Poor PseudoCode A (unstructured and too far from real code)

The lack of structure makes this example difficult to read and understand.

matchPat(pat , str): if pat is empty return true, else if str is empty return false, else if head of pat is a star character, check the second letter of the pattern to through the string by using the head of the tail of pat. The program keeps track of the star and makes the star represent characters 0 up to the end of the string. It walks through the str as function searchStr does, until either finding a match or returning false. Else if the head is not a star then search through the str as the the function searchChar searches.

Poor PseudoCode B (too close to real code)

Just like Python code, which is hard to follow if you don't know Python.

```
def decide(nodelist, index):

    if nodelist[index][1] and nodelist[index][2] != -1:

        print(nodelist[index][0])
        answer = input("Answer with 'y' or 'n'")

        while answer != 'y' and answer != 'n':
            print(nodelist[index][0])
            answer = input("Try again: Answer with 'y' or 'n' only")

        if answer == 'y':
            decide(nodelist, nodelist[index][1])
        else:
            decide(nodelist, nodelist[index][2])
    else:
        print("Result:", nodelist[index][0])
```

Good PseudoCode (structured, English, & not a programming language)

Written in plain English with minimal Python syntax, it uses line formatting that's easy to follow, and conversion to code is easy.

```
decide(nodelist, index)
        if neither node is -1:
                print node string
                ask for input, make sure it is y or n
                if input is y:
                        call decide (nodelist, left node)
                else:
                        call decide (nodelist, right node)
        else:
                print node string
```

Make your pseudocode look like the example above. Don't write paragraphs or use real code.