Maze is just the beginning..

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are stuck in a maze in the form of a binary tree. To move ahead in your DSA course and face more challenges you have to cross this maze. You are at it's starting point and you can exit the maze from only those points which don't have any more paths branching out. Your initial energy level is the same as the value present at the starting point and you can move only in one direction towards exit points.

Energy boosters or energy depleters are present at each intersection of a new path. If x_i is present at the intersection, an energy booster (positive value) will increase your energy by x_i while an energy depleter (negative value) will decrease your energy by $|x_i|$ amounts.

You need to choose a path that *maximizes* your energy after you leave the maze so that you are prepared for more challenges ahead.

Print the maximized energy that you have at the end of the maze.

Note: If you face only obstacles in your path you still need to maximize your energy after you reach the exit point.

Input

A single line containing a string of nodes given in level order traversal of the tree where a number represent an energy booster/depleter and a character 'n' denotes a NULL child.

If number of points in maze are N then $1 \leq N \leq 10^5$

$-10^9 \leq x_i \leq 10^9$

The following code snippet can be used to build the tree in java: (the Node class consists of the fields- Node left, Node right and int data)

```
public static Node takeInput(){
Scanner sc = new Scanner(System.in);
Queue<Node> queue = new LinkedList<Node>();
String treeStr = sc.nextLine();
String[] tree = treeStr.split("\\s+");
if(tree[0].equals("n")){
return null;
}
Node root = new Node(Integer.parseInt(tree[0]));
queue.add(root);
int i = 1;
while(!queue.isEmpty() && i < tree.length){
Node front = queue.poll();
if(!tree[i].equals("n")){
int leftChild = Integer.parseInt(tree[i]);
Node left = new Node(leftChild);
front.left = left;
queue.add(left);
}
i++;
if(i >= tree.length) {
break;
}
if(!tree[i].equals("n")){
```

```
int rightChild = Integer.parseInt(tree[i]);
Node right = new Node(rightChild);
front.right = right;
queue.add(right);
}
i++;
}
return root;
}
```

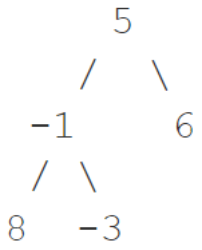
Output

A single integer denoting the maximum energy at the end of the maze.

Examples

standard input	standard output
5 -1 6 8 -3 n n	12
1 2 n 3 n	6

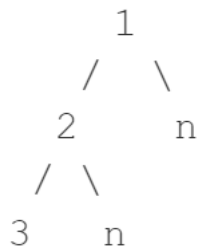
Note



3 paths are present: 5 -> 6

5 -> -1 -> -3

5 -> -1 -> 8 (Maximized energy path)



Path: 1 -> 2 -> 3