Problem 1 : Trie – 1

```
class TrieNode:
    def __init__(self):
        self.children = {}
        self.is_end_of_word = False


class Trie:
    def __init__(self):
        self.root = TrieNode()


    def insert(self, word):
        node = self.root
        for char in word:
            if char not in node.children:
                node.children[char] = TrieNode()
            node = node.children[char]
        node.is_end_of_word = True


    def search(self, word):
        node = self.root
        for char in word:
            if char not in node.children:
                return False
            node = node.children[char]
        return node.is_end_of_word


    def starts_with(self, prefix):
        node = self.root
        for char in prefix:
            if char not in node.children:
                return False
```

```
            node = node.children[char]
        return True

type_input = [1, 1, 2, 3, 2]
value_input = ["hello", "help", "help", "hel", "hel"]
trie = Trie()
output = []

for type_op, value in zip(type_input, value_input):
    if type_op == 1:
        trie.insert(value)
        output.append(True)
    elif type_op == 2:
        output.append(trie.search(value))
    elif type_op == 3:
        output.append(trie.starts_with(value))

print(output)
```