming Questions

Element at top of the stack: 15 Elements:
15
12

1
9
5
3
Stack full: false Stack empty: true

ii) Queue

```
#include <stdio.h> #include <string.h> #include <stdlib.h> #include <stdbool.h>

#define MAX 6
```

Wipro Elite NLTH Coding / Programming Questions

```
int intArray[MAX]; int front = 0;
int rear = -1;
int itemCount = 0;

int peek() {
return intArray[front];

}

bool isEmpty() {
return itemCount == 0;

}

bool isFull() {
return itemCount == MAX;

}
```

Wipro Elite NLTH Coding / Programming Questions

```
int size() {
return itemCount;

}

void insert(int data) {

if(!isFull()) {

if(rear == MAX-1) { rear = -1;

}

intArray[++rear] = data;
```

```
itemCount++; }

}
```

Wipro Elite NLTH Coding / Programming Questions

```
int removeData() {
int data = intArray[front++];

if(front == MAX) { front = 0;

}

itemCount--;

return data; }

int main() {
/* insert 5 items */ insert(3); insert(5); insert(9);
```

Wipro Elite NLTH Coding / Programming Questions

```
insert(1); insert(12);

// front : 0
// rear : 4
// ------------------
// index : 0 1 2 3 4
```

// ------------------
// queue : 3 5 9 1 12 insert(15);

// front : 0
// rear : 5
// --------------------- //index:01234 5
// ---------------------
// queue : 3 5 9 1 12 15

Wipro Elite NLTH Coding / Programming Questions

```
if(isFull()) {
printf("Queue is full!\n");

}

// remove one item
int num = removeData();

printf("Element removed: %d\n",num); // front : 1
// rear : 5
// -------------------

//index:1234 5
// -------------------
// queue : 5 9 1 12 15

// insert more items
```

Wipro Elite NLTH Coding / Programming Questions

insert(16);

// front : 1
// rear : -1
// ---------------------- //index:0 1234 5
// ----------------------
// queue : 16 5 9 1 12 15

// As queue is full, elements will not be inserted. insert(17);
insert(18);

// ---------------------- //index:0 1234 5
// ----------------------
// queue : 16 5 9 1 12 15

Wipro Elite NLTH Coding / Programming Questions

printf("Element at front: %d\n",peek());

printf("----------------------\n"); printf("index : 5 4 3 2 1 0\n"); printf("----------------------\n");
printf("Queue: ");

while(!isEmpty()) {
int n = removeData(); printf("%d ",n);

} }

Output:

Queue is full! Element removed: 3 Element at front: 5 ----------------------

Wipro Elite NLTH Coding / Programming Questions

index : 5 4 3 2 1 0

Queue: 5 9 1 12 15 16

10. Sorting & Searching Techniques

i) Sorting

```
/*
* C program to accept N numbers and arrange them in an ascending order */

#include <stdio.h> void main()
{

int i, j, a, n, number[30]; printf("Enter the value of N \n");
```

Wipro Elite NLTH Coding / Programming Questions

```
scanf("%d", &n);

printf("Enter the numbers \n"); for (i = 0; i < n; ++i)

scanf("%d", &number[i]);

for (i = 0; i < n; ++i) {

for (j = i + 1; j < n; ++j) {
```

```
if (number[i] > number[j]) {
```

Wipro Elite NLTH Coding / Programming Questions

```
a = number[i]; number[i] = number[j]; number[j] = a;

}

}

}
printf("The numbers arranged in ascending order are given below \n");

for (i = 0; i < n; ++i) printf("%d\n", number[i]);

}
```

Output:

Enter the value of N: 6

Wipro Elite NLTH Coding / Programming Questions

Enter the numbers 3
78
90

456
780
200
The numbers arranged in ascending order are given below 3

78
90
200
456
780

Wipro Elite NLTH Coding / Programming Questions

ii) Searching

1. 2. 3. 4. 5. 6. 7. 8. 9. 10. 11. 12. 13. 14. 15. 16. 17. 18. 19. 20. 21. 22. 23. 24.

```
#include <stdio.h>

int main() {

int array[100], search, c, n;
printf("Enter number of elements in array\n");

scanf("%d", &n);

printf("Enter %d integer(s)\n" , n);

for (c = 0; c < n; c+ +) scanf("%d", &array[c ]);

printf("Enteranumbertosearch\n"); scanf("%d", &search) ;

for (c = 0; c < n; c+ +) {

if (array[c] == search)  {

/* If required element is found */
```

```
printf("%d is present at location %d.\n", search, c+1);

break; }
```

FRESHERS NOW

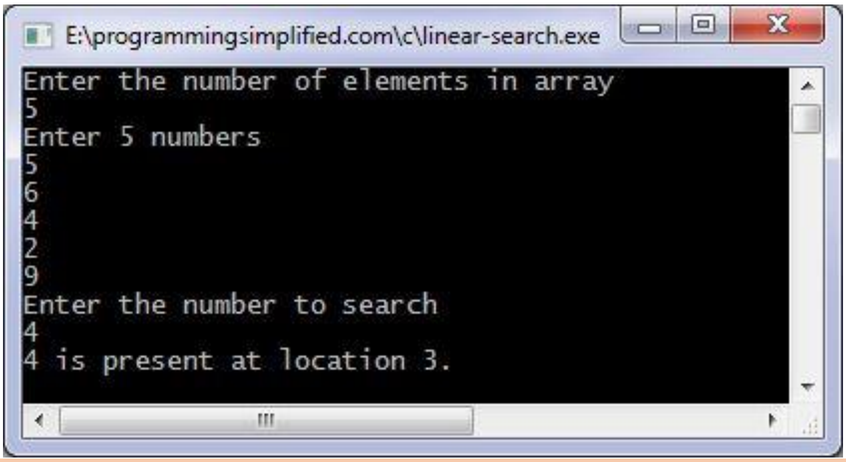
Wipro Elite NLTH Coding / Programming Questions

```
25.}
26.if (c == n)
27.printf("%d isn't present in the array.\n", search);

28.
29. return 0; 30.}
```

Output:

11. Dynamic Programming

```
#include<stdio.h>

int max(int a, int b) { return (a > b)? a : b; }
```



Wipro Elite NLTH Coding / Programming Questions

```
int knapSack(int W, int wt[], int val[], int n) {

int i, w;
int K[n+1][W+1];

for (i = 0; i <= n; i++) {

for (w = 0; w <= W; w++) {

if (i==0 || w==0) K[i][w] = 0;

else if (wt[i-1] <= w)
K[i][w] = max(val[i-1] + K[i-1][w-wt[i-1]], K[i-1][w]);

else
K[i][w] = K[i-1][w];
```

Wipro Elite NLTH Coding / Programming Questions

```
} }

return K[n][W]; }

int main()
{
int i, n, val[20], wt[20], W;

printf("Enter number of items:"); scanf("%d", &n);

printf("Enter value and weight of items:\n"); for(i = 0;i < n; ++i){

scanf("%d%d", &val[i], &wt[i]); }
```

printf("Enter size of knapsack:"); scanf("%d", &W);

Wipro Elite NLTH Coding / Programming Questions

printf("%d", knapSack(W, wt, val, n));

return 0; }

Output:

Enter number of items:3
Enter value and weight of items:
100 20
50 10
150 30
Enter size of knapsack:50
250

12. Greedy Algorithm In C Language #include <stdio.h>
int main () {

int num_denominations, coin_list[100], use_these[100], i, owed;

Wipro Elite NLTH Coding / Programming Questions

printf("Enter number of denominations : "); scanf("%d", &num_denominations);

printf("\nEnter the denominations in descending order: ");

for(i=0; i< num_denominations; i++) { scanf("%d", &coin_list[i]);

// use_these[i] = 0; }

printf("\nEnter the amount owed : "); scanf("%d", &owed);
for(i=0; i < num_denominations; i++) {

use_these[i] = owed / coin_list[i];

