www.MyPlacementPrep.com

Pro Material Series

Wipro Elite NLTH Coding Placement Questions

1) Print the below pattern (half diamond using numbers) Input:

34 Output:

```
   3
   44
   555
   6666
   555
   44
   3
```

Input :
44

Output:
4

```
   55
   666
   7777
   666
   55
```

4

Program: #include int main() {

int i,j,s,N,count=0; scanf(“%d%d”,&s,&N); for(i=s;count<4;count++) {

for(j=0;j<count+1;j++) printf(“%d”,i);

printf(“\n”);

i=i+1; }

for(i=s+N-2;count>0;count–)

Wipro Elite NLTH Coding Placement Questions

```
{
for(j=0;j<count-1;j++)

printf("%d",i); printf("\n"); i=i-1;

}

return 0; }
```

2) Print the following pattern (half diamond using numbers) Input :
3
Output:

```
1
2*2 3*3*3 3*3*3 2*2
1
```

Input : 4 Output: 1

```
2*2 3*3*3 4*4*4*4 4*4*4*4 3*3*3 2*2
1
```

Program: #include int main() {

```
int i,j,k,N,count=0;
```

Wipro Elite NLTH Coding Placement Questions

```
scanf("%d",&N); for(i=1;i<=N;i++) {

k=1; for(j=0;j<i;j++) {

printf("%d",i); if(k<i)
{

printf("*");

k=k+1; }

}

printf("\n"); }

for(i=N;i>0;i–) {
```

k=1; for(j=0;j<i;j++) {

printf(“%d”,i); if(k<i)
{

printf(“*”);

k=k+1; }

}

printf(“\n”); }

return 0; }

3) Print the below pattern.

Input: 4 Output:

1
2*3 4*5*6 7*8*9*10 7*8*9*10 4*5*6 2*3
1

Program: #include int main() {

int i,j,count=1,n; printf(“Enter a number\n”); scanf(“%d”,&n);

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

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

} }

if(j<i) printf(“%d*”,count++);

else printf(“%d”,count++);

printf(“\n”);

count=count-n; for(i=n;i>=1;i–)

{

for(j=1;j<=i;j++)

{
if(j<i)

printf(“%d*”,count++); else

printf(“%d”,count++); }

count=(count+1)-2*i;

printf(“\n”); }

Wipro Elite NLTH Coding Placement Questions

Wipro Elite NLTH Coding Placement Questions

return 0; }

4) Print the following pattern.

Input: 34 Output: 3

44
555
6666
6666
555
44

3

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

int i,j,s,N,count=0; scanf(“%d%d”,&s,&N); for(i=s;count<4;count++) {

for(j=0;j<count+1;j++) printf(“%d”,i);

printf(“\n”);

i=i+1; }

```
for(i=s+N-2;count>0;count–) {

for(j=0;j<count-1;j++) printf(“%d”,i);

printf(“\n”);

i=i-1; }
```



```
return 0; }
```

5) Print the below pattern.

Input: 5 Output: 1

```
3*2
4*5*6
10*9*8*7 11*12*13*14*15

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

int i,j,k,l=1,N,d,r,count=0; scanf(“%d”,&N); for(i=1;i<=N;i++)
{

k=1;
d=i%2; r=l+i-1; for(j=0;j<i;j++) {

if(d==0) {

k=k+1; }

printf(“%d”,r); r–;
if(k<i)
{

printf(“*”);
```

Wipro Elite NLTH Coding Placement Questions

l++;

continue; }

printf(“%d”,l); l++;
if(k<i)
{

printf(“*”);

k=k+1; }

}

printf(“\n”); }

return 0; }

6) Print the below pattern.

Input:
4
Output: 1*2*3*4*17*18*19*20 – -5*6*7*14*15*16

– – – -8*9*12*13 – – – – – -10*11

Program: #include<stdio.h> void pattern(int); int main()
{

int n; scanf(“%d”, &n); pattern(n); return 0;

}
void pattern(int n)

{

} }

Wipro Elite NLTH Coding Placement Questions

int i, j, k, s, a = 1,b = n*n + 1; for (i = n; i >= 1; i–) {

for (s = 0; s < n – i; s++) printf(“–“);

```
for (j = 0; j < i; j++) printf("%d*", a++);

for (k = 0; k < i – 1; k++) printf("%d*", b++);

printf("%d\n", b); b -= 2*(i – 1);

// last b should without *
```

7) Prims Algorithm

```
// A C / C++ program for Prim's Minimum
// Spanning Tree (MST) algorithm. The program is // for adjacency matrix representation of
thegraph #include <stdio.h>
#include <limits.h>
#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;
return min_index;
```

Wipro Elite NLTH Coding Placement Questions

```
}

// 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 // 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];
// https://www.freshersnow.com/placement-papers-download/

// 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.
// 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
```

Wipro Elite NLTH Coding Placement Questions

```
// 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
// 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);
}
// 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},
{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 0-12 1-23 0-36 1-45

8) Print the below pattern.

Input: 3 Output: 333 313 323 333

Program: #include<stdio.h> int main()

```
{

int i, j, n, c=1; scanf("%d", &n); for(i=1; i<=n+1; i++) {

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

Wipro Elite NLTH Coding Placement Questions

```
if(i!=1 && j==n-1) {
printf("%d ", c);

c++; }

else
printf("%d ", n);

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

```
}
return 0;
```

9) 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;
}
```

Enter n: 5
Enter number1: 45 Enter number2: 35 Enter number3: 38 Enter number4: 31 Enter number5: 49

```
}
```

Wipro Elite NLTH Coding Placement Questions

Average = 39

10) Operations On Linked List

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

Wipro Elite NLTH Coding Placement Questions

```
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;
}
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->next;
```

Wipro Elite NLTH Coding Placement Questions

```
}
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;
/* 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); // 1 --->2 --->3 --->
}
Output:
```

Wipro Elite NLTH Coding Placement Questions

```
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 --->5 ---> List elements are -
1 --->2 --->3 --->
List elements are -
1 --->2 --->10 --->3 ---> List elements are -
1 --->2 --->3 --->
```