Aim:

The `addNodes()` function creates a new list and adds elements to the list until delimiter `-1` is occurred.

Fill in the missing code in the below functions `addNodes(NODE first, int x)` and `traverseList(NODE first)` in the file `CreateAndAddNodes.c`.

Source Code:

SingleLL1.c

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

#include "CreateAndAddNodes.c"

void main() {
    NODE first = NULL;
    int x;
    printf("Enter elements up to -1 : ");
    scanf("%d", &x);
    while (x != -1) {
        first = addNodes(first, x);
        scanf("%d", &x);
    }
    if (first == NULL) {
        printf("Single Linked List is empty\n");
    } else {
        printf("The elements in SLL are : ");
        traverseList(first);
    }
}
```

CreateAndAddNodes.c

```c
struct node {
    int data;
    struct node *next;
};
typedef struct node *NODE;
NODE createNode() {
NODE temp;
temp=(NODE)malloc(sizeof(struct node));
temp->next=NULL;
return temp;
}
NODE first=NULL;
NODE addNodes(NODE first, int x) {
NODE temp;
temp=createNode();
temp->data=x;
if(first==NULL)
{
```

```
    first=temp;
    }
    else
    {
        NODE lastNode = first;
        while(lastNode->next!=NULL)
        {
            lastNode=lastNode->next;
        }
        lastNode->next=temp;
        }
        return first;
    }
    void traverseList(NODE first) {
    if(first==NULL)
    {
        printf("List is empty");
    }
    else
    {
        NODE temp=first;
        while(temp!=NULL)
    {
        printf("%d --> ",temp->data);
        temp=temp->next;
    }
    printf("NULL\n");
    }
    }
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter elements up to -1 : 9 18 27 36 45 -1
The elements in SLL are : 9 --> 18 --> 27 --> 36 --> 45 --> NULL

Test Case - 2
User Output
Enter elements up to -1 : 12 14 19 23 -1
The elements in SLL are : 12 --> 14 --> 19 --> 23 --> NULL