Wipro Elite NLTH Coding / Programming Questions

owed %= coin_list[i]; }

printf("\nSolution: \n");
for(i=0; i < num_denominations; i++) {

printf("%dx%d ", coin_list[i], use_these[i]); }

}

Output:

```
~/cpe/greedy_coin_change
sandeepa@sn ~/cpe/greedy_coin_change
$ ./a.exe
Enter number of denominations : 3

Enter the denominations in descending order: 10 5 1

Enter the amount owed : 17

Solution:
10x1 5x1 1x2
sandeepa@sn ~/cpe/greedy_coin_change
$ ./a.exe
Enter number of denominations : 5

Enter the denominations in descending order: 100
50 20 5 1

Enter the amount owed : 78

Solution:
100x0 50x1 20x1 5x1 1x3
sandeepa@sn ~/cpe/greedy_coin_change
$ |
```

Wipro Elite NLTH Coding / Programming Questions

13. String Matching Program In C Language

```
#include<stdio.h> #include<conio.h>

int length(char x[]) {

int i; for(i=0;x[i]!='\0';i++) {}
return i;

}
void main() {

char s[20],p[20];
```

Wipro Elite NLTH Coding / Programming Questions

```
int i,l,count=0; clrscr();

printf("\n enter Your String = "); scanf("%s",s);
printf("enter the string to be matched = "); scanf("%s",p );

l=length(p);

for(i=0;s[i]!='\0';i++) {

if(s[i]==p[count] ) count++;
```

else {

Wipro Elite NLTH Coding / Programming Questions

count=0; }

if (count == l) {

printf("Substring %s found in the given string",p); break;

} }if(count!=l)

printf("not found"); getch();

}

Output:

```
 enter Your String   =  110101010100011
enter the string to be matched = 1010
Substring 1010 found in the given string
 enter Your String   =  11001010101010101011010101010101
enter the string to be matched = 101
Substring 101 found in the given string
```

Wipro Elite NLTH Coding / Programming Questions

14. Divide & Conquer Program In C language

```
#include <stdio.h>

#define max 10

int a[11] = { 10, 14, 19, 26, 27, 31, 33, 35, 42, 44, 0 }; int b[10];

void merging(int low, int mid, int high) { int l1, l2, i;

for(l1 = low, l2 = mid + 1, i = low; l1 <= mid && l2 <= high; i++) { if(a[l1] <= a[l2])

b[i] = a[l1++]; else
```


Wipro Elite NLTH Coding / Programming Questions

```
b[i] = a[l2++]; }

while(l1 <= mid) b[i++] = a[l1++];

while(l2 <= high) b[i++] = a[l2++];

for(i = low; i <= high; i++) a[i] = b[i];

}

void sort(int low, int high) { int mid;
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
if(low < high) {
mid = (low + high) / 2; sort(low, mid); sort(mid+1, high); merging(low, mid, high);

} else { return;

} }

int main() { int i;

printf("List before sorting\n");
```

Wipro Elite NLTH Coding / Programming Questions

```
for(i = 0; i <= max; i++) printf("%d ", a[i]);

sort(0, max);

printf("\nList after sorting\n");

for(i = 0; i <= max; i++) printf("%d ", a[i]);

}
```

Output:

List before sorting
10 14 19 26 27 31 33 35 42 44 0 List after sorting
0 10 14 19 26 27 31 33 35 42 44

Wipro Elite NLTH Coding / Programming Questions

15. Disjoint sets Program In C Language

```
// A union-find algorithm to detect cycle in a graph #include <stdio.h>
#include <stdlib.h>
#include <string.h>

// a structure to represent an edge in graph struct Edge
{

int src, dest; };

// a structure to represent a graph struct Graph
{
```

Wipro Elite NLTH Coding / Programming Questions

```
// V-> Number of vertices, E-> Number of edges int V, E;

// graph is represented as an array of edges

struct Edge* edge; };

// Creates a graph with V vertices and E edges struct Graph* createGraph(int V, int E)
{

struct Graph* graph =
(struct Graph*) malloc( sizeof(struct Graph) );

graph->V = V; graph->E = E;
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
graph->edge =
(struct Edge*) malloc( graph->E * sizeof( struct Edge ) );

return graph; }

// A utility function to find the subset of an element i int find(int parent[], int i)
{

if (parent[i] == -1) return i;

return find(parent, parent[i]); }

// A utility function to do union of two subsets
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
void Union(int parent[], int x, int y) {

int xset = find(parent, x); int yset = find(parent, y); if(xset!=yset){

parent[xset] = yset; }

}
```

```
// The main function to check whether a given graph contains // cycle or not
int isCycle( struct Graph* graph )
{

// Allocate memory for creating V subsets
int *parent = (int*) malloc( graph->V * sizeof(int) );
```


