Algorithms

Likelihood to show event on feed:

```
calculatePostScore(Post: post, likeWeight: float, commentWeight: float)

Return post.likes * likeWeight + post.comment * commentWeight
```

Sort by post score:

```
partitionPosts(Posts: List)

 LET pivot = end
 LET  pivotValue = pivot.score
 LET currentNode = start
 LET prevNode = null

WHILE currentNode != pivot:
            if currentNode.score < pivotValue:
           prevNode = currentNode
           SWAP(currentNode, prevNode)
            ENDIF
   currentNode = currentNode.next
ENDWHILE
            SWAP (currentNode, prevNode.next)
            return prevNode.next
```

```
sortPostsByScore(start: Post, end: Post)

 IF start != null AND start != end AND start != end.next:
       pivotNode = partitionPost(start, end)
ENDIF
sortPostsByScore(start, pivotNode)
sortPostsByScore(pivotNode.next, end)
```

Friend Suggestion:

```
friendSuggestion(friends: list)
LET friendOne = first friend in friends list
LET friendTwo = first friend in friends list
LET place = 0
FOR int x from 0 to the length of friends:
        FOR int y from x to the length of friends:
                IF sort criteria of friendOne < sort criteria of friend at index y
                        LET place = index y
                        LET friend2 = friend at index y
                ENDIF

                friendOne = friends[x];
                friends[x] = friendTwo;
                friends[place] = friendOne;

        END
END

return friends;
```

Check Capacity:

```
checkCapacity(numOfPeople: Int, listOfEvents: list)
LET max = maximum capacity of event
BOOLEAN accepted = false;
FOR int x from 0 to the length of listOfEvents:
        IF numOfPeople < max
                Accepted = TRUE;
                Return accepted;

        ENDIF
END

return accepted;
```