Code :

10

/ \

28 13

/ \

14 15

/ \ / \

21 22 23 24

For example we taken above scenario based on this we perform coding

#include<bits/stdc++.h>

using namespace std;

struct Node

{

int key;

struct Node \*left, \*right;

};

Node\* n\_node(int key)

{

Node\* temp = new Node;

temp->key = key;

temp->left = temp->right = NULL;

return (temp);

}

void printPathsUtil(Node\* curr\_node, int sum,

int sum\_so\_far, vector<int> &path)

{

if (curr\_node == NULL)

return;

sum\_so\_far += curr\_node->key;

path.push\_back(curr\_node->key);

if (sum\_so\_far == sum )

{

cout << "Path found: ";

for (int i=0; i<path.size(); i++)

cout << path[i] << " ";

cout << endl;

}

if (curr\_node->left != NULL)

printPathsUtil(curr\_node->left, sum, sum\_so\_far, path);

if (curr\_node->right != NULL)

printPathsUtil(curr\_node->right, sum, sum\_so\_far, path);

path.pop\_back();

}

void printPaths(Node \*root, int sum)

{

vector<int> path;

printPathsUtil(root, sum, 0, path);

}

int main ()

{

/\* 10

/ \

28 13

/ \

14 15

/ \ / \

21 22 23 24\*/

Node \*root = n\_node(10);

root->left = n\_node(28);

root->right = n\_node(13);

root->right->left = n\_node(14);

root->right->right = n\_node(15);

root->right->left->left = n\_node(21);

root->right->left->right = n\_node(22);

root->right->right->left = n\_node(23);

root->right->right->right = n\_node(24);

int sum = 38;

printPaths(root, sum);

return 0;

}

Screenshot:

