In [11]:
```
#Reflex Agent
class ModelBasedVaccumAgent():
    def __init__(self,init_a,init_b):
        self.model = {"Loc_a": init_a,"Loc_b": init_b}
    def DoAction(self,location,status):
        self.model[location] = status
        print(self.model)
        if self.model["Loc_a"] == self.model["Loc_b"] == "clean":
            return "NOOP"
        elif status == "dirty":
            return "Suck"
        elif location == "Loc_a":
            return "Right"
        else:
            return "Left"


locA=input("Enter status for Location A");
locB=input("Enter status for Location B");

a=ModelBasedVaccumAgent(locA,locB)
print(a.DoAction("Loc_A",locA))
```

```
Enter status for Location Aclean
Enter status for Location Bdirty
{'Loc_a': 'clean', 'Loc_b': 'dirty', 'Loc_A': 'clean'}
Left
```

In [12]:
```
 #BFS
graph = {
    '5' : ['3','7'],
    '3' : ['2','4'],
    '7' : ['8'],
    '2' : [],
    '4' : ['8'],
    '8' : []

}
visited = []
queue = []
def bfs(visited , graph, node):
    visited.append(node)
    queue.append(node)
    while queue:
        m= queue.pop(0)
        print(m,end=" ")
        for neighbour in graph[m]:
            if neighbour not in visited:
                visited.append(neighbour)
                queue.append(neighbour)

print("Following is the BFS")
bfs(visited,graph,'5')
```

```
Following is the BFS
5 3 7 2 4 8
```

In [ ]: