

Print all k-sum paths in a binary tree

A binary tree and a number k are given. Print every path in the tree with sum of the nodes in the path as k.

A path can start from any node and end at any node and must be downward only, i.e. they need not be root node and leaf node; and negative numbers can also be there in the tree.

Input : k = 5

Root of below binary tree:

```
1
/
3 -1
/\
2 1 4 5
//\
1 1 2 6
```

Output :

```
3 2
3 1 1
1 3 1
4 1
1 -1 4 1
-1 4 2
5
1 -1 5
```

```
def printVector(v, i):
    for j in range(i, len(v)):
        print(v[j], end = " ")
    print()

# Binary Tree Node
""" utility that allocates a newNode
with the given key """
class newNode:

    # Construct to create a newNode
    def __init__(self, key):
        self.data = key
        self.left = None
```

```

        self.right = None

# This function prints all paths
# that have sum k
def printKPathUtil(root, path, k):

    # empty node
    if (not root) :
        return

    # add current node to the path
    path.append(root.data)

    # check if there's any k sum path
    # in the left sub-tree.
    printKPathUtil(root.left, path, k)

    # check if there's any k sum path
    # in the right sub-tree.
    printKPathUtil(root.right, path, k)

    # check if there's any k sum path that
    # terminates at this node
    # Traverse the entire path as
    # there can be negative elements too
    f = 0
    for j in range(len(path) - 1, -1, -1):
        f += path[j]

        # If path sum is k, print the path
        if (f == k) :
            printVector(path, j)

    # Remove the current element
    # from the path
    path.pop(-1)

# A wrapper over printKPathUtil()
def printKPath(root, k):

    path = []
    printKPathUtil(root, path, k)

```

```
# Driver Code
```

```
if __name__ == '__main__':
```

```
    root = newNode(1)
    root.left = newNode(3)
    root.left.left = newNode(2)
    root.left.right = newNode(1)
    root.left.right.left = newNode(1)
    root.right = newNode(-1)
    root.right.left = newNode(4)
    root.right.left.left = newNode(1)
    root.right.left.right = newNode(2)
    root.right.right = newNode(5)
    root.right.right.right = newNode(2)
```

```
    k = 5
```

```
    printKPath(root, k)
```