Wipro Elite NLTH Coding / Programming Questions

```
// Initialize all subsets as single element sets memset(parent, -1, sizeof(int) * graph->V);

// Iterate through all edges of graph, find subset of both // vertices of every edge, if both
subsets are same, then // there is cycle in graph.
for(int i = 0; i < graph->E; ++i)

{
int x = find(parent, graph->edge[i].src); int y = find(parent, graph->edge[i].dest);

if (x == y) return 1;
```


Wipro Elite NLTH Coding / Programming Questions

```
Union(parent, x, y); }

return 0; }

// Driver program to test above functions int main()
{
```

```
/* Let us create following graph 0

|\
|\ 1-----2 */

int V = 3, E = 3;
struct Graph* graph = createGraph(V, E);
```


Wipro Elite NLTH Coding / Programming Questions

```
// add edge 0-1 graph->edge[0].src = 0; graph->edge[0].dest = 1;

// add edge 1-2 graph->edge[1].src = 1; graph->edge[1].dest = 2;

// add edge 0-2 graph->edge[2].src = 0; graph->edge[2].dest = 2;

if (isCycle(graph))
printf( "graph contains cycle" );
```


Wipro Elite NLTH Coding / Programming Questions

```
else
printf( "graph doesn't contain cycle" );

return 0; }
```

Output:

graph contains cycle

16. Computational Geometry

```
#include <bits/stdc++.h> using namespace std;

struct Point {

int x, y; };
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
// To find orientation of ordered triplet (p, q, r). // The function returns following values
// 0 --> p, q and r are colinear
// 1 --> Clockwise

// 2 --> Counterclockwise
int orientation(Point p, Point q, Point r) {

int val = (q.y - p.y) * (r.x - q.x) - (q.x - p.x) * (r.y - q.y);

if (val == 0) return 0; // colinear

return (val > 0)? 1: 2; // clock or counterclock wise }
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
// Prints convex hull of a set of n points. void convexHull(Point points[], int n)
{

// There must be at least 3 points if (n < 3) return;

// Initialize Result vector<Point> hull;

// Find the leftmost point int l = 0;
for (int i = 1; i < n; i++)

if (points[i].x < points[l].x) l = i;
```

Wipro Elite NLTH Coding / Programming Questions

```
// Start from leftmost point, keep moving counterclockwise // until reach the start point again.
This loop runs O(h)
// times where h is number of points in result or output.
int p = l, q;

do {

// Add current point to result hull.push_back(points[p]);

// Search for a point 'q' such that orientation(p, x,
// q) is counterclockwise for all points 'x'. The idea
// is to keep track of last visited most counterclock-
// wise point in q. If any point 'i' is more counterclock- // wise than q, then update q.
q = (p+1)%n;
```

Wipro Elite NLTH Coding / Programming Questions

```
for (int i = 0; i < n; i++) {

// If i is more counterclockwise than current q, then // update q
if (orientation(points[p], points[i], points[q]) == 2)

q = i; }
```

- // Now q is the most counterclockwise with respect to p
- // Set p as q for next iteration, so that q is added to
- // result 'hull'

```
p = q;

} while (p != l); // While we don't come to first point
```

Wipro Elite NLTH Coding / Programming Questions

```
// Print Result
for (int i = 0; i < hull.size(); i++)

cout << "(" << hull[i].x << ", " << hull[i].y << ")\n";

}
// Driver program to test above functions

int main() {

Point points[] = {{0, 3}, {2, 2}, {1, 1}, {2, 1}, {3, 0}, {0, 0}, {3, 3}};

int n = sizeof(points)/sizeof(points[0]); convexHull(points, n);
return 0;

}
```

Output:

Wipro Elite NLTH Coding / Programming Questions

The output is points of the convex hull. (0, 3)
(0, 0)
(3, 0)

(3, 3)
17. // Program to print BFS traversal from a given // source vertex. BFS(int s) traverses vertices
// reachable from s.

```
#include<iostream>
#include <list>

using namespace std;

// This class represents a directed graph using // adjacency list representation
```


