برای حل این مسئله می توانیم از الگوریتم جستجوی سطح_ترتیب استفاده کنیم و گره ها را به ترتیب در هر سطح چاپ کنیم. برای این کار از صف استفاده می کنیم که اولین عضو آن ریشه با عمق صفر است. درون حلقه فرزندان ریشه را به صف اضافه کرده و آن ها عمق یک را نسبت می دهیم تا در تکرار های بعدی همراه با عمقشان چاپ شوند. این روند را برای هر گره تکرار می کنیم تا به برگ ها برسیم و پیمایش سطح ترتیب به اتمام برسد.

PrintLevelOrder(*root*)

Require: Root of tree T
Ensure: Prints all nodes and their depth
```
queue = [ ]
root.depth = 0
append root to the queue
while queue is not empty do
    node = queue.pop()
    print node value and its depth
    if node has left child then
        node.left.depth = node.depth + 1
        append node left child to queue
    if node has right child then
        node.right.depth = node.depth + 1
        append node right child to queue
```

Listing 1: Python Implementation

```python
def print_depths(root):
    queue = []
    queue.append((root, 0))

    while len(queue) > 0:
        (node, depth) = queue.pop(0)
        print(node.data, depth)
        if node.right:
            queue.append((node.right, depth + 1))
        if node.left:
            queue.append((node.left, depth + 1))
```

این الگوریتم تمام گره های درخت را یک بار و فقط یک بار بررسی و چاپ می کند بنابراین کارایی زمانی آن $\theta(n)$ خواهد بود.