**Approach # 2: Runtime O(n), requires up to 2n for space complexity**

**//This is returns a boolean value if there exists a 2 integer subset whose sum is equal to //the passed in value “sum”**

**1 twoSum (numList, listLength, sum)**

**2 If listLength < 1**

**3 Then return false // if empty list return false**

**4 EndIf**

**5**

**6** Initialize a hash table **T** // create a hash table, preferably with chaining

**7 for** i ← 1 **to** listLength //for each element in the list

**8 If** T does not contain (sum - numList[i]) //check for its counterpart

**9 then** insert numList[i] **into** T// if no counterpart, insert into hash table

**10 Else return** true // else the counter part was found

**11 EndIf** // meaning two elements add to “sum”

**12** // return true

**13**

**14 return false** // if end of list is reached, return false