Wipro Elite NLTH Coding / Programming Questions


```
class Graph {

int V; // No. of vertices

// Pointer to an array containing adjacency // lists
list<int> *adj;

public:
Graph(int V); // Constructor
```

```
// function to add an edge to graph void addEdge(int v, int w);

// prints BFS traversal from a given source s void BFS(int s);

};
```

Wipro Elite NLTH Coding / Programming Questions

```
Graph::Graph(int V) {

this->V = V;

adj = new list<int>[V]; }

void Graph::addEdge(int v, int w) {

adj[v].push_back(w); // Add w to v’s list. }

void Graph::BFS(int s) {
```

Wipro Elite NLTH Coding / Programming Questions

```
// Mark all the vertices as not visited bool *visited = new bool[V];
for(int i = 0; i < V; i++)

visited[i] = false;

// Create a queue for BFS list<int> queue;
```

```
// Mark the current node as visited and enqueue it visited[s] = true;
queue.push_back(s);

// 'i' will be used to get all adjacent // vertices of a vertex list<int>::iterator i;
```

Wipro Elite NLTH Coding / Programming Questions

```
while(!queue.empty()) {

// Dequeue a vertex from queue and print it s = queue.front();
cout << s << " ";
queue.pop_front();

// Get all adjacent vertices of the dequeued // vertex s. If a adjacent has not been visited, //
then mark it visited and enqueue it
for (i = adj[s].begin(); i != adj[s].end(); ++i)
{

if (!visited[*i]) {
```

Wipro Elite NLTH Coding / Programming Questions

```
visited[*i] = true;

queue.push_back(*i); }

} }
```

```
}

// Driver program to test methods of graph class int main()
{

// Create a graph given in the above diagram Graph g(4);
g.addEdge(0, 1);
g.addEdge(0, 2);

g.addEdge(1, 2);
```


Wipro Elite NLTH Coding / Programming Questions

```
g.addEdge(2, 0); g.addEdge(2, 3); g.addEdge(3, 3);

cout << "Following is Breadth First Traversal " << "(starting from vertex 2) \n";

g.BFS(2);

return 0; }
```

Output

Following is Breadth First Traversal (starting from vertex 2) 2031

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

18. #include <stdio.h>

```
#include <stdlib.h>

struct node { int data;

struct node* left;

struct node* right; };

struct node* createNode(value){
struct node* newNode = malloc(sizeof(struct node)); newNode->data = value;
newNode->left = NULL;
newNode->right = NULL;
```

Wipro Elite NLTH Coding / Programming Questions

```
return newNode; }

struct node* insertLeft(struct node *root, int value) { root->left = createNode(value);
return root->left;

}

struct node* insertRight(struct node *root, int value){ root->right = createNode(value);
return root->right;

}

int main(){
```

Wipro Elite NLTH Coding / Programming Questions

```
struct node *root = createNode(1); insertLeft(root, 2); insertRight(root, 3);

printf("The elements of tree are %d %d %d", root->data, root->left->data, root->right->data);
}
```

Output - 1 2 3

19. Dijkstra's Algorithm

```
#include<stdio.h> #include<conio.h> #define INFINITY 9999 #define MAX 10

void dijkstra(int G[MAX][MAX],int n,int startnode);
```

Wipro Elite NLTH Coding / Programming Questions

```
int main() {

int G[MAX][MAX],i,j,n,u;
printf("Enter no. of vertices:"); scanf("%d",&n);
printf("\nEnter the adjacency matrix:\n");

for(i=0;i<n;i++) for(j=0;j<n;j++)

scanf("%d",&G[i][j]);

printf("\nEnter the starting node:"); scanf("%d",&u);
dijkstra(G,n,u);
```

Wipro Elite NLTH Coding / Programming Questions

return 0; }

void dijkstra(int G[MAX][MAX],int n,int startnode) {

int cost[MAX][MAX],distance[MAX],pred[MAX]; int
visited[MAX],count,mindistance,nextnode,i,j;

//pred[] stores the predecessor of each node //count gives the number of nodes seen so far
//create the cost matrix
for(i=0;i<n;i++)

for(j=0;j<n;j++) if(G[i][j]==0)

