Here I have given two solution

1. I am using one inner class as MaxNode, It has left and right node.
2. MAXPATHSUM2 Example for TopToBottom Solution and
3. MAXPATHSUM Example for BottomToTop Solution
4. Here I am giving the maximumpathSum method and Printpath method

\*\*\*\*\*\*\*\*\*\*\* This Inner class for DATA\*\*\*\*\*\*\*\*\*\*\*

**class** MaxNode{

**int** data;

MaxNode left;

MaxNode right;

**public** MaxNode(**int** data){

**this**.data=data;

left=**null**;

right=**null**;

}

}

\*\*\*\*\*\*\*\*\*\*\* BOTTOM TO TOP \*\*\*\*\*\*\*\*\*\*\*\*\*\*

OUTPUT

Max sum is 20

Print path is:

6 5 8 1

**public** **class** MaxPathSum {

MaxNode root;

**public** **static** **void** main(String[] args) {

MaxPathSum tree=**new** MaxPathSum();

tree.root = **new** MaxNode(1);

tree.root.left = **new** MaxNode(8);

tree.root.right = **new** MaxNode(9);

tree.root.left.left = **new** MaxNode(1);

tree.root.left.left.left = **new** MaxNode(2);

tree.root.left.left.left.left = **new** MaxNode(3);

tree.root.left.left.right = **new** MaxNode(5);

tree.root.left.right = **new** MaxNode(5);

tree.root.left.right.left = **new** MaxNode(2);

tree.root.left.right.right = **new** MaxNode(6);

**int** sum=tree.findmaxSum(tree.root);

System.***out***.println("Max sum is "+sum);

System.***out***.println("Print path is: ");

List<Integer> list = **new** ArrayList<>();

*printPath*(tree.root,sum,list);

}

**private** **static** **boolean** printPath(MaxNode node, **int** sum, List<Integer> list) {

**if**(sum==0){

**return** **true**;

}

**if**(node==**null**)

**return** **false**;

**boolean** left=*printPath*(node.left,sum-node.data, list);

**boolean** right=*printPath*(node.right,sum-node.data, list);

**if**(left||right){

System.***out***.print(node.data+" ");

}

**return** left||right;

}

**private** **int** findmaxSum(MaxNode node) {

**if**(node==**null**)

**return** 0;

**if**(node.data %2 ==0 && node.left != **null** && node.right!=**null** ){

**if**(node.left.data %2 ==0 && node.right.data %2 ==0)

**return** 0;

}

**if**(node.data %2 !=0 && node.left != **null** && node.right!=**null** ){

**if**(node.left.data %2 !=0 && node.right.data %2 !=0)

**return** 0;

}

**int** left=findmaxSum(node.left);

**int** right=findmaxSum(node.right);

**if**(node.data %2 ==0 && node.left != **null** && node.right!=**null**)

**if**(node.left.data %2!=0 && node.right.data%2 !=0)

**return** (left>right?left:right)+node.data;

**else** **if**(node.data %2 !=0 && node.left != **null** && node.right!=**null**)

**if**(node.left.data %2==0 && node.right.data%2==0)

**return** (left>right?left:right)+node.data;

**if**(node.data %2 ==0 && node.left != **null** && node.right!=**null**)

**if**(node.left.data %2==0 && node.right.data%2 !=0)

**return** right+node.data;

**else** **if**(node.data %2 ==0 && node.left != **null** && node.right!=**null**)

**if**(node.left.data %2!=0 && node.right.data%2 ==0)

**return** left+node.data;

**else** **if**(node.data %2 !=0 && node.left != **null** && node.right!=**null**)

**if**(node.left.data %2==0 && node.right.data%2 !=0)

**return** left+node.data;

**else** **if**(node.data %2 !=0 && node.left != **null** && node.right!=**null**)

**if**(node.left.data %2!=0 && node.right.data%2 ==0)

**return** right+node.data;

**return** (left>right?left:right)+node.data;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TOP TO BOTTOM \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

OUT PUT

Max sum is 20

Print path is:

1 8 5 6

**public** **class** MaxSumPath2 {

MaxNode root;

**public** **static** **void** main(String[] args) {

MaxSumPath2 tree=**new** MaxSumPath2();

tree.root = **new** MaxNode(1);

tree.root.left = **new** MaxNode(8);

tree.root.right = **new** MaxNode(9);

tree.root.left.left = **new** MaxNode(1);

tree.root.left.left.left = **new** MaxNode(2);

tree.root.left.left.left.left = **new** MaxNode(3);

tree.root.left.left.right = **new** MaxNode(5);

tree.root.left.right = **new** MaxNode(5);

tree.root.left.right.left = **new** MaxNode(2);

tree.root.left.right.right = **new** MaxNode(6);

**int** sum=tree.findmaxSum(tree.root);

System.***out***.println("Max sum is "+sum);

System.***out***.println("Print path is: ");

Stack<Integer> stack = **new** Stack<>();

*pushDataToStack*(tree.root,sum,stack);

}

**public** **static** **void** pushDataToStack(MaxNode node, **int** sum, Stack<Integer> stack){

**if**(sum!=0){

stack.push(node.data);

}

sum=sum - node.data;

**if**(sum==0){

Stack<Integer> reverse = **new** Stack<>();

**while**(!stack.empty()){

reverse.push(stack.pop());

}

**while**(!reverse.empty()){

System.***out***.print(reverse.pop()+" ");

}

}

**if**(node!=**null** && node.left !=**null**){

*pushDataToStack*(node.left,sum,stack);

}

**if**(node!=**null** && node.right!=**null**){

*pushDataToStack*(node.right,sum,stack);

}

**if**(!stack.isEmpty()){

stack.pop();

}

}

**private** **int** findmaxSum(MaxNode node) {

**if**(node==**null**)

**return** 0;

**if**(node.data %2 ==0 && node.left != **null** && node.right!=**null** ){

**if**(node.left.data %2 ==0 && node.right.data %2 ==0)

**return** 0;

}

**if**(node.data %2 !=0 && node.left != **null** && node.right!=**null** ){

**if**(node.left.data %2 !=0 && node.right.data %2 !=0)

**return** 0;

}

**int** left=findmaxSum(node.left);

**int** right=findmaxSum(node.right);

**if**(node.data %2 ==0 && node.left != **null** && node.right!=**null**)

**if**(node.left.data %2!=0 && node.right.data%2 !=0)

**return** (left>right?left:right)+node.data;

**else** **if**(node.data %2 !=0 && node.left != **null** && node.right!=**null**)

**if**(node.left.data %2==0 && node.right.data%2==0)

**return** (left>right?left:right)+node.data;

**if**(node.data %2 ==0 && node.left != **null** && node.right!=**null**)

**if**(node.left.data %2==0 && node.right.data%2 !=0)

**return** right+node.data;

**else** **if**(node.data %2 ==0 && node.left != **null** && node.right!=**null**)

**if**(node.left.data %2!=0 && node.right.data%2 ==0)

**return** left+node.data;

**else** **if**(node.data %2 !=0 && node.left != **null** && node.right!=**null**)

**if**(node.left.data %2==0 && node.right.data%2 !=0)

**return** left+node.data;

**else** **if**(node.data %2 !=0 && node.left != **null** && node.right!=**null**)

**if**(node.left.data %2!=0 && node.right.data%2 ==0)

**return** right+node.data;

**return** (left>right?left:right)+node.data;

}

}