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

struct TNode
{
  char data;
  struct TNode* left;
  struct TNode* right;
};

struct TNode* newNode(char data);

struct TNode* arrayToTree(char arr[], int start, int end)
{
  if (start > end)
  return NULL;
```

```
13 | struct TNode* arrayToTree(char arr[], int start, int end)
14 | {
15 |  if (start > end)
16 |  return NULL;
17 |
18 |  int mid = (start + end)/2;
19 |  struct TNode *root = newNode(arr[mid]);
20 |
21 |  root->left = arrayToTree(arr, start, mid-1);
22 |
23 |  root->right = arrayToTree(arr, mid+1, end);
24 |
25 |  return root;
26 | }
27 |
```

```
33 |  node->right = NULL;
34 |
35 |  return node;
36 | }
37 |
38 | void preOrder(struct TNode* node)
39 | {
40 |  if (node == NULL)
41 |  return;
42 |  printf("%c", node->data);
43 |  preOrder(node->left);
44 |  preOrder(node->right);
45 | }
46 |
47 | void reverseArray(char *arr, int start, int end) {
48 |  while(start<end) {
49 |  char x = arr[start];
50 |  arr[start++] = arr[end];
51 |  arr[end--] = x;
52 |  }
```

```
63 | int main()
64 | {
65 |   char arr[] = "kbveqf*ocoan";
66 |
67 |   int n;
68 |   for(n=0; arr[n]!='\0'; ++n);
69 |
70 |   struct TNode *root = arrayToTree(arr, 0, n-1
71 |   preOrder(root);
72 |   printf("-");
```

Question: 4

Given a sorted array containing N integers, both positive and negative.
Create another array containing the squares of all the elements in the input array and return it in

Example:

Input: [-7, -5, -2, 0, 1]

Output: [0, 1, 4, 25, 49]

Expected Time complexity: O(n)

Question: 2

Provided a sorted linked list, write a program to delete all the nodes that have duplicate numbers, leaving only

Example,
Given 2->4->5->5->6->6->8, return 2->4->8.
Given 2->2->2->4->5, return 4->5.

Expected Time complexity: O(n)
Expected Space complexity: O(1)

Question: 3

Provided a binary tree containing digits only from 0-9, each root-to-leaf path represents a number. Return the total sum of a

Example :

Input:

```
  2
 / \
3   5
```

The root-to-leaf paths are 2->3 (representing the number 23) and 2->5 (representing the number 25).

Value returned: 48(23 + 25).

Expected Time complexity: O(n)

wer:

Question: 1

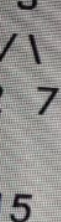
Given a string S, find the second non-repeating character of S. String contains lower case characters only [a
If no such character exists then return "-".

For example :
Input : abbcdefgabbcdabcg
Output: f
Note: The solution should be implemented with O(n) time complexity and constant space complexity

wer:

A tree is considered special if the sums of all the nodes at each level are in an Arithmetic Progression (AP).
Given the root note of a binary tree. Return an array representing the minimum number that can be added at each

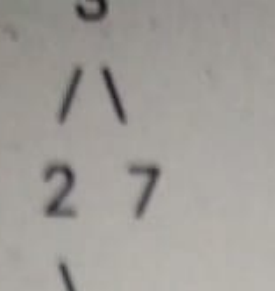
Input:



Output: [0, 0, 0]

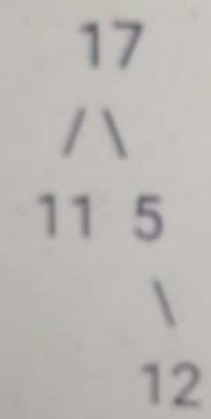
nput:

17



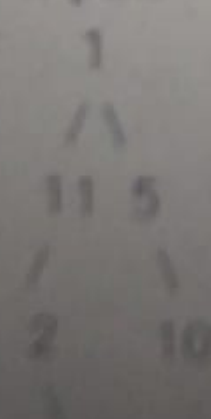
Output: [0, 0, 0]

Input:



Output: [0, 0, 3]

Input:



Question: 3

Given a linked list, arrange the linked list in the manner of alternate last and first. Make sure to
Note: The solution should be implemented with O(n) time complexity and O(1) space complex
For example:
Input: 1->2->3->4->5->6->7->8
Output: 8->1->7->2->6->3->5->4

Answer:

anguage: C

Bob handles the job of packaging at a chocolate shop.
Bob has N chocolates, and needs to decide how many chocolates to place in each package.
Each package must contain the same number of chocolates.
Bob will choose an integer A between 1 and N, inclusive, and place exactly A chocolates into each package.
Bob makes as many packages as possible and then gets to eat the remaining chocolates. Bob enjoys eating chocolate
eat as many chocolates as possible.

Input: N (The number of chocolates came for packaging)
Output: A (The package size that will maximize the number of leftover chocolates. If multiple package sizes will resu
largest such size.)

Examples:

Input: 2
Output: 2
Explanation: There will be no leftover chocolates, regardless of the size Bob chooses.

Input: 5