Wipro Elite NLTH Coding / Programming Questions

cost[i][j]=INFINITY; else

cost[i][j]=G[i][j];

//initialize pred[],distance[] and visited[] for(i=0;i<n;i++)
{

distance[i]=cost[startnode][i]; pred[i]=startnode; visited[i]=0;

}

```
distance[startnode]=0; visited[startnode]=1; count=1;
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
while(count<n-1) {

mindistance=INFINITY;

//nextnode gives the node at minimum distance for(i=0;i<n;i++)

if(distance[i]<mindistance&&!visited[i]) {

mindistance=distance[i];

nextnode=i; }

//check if a better path exists through nextnode visited[nextnode]=1;
```

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

```
for(i=0;i<n;i++) if(!visited[i])

if(mindistance+cost[nextnode][i]<distance[i]) {

distance[i]=mindistance+cost[nextnode][i];

pred[i]=nextnode; }
```

```
count++; }

//print the path and distance of each node for(i=0;i<n;i++)

if(i!=startnode) {

printf("\nDistance of node%d=%d",i,distance[i]);
```

Wipro Elite NLTH Coding / Programming Questions

```
printf("\nPath=%d",i);

j=i; do {

j=pred[j];

printf("<-%d",j); }while(j!=startnode);

} }
```

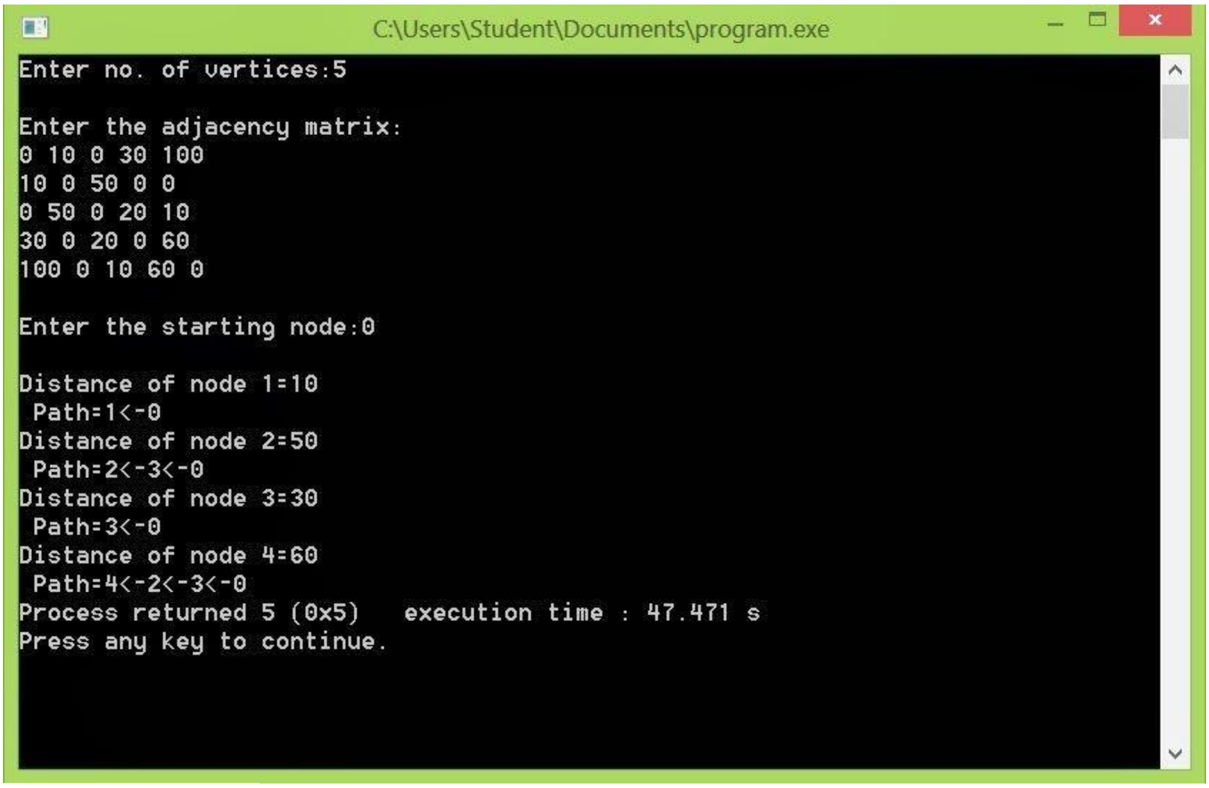
Wipro Elite NLTH Coding / Programming Questions

Output:

20. Prims Algorithm

```
// A C / C++ program for Prim's Minimum
// Spanning Tree (MST) algorithm. The program is // for adjacency matrix representation of
```

```
the graph #include <stdio.h>
#include <limits.h>
```



```
C:\Users\Student\Documents\program.exe
Enter no. of vertices:5

Enter the adjacency matrix:
0 10 0 30 100
10 0 50 0 0
0 50 0 20 10
30 0 20 0 60
100 0 10 60 0

Enter the starting node:0

Distance of node 1=10
 Path=1<-0
Distance of node 2=50
 Path=2<-3<-0
Distance of node 3=30
 Path=3<-0
Distance of node 4=60
 Path=4<-2<-3<-0
Process returned 5 (0x5)   execution time : 47.471 s
Press any key to continue.
```


Wipro Elite NLTH Coding / Programming Questions

```
#include<stdbool.h>
// Number of vertices in the graph #define V 5

// A utility function to find the vertex with
// minimum key value, from the set of vertices // not yet included in MST
int minKey(int key[], bool mstSet[])
{
// Initialize min value
int min = INT_MAX, min_index;
```

```
for (int v = 0; v < V; v++)
if (mstSet[v] == false && key[v] < min)

min = key[v], min_index = v;
```

Wipro Elite NLTH Coding / Programming Questions

```
return min_index; }

// A utility function to print the
// constructed MST stored in parent[]
int printMST(int parent[], int n, int graph[V][V]) {
printf("Edge \tWeight\n");
for (int i = 1; i < V; i++)

printf("%d - %d \t%d \n", parent[i], i, graph[i][parent[i]]); }

// Function to construct and print MST for // a graph represented using adjacency
```

Wipro Elite NLTH Coding / Programming Questions

```
// matrix representation
void primMST(int graph[V][V]) {

// Array to store constructed MST
int parent[V];
// Key values used to pick minimum weight edge in cut int key[V];
// To represent set of vertices not yet included in MST bool mstSet[V];
```

// Initialize all keys as INFINITE for (int i = 0; i < V; i++)

key[i] = INT_MAX, mstSet[i] = false;

// Always include first 1st vertex in MST.

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

// Make key 0 so that this vertex is picked as first vertex. key[0] = 0;
parent[0] = -1; // First node is always root of MST

// The MST will have V vertices
for (int count = 0; count < V-1; count++) {

// Pick the minimum key vertex from the // set of vertices not yet included in MST int u = minKey(key, mstSet);

// Add the picked vertex to the MST Set mstSet[u] = true;

// Update key value and parent index of

FRESHERS NOW

Wipro Elite NLTH Coding / Programming Questions

// the adjacent vertices of the picked vertex. // Consider only those vertices which are not // yet included in MST
for (int v = 0; v < V; v++)

```
// graph[u][v] is non zero only for adjacent vertices of m
// mstSet[v] is false for vertices not yet included in MST
// Update the key only if graph[u][v] is smaller than key[v]
if (graph[u][v] && mstSet[v] == false && graph[u][v] < key[v])

parent[v] = u, key[v] = graph[u][v]; }

// print the constructed MST

printMST(parent, V, graph); }
```


Wipro Elite NLTH Coding / Programming Questions

```
// driver program to test above function int main()
{
/* Let us create the following graph

23 (0)--(1)--(2) |/\|
6| 8/ \5 |7 |/ \| (3)-------(4)

9 */
int graph[V][V] = {{0, 2, 0, 6, 0},

{2, 0, 3, 8, 5},
```

Wipro Elite NLTH Coding / Programming Questions

```
{0, 3, 0, 0, 7}, {6, 8, 0, 0, 9}, {0, 5, 7, 9, 0}};

// Print the solution primMST(graph);

return 0; }
```

Output:
Edge Weight 0-1 2

1-2 3
0-3 6
1